IBM® DB2® Universal Database

# Command Reference

*Version 8.2*

SC09-4828-01

IBM® DB2® Universal Database

# Command Reference

*Version 8.2*

SC09-4828-01

Before using this information and the product it supports, be sure to read the general information under *Notices*.

You can order IBM publications online or through your local IBM representative.

- To order publications online, go to the IBM Publications Center at www.ibm.com/shop/publications/order
- To find your local IBM representative, go to the IBM Directory of Worldwide Contacts at www.ibm.com/planetwide

To order DB2 publications from DB2 Marketing and Sales in the United States or Canada, call 1-800-IBM-4YOU (426-4968).

# Contents

# About This Book

This book provides information about the use of system commands and the IBM DB2 Universal Database command line processor (CLP) to execute database administrative functions.

## Who Should Use this Book

It is assumed that the reader has an understanding of database administration and a knowledge of Structured Query Language (SQL).

## How this Book is Structured

This book provides the reference information needed to use the CLP.

The following topics are covered:

**Chapter 1**
Describes the commands that can be entered at an operating system command prompt or in a shell script to access the database manager.

**Chapter 2**
Explains how to invoke and use the command line processor, and describes the CLP options.

**Chapter 3**
Provides a description of all database manager commands.

**Chapter 4**
Provides information on how to use SQL statements from the command line.

**Appendix A**
Explains the conventions used in syntax diagrams.

**Appendix B**
Explains the conventions used to name objects such as databases and tables.

# Chapter 1. System Commands

This chapter provides information about the commands that can be entered at an operating system command prompt, or in a shell script, to access and maintain the database manager.

**Notes:**

1. Slashes (/) in directory paths are specific to UNIX based systems, and are equivalent to back slashes (\) in directory paths on Windows operating systems.
2. The term Windows normally refers to all supported versions of Microsoft Windows. Supported versions include those versions based on Windows NT and those based on Windows 9x. Specific references to ″Windows NT-based operating systems″ may occur when the function in question is supported on Windows NT 4, Windows 2000, Windows .NET and Windows XP but not on Windows 9x. If there is a function that is specific to a particular version of Windows, the valid version or versions of the operating system will be noted.

## How the command descriptions are organized

A short description of each command precedes some or all of the following subsections.

**Scope:**

The command's scope of operation within the instance. In a single-database-partition system, the scope is that single database partition only. In a multi-database-partition system, it is the collection of all logical database partitions defined in the database partition configuration file, `db2nodes.cfg`.

**Authorization:**

The authority required to successfully invoke the command.

**Required connection:**

One of the following: database, instance, none, or establishes a connection. Indicates whether the function requires a database connection, an instance attachment, or no connection to operate successfully. An explicit connection to the database or attachment to the instance may be required before a particular command can be issued. Commands that require a database connection or an instance attachment can be executed either locally or remotely. Those that require neither cannot be executed remotely; when issued at the client, they affect the client environment only.

**Command syntax:**

A syntax diagram shows how a command should be specified so that the operating system can correctly interpret what is typed. For more information about syntax diagrams, see Appendix A, "How to read the syntax diagrams," on page 751.

**Command parameters:**

A description of the parameters available to the command.

**Usage notes:**

Other information.

**Related reference:**

A cross-reference to related information.

## dasauto - Autostart DB2 Administration Server

Enables or disables autostarting of the DB2 administration server.

This command is available on UNIX-based systems only. It is located in the `DB2DIR/das/adm` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems.

**Authorization:**

*dasadm*

**Required connection:**

None

**Command syntax:**

```
►►──dasauto──┬──────┬──┬─-on──┬──────────────────────────────────────────────►◄
             ├─-h───┤  └─-off─┘
             └─-?───┘
```

**Command parameters:**

**-h/-?** Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-on** Enables autostarting of the DB2 administration server. The next time the system is restarted, the DB2 administration server will be started automatically.

**-off** Disables autostarting of the DB2 administration server. The next time the system is restarted, the DB2 administration server will not be started automatically.

# dascrt - Create a DB2 Administration Server

The DB2 administration server (DAS) provides support services for DB2 tools such as the Control Center and the Configuration Assistant. If a system does not have a DAS, you can use this command to manually generate it.

This command is available on UNIX-based systems only. On Windows systems, you can use the command **db2admin create** for the same purpose.

**Authorization:**

`Root` authority.

**Required connection:**

None.

**Command syntax:**

```
►►──dascrt──-u──DASuser──────────────────────────────────────────────►◄
```

**Command parameters:**

**-u** *DASuser*
: *DASuser* is the user ID under which the DAS will be created. The DAS will be created under the /home/*DASuser*/das directory.

**Usage notes:**

- In previous versions of DB2, this command was known as **dasicrt**.
- The **dascrt** command is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems.

# dasdrop - Remove a DB2 Administration Server

On UNIX operating systems only, removes the DB2 Administration Server (DAS). The Administration Server provides support services for DB2 tools such as the Control Center and the Configuration Assistant.

**Authorization:**

`Root` authority.

**Required Connection:**

None.

**Command Syntax:**

```
►►──dasdrop──────────────────────────────────────────────────────────►◄
```

**Usage Notes:**

- The dasdrop command is found in the `instance` subdirectory under the subdirectory specific to the installed DB2 version and release.
- If you have a FixPak or modification level installed in an alternate path, you can drop any DAS by running the dasdrop utility from an installation path; to do this, the install code must still be located in the installation path of the DAS that you are dropping. If you remove install code from an installation path, and then try to drop the DAS in that path by invoking the dasdrop utility from a different installation path, you will not be able to drop the DAS.

**Related tasks:**

- "Removing the DAS" in the *Administration Guide: Implementation*

## dasmigr - Migrate the DB2 Administration Server

Migrates the DB2 administration server following installation.

On UNIX-based systems, this utility is located in the `DB2DIR/instance` directory, where DB2DIR represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX based systems. On Windows operating systems, it is located in the `sqllib\bin` subdirectory.

**Authorization:**

`Root` access on UNIX-based systems or Local Administrator authority on Windows operating systems.

**Required connection:**

None.

**Command syntax:**

On UNIX:

```
►►──dasmigr──previous_das_name──new_das_name──────────────────────────────►◄
```

On Windows:

```
►►──dasmigr──new_das_name─────────────────────────────────────────────────►◄
```

**Command parameters:**

*previous_das_name*
: Name of the DAS in the version you are migrating from. This parameter is not valid on Windows.

*new_das_name*
: Name of the DAS in the version you are migrating to.

**Examples:**

```
   dasmigr db2as dasusr1
```

**Usage notes:**

Migrating the DB2 administration server requires that a tools catalog database be created and available for connection.

**Related tasks:**

- "Configuring the DAS" in the *Administration Guide: Implementation*
- "Migrating the DB2 Administration Server (DAS)" in the *Quick Beginnings for DB2 Servers*

**Related reference:**

- "CREATE TOOLS CATALOG" on page 339

# dasupdt - Update DAS

On UNIX-based operating systems, if DB2 is updated by installing a Program Temporary Fix (PTF) or a code patch, **dasupdt** updates each DB2 Administration Server (DAS). It is located in DB2DIR/`instance`, where DB2DIR represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems.

**Authorization:**

`Root` authority.

**Required connection:**

None

**Command syntax:**

```
►►──dasupdt──┬────┬──┬────┬──┬────┬──────────────────────────────────────────►◄
             └─-d─┘  └─-D─┘  ├─-h─┤
                             └─-?─┘
```

**Command parameters:**

**-d** Sets the debug mode, which is used for problem analysis.

**-D** Moves the DAS from a higher code level on one path to a lower code level installed on another path.

**-h/-?** Displays usage information.

**Examples:**

The DAS is running Version 8.1.2 code in the Version 8 install path. If FixPak 3 is installed in the Version 8 install path, the following command, invoked from the Version 8 install path, will update the DAS to FixPak 3;

```
dasupdt
```

The DAS is running Version 8.1.2 code in an alternate install path. If FixPak 1 is installed in another alternate install path, the following command, invoked from the FixPak 1 alternate install path, will update the DAS to FixPak 1, running from the FixPak 1 alternate install path:

```
dasupdt -D
```

## db2admin - DB2 Administration Server

This utility is used to manage the DB2 Administration Server.

**Authorization:**

Local administrator on Windows, or DASADM on UNIX based systems.

**Required connection:**

None

**Command syntax:**

```
►►──db2admin────────────────────────────────────────────────────────────────►

►──┬─────────────────────────────────────────────────────────────┬──────────►◄
   ├─START───────────────────────────────────────────────────────┤
   ├─STOP──┬──────────┬──────────────────────────────────────────┤
   │       └─/FORCE───┘                                          │
   ├─CREATE──┬─────────────────────────┬──┬──────────────────────────────┬─┤
   │         └─/USER:──user-account────┘  └─/PASSWORD:──user-password───┘ │
   ├─DROP────────────────────────────────────────────────────────┤
   ├─SETID──user-account──user-password──────────────────────────┤
   ├─SETSCHEDID──sched-user──sched-password──────────────────────┤
   ├─-?──────────────────────────────────────────────────────────┤
   └─-q──────────────────────────────────────────────────────────┘
```

**Command parameters:**

**Note:** If no parameters are specified, and the DB2 Administration Server exists, this command returns the name of the DB2 Administration Server.

**START**
Start the DB2 Administration Server.

**STOP /FORCE**
Stop the DB2 Administration Server. The force option is used to force the DB2 Administration Server to stop, regardless of whether or not it is in the process of servicing any requests.

**CREATE** */USER: user-account /PASSWORD: user-password*
Create the DB2 Administration Server. If a user name and password are specified, the DB2 Administration Server will be associated with this user account. If the specified values are not valid, the utility returns an authentication error. The specified user account must be a valid SQL identifier, and must exist in the security database. It is recommended that a user account be specified to ensure that all DB2 Administration Server functions can be accessed.

**Note:** To create a DAS on UNIX systems, use the dascrt command.

**DROP** Deletes the DB2 Administration Server.

**Note:** To drop a DAS on UNIX you must use the dasdrop command.

**SETID** *user-account/user-password*
Establishes or modifies the user account associated with the DB2 Administration Server.

**SETSCHEDID** *sched-user/sched-password*
Establishes the logon account used by the scheduler to connect to the tools catalog database. Only required if the scheduler is enabled and the tools catalog database is remote to the DB2 Administration Server. For more information about the scheduler, see the *Administration Guide*.

**-?** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-q** Run the db2admin command in quiet mode. No messages will be displayed when the command is run. This option can be combined with any of the other command options.

**Usage notes:**

On UNIX-based operating systems, the executable file for the db2admin command can be found in the `home/DASuser/das/bin` directory, where `DASuser` is the name of the DB2 Administration Server user. On Windows, the db2admin executable is found under the `sqllib/bin` directory.

**Related reference:**

- "dasdrop - Remove a DB2 Administration Server" on page 5
- "dascrt - Create a DB2 Administration Server" on page 4

# db2adutl - Managing DB2 objects within TSM

Allows users to query, extract, verify, and delete backup images, logs, and load copy images saved using Tivoli Storage Manager. Also allows users to grant and revoke access to objects on a TSM server.

On UNIX-based operating systems, this utility is located in the `sqllib/adsm` directory. On Windows it is located in `sqllib\bin`.

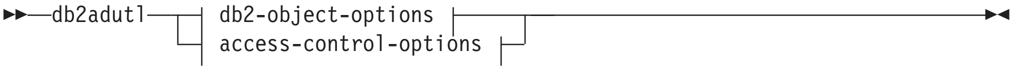
**Authorization:**

None

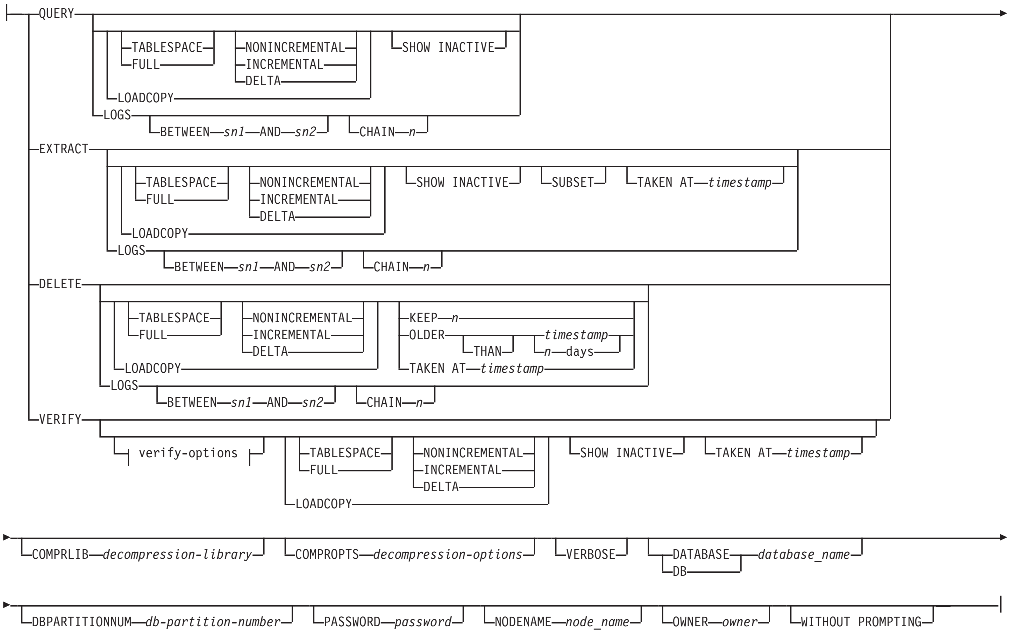
**Required connection:**

None

**Command syntax:**



## db2-object-options:



## verify-options:

```
├──┬─ALL──────────────┬─────────────────────────────────────────────┤
   ├─CHECK────────────┤
   ├─DMS──────────────┤
   ├─HEADER───────────┤
   ├─LFH──────────────┤
   ├─TABLESPACES──────┤
   ├─HEADERONLY───────┤
   ├─TABLESPACESONLY──┤
   ├─OBJECT───────────┤
   └─PAGECOUNT────────┘
```

### access-control-options:

```
├──┬─GRANT───┬─ALL──────────────┬──ON──┬─ALL────────────────────┬──FOR──┬─DATABASE─┬──database_name──────┬──►
   │         └─USER──user_name──┘      └─NODENAME──node_name────┘       └─DB───────┘                     │
   ├─REVOKE──┬─ALL──────────────┬──ON──┬─ALL────────────────────┬──FOR──┬─ALL────────────────────────────┤
   │         └─USER──user_name──┘      └─NODENAME──node_name────┘       └─┬─DATABASE─┬──database_name──┘ │
   │                                                                      └─DB───────┘                    │
   └─QUERYACCESS──FOR──┬─ALL──────────────────────────────────────────────────────────────────────────────┘
                       └─┬─DATABASE─┬──database_name──┘
                         └─DB───────┘

►──┬──────────────────────┬─────────────────────────────────────────────────────────────────────────────┤
   └─PASSWORD──password───┘
```

**Command parameters:**

**QUERY**
Queries the TSM server for DB2 objects.

**EXTRACT**
Copies DB2 objects from the TSM server to the current directory on the local machine.

**DELETE**
Either deactivates backup objects or deletes log archives on the TSM server.

**VERIFY**
Performs consistency checking on the backup copy that is on the server.

**Note:** This parameter causes the entire backup image to be transferred over the network.

**ALL** Displays all available information.

**CHECK**
Displays results of checkbits and checksums.

**DMS** Displays information from headers of DMS table space data pages.

**HEADER**
Displays the media header information.

**HEADERONLY**
Displays the same information as HEADER but only reads the 4 K media header information from the beginning of the image. It does not validate the image.

**LFH** Displays the log file header (LFH) data.

**OBJECT**
Displays detailed information from the object headers.

**PAGECOUNT**
Displays the number of pages of each object type found in the image.

**TABLESPACES**
Displays the table space details, including container information, for the table spaces in the image.

**TABLESPACESONLY**
Displays the same information as TABLESPACES but does not validate the image.

**TABLESPACE**
Includes only table space backup images.

**FULL** Includes only full database backup images.

**NONINCREMENTAL**
Includes only non-incremental backup images.

**INCREMENTAL**
Includes only incremental backup images.

**DELTA**
Includes only incremental delta backup images.

**LOADCOPY**
Includes only load copy images.

**LOGS** Includes only log archive images

**BETWEEN** *sn1* **AND** *sn2*
Specifies that the logs between log sequence number 1 and log sequence number 2 are to be used.

**CHAIN** *n*
Specifies the chain ID of the logs to be used.

**SHOW INACTIVE**
Includes backup objects that have been deactivated.

**SUBSET**
Extracts pages from an image to a file. To extract pages, you will need an input and an output file. The default input file is called extractPage.in. You can override the default input file name by setting the DB2LISTFILE environment variable to a full path. The format of the input file is as follows:

For SMS table spaces:

```
S <tbspID> <objID> <objType> <startPage> <numPages>
```

For DMS table spaces:

```
D <tbspID> <objType> <startPage> <numPages>
```

**Note:** <objType> is only needed if verifying DMS load copy images
For log files:

```
L <log num> <startPos> <numPages>
```

For other data (for example, initial data):

```
O <objType> <startPos> <numBytes>
```

The default output file is extractPage.out. You can override the default output file name by setting the DB2EXTRACTFILE environment variable to a full path.

**TAKEN AT** *timestamp*
Specifies a backup image by its time stamp.

**KEEP** *n*
Deactivates all objects of the specified type except for the most recent *n* by time stamp.

**OLDER THAN** *timestamp* **or** *n* **days**
Specifies that objects with a time stamp earlier than *timestamp* or *n* days will be deactivated.

**COMPRLIB** *decompression-library*
Indicates the name of the library to be used to perform the decompression. The name must be a fully qualified path referring to a file on the server. If this parameter is not specified, DB2 will attempt to use the library stored in the image. If the backup was not compressed, the value of this parameter will be ignored. If the specified library cannot be loaded, the operation will fail.

**COMPROPTS** *decompression-options*
Describes a block of binary data that will be passed to the initialization routine in the decompression library. DB2 will pass this string directly from the client to the server, so any issues of byte reversal or code page conversion will have to be handled by the decompression library. If the first character of the data block is '@', the remainder of the data will be interpreted by DB2 as the name of a file residing on the server. DB2 will then replace the contents of the data block with the contents of this file and will pass this new value to the initialization routine instead. The maximum length for this string is 1024 bytes.

**DATABASE** *database_name*
Considers only those objects associated with the specified database name.

**DBPARTITIONNUM** *db-partition-number*
Considers only those objects created by the specified database partition number.

**PASSWORD** *password*
Specifies the TSM client password for this node, if required. If a database is specified and the password is not provided, the value specified for the *tsm_password* database configuration parameter is passed to TSM; otherwise, no password is used.

**NODENAME** *node_name*
Considers only those images associated with a specific TSM node name.

**OWNER** *owner*
Considers only those objects created by the specified owner.

**WITHOUT PROMPTING**
The user is not prompted for verification before objects are deleted.

**VERBOSE**
Displays additional file information.

**GRANT ALL / USER** *user_name*
Adds access rights to the TSM files on the current TSM node to all users or

to the users specified. Granting access to users gives them access for all current and future files related to the database specified.

**REVOKE ALL / USER** *user_name*
Removes access rights to the TSM files on the current TSM node from all users or to the users specified.

**QUERYACCESS**
Retrieves the current access list. A list of users and TSM nodes is displayed.

**ON ALL / NODENAME** *node_name*
Specifies the TSM node for which access rights will be changed.

**FOR ALL / DATABASE** *database_name*
Specifies the database to be considered.

**Examples:**

1. The following is sample output from the command `db2 backup database rawsampl use tsm`

```
Backup successful. The timestamp for this backup is : 20031209184503
```

The following is sample output from the command `db2adutl query` issued following the backup operation:

```
Query for database RAWSAMPL

Retrieving FULL DATABASE BACKUP information.
    1 Time: 20031209184403, Oldest log: S0000050.LOG, Sessions: 1

Retrieving INCREMENTAL DATABASE BACKUP information.
  No INCREMENTAL DATABASE BACKUP images found for RAWSAMPL

Retrieving DELTA DATABASE BACKUP information.
  No DELTA DATABASE BACKUP images found for RAWSAMPL

Retrieving TABLESPACE BACKUP information.
  No TABLESPACE BACKUP images found for RAWSAMPL

Retrieving INCREMENTAL TABLESPACE BACKUP information.
  No INCREMENTAL TABLESPACE BACKUP images found for RAWSAMPL

Retrieving DELTA TABLESPACE BACKUP information.
  No DELTA TABLESPACE BACKUP images found for RAWSAMPL

Retrieving LOCAL COPY information.
  No LOCAL COPY images found for RAWSAMPL

Retrieving log archive information.
   Log file: S0000050.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.46.13
   Log file: S0000051.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.46.43
   Log file: S0000052.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.47.12
   Log file: S0000053.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.50.14
   Log file: S0000054.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.50.56
   Log file: S0000055.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.52.39
```

2. The following is sample output from the command `db2adutl delete full taken at 20031209184503 db rawsampl`

```
Query for database RAWSAMPL

Retrieving FULL DATABASE BACKUP information.
  Taken at: 20031209184503  DB Partition Number: 0    Sessions: 1

  Do you want to delete this file (Y/N)? y

    Are you sure (Y/N)? y


Retrieving INCREMENTAL DATABASE  BACKUP information.
  No INCREMENTAL DATABASE BACKUP images found for RAWSAMPL

Retrieving DELTA DATABASE  BACKUP information.
  No DELTA DATABASE BACKUP images found for RAWSAMPL
```

The following is sample output from the command `db2adutl query` issued following the operation that deleted the full backup image. Note the timestamp for the backup image.

```
Query for database RAWSAMPL

Retrieving FULL DATABASE BACKUP information.
    1 Time: 20031209184403, Oldest log: S0000050.LOG, Sessions: 1

Retrieving INCREMENTAL DATABASE BACKUP information.
  No INCREMENTAL DATABASE BACKUP images found for RAWSAMPL

Retrieving DELTA DATABASE BACKUP information.
  No DELTA DATABASE BACKUP images found for RAWSAMPL

Retrieving TABLESPACE BACKUP information.
  No TABLESPACE BACKUP images found for RAWSAMPL

Retrieving INCREMENTAL TABLESPACE BACKUP information.
  No INCREMENTAL TABLESPACE BACKUP images found for RAWSAMPL

Retrieving DELTA TABLESPACE BACKUP information.
  No DELTA TABLESPACE BACKUP images found for RAWSAMPL

Retrieving LOCAL COPY information.
  No LOCAL COPY images found for RAWSAMPL

Retrieving log archive information.
   Log file: S0000050.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.46.13
   Log file: S0000051.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.46.43
   Log file: S0000052.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.47.12
   Log file: S0000053.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.50.14
   Log file: S0000054.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.50.56
   Log file: S0000055.LOG, Chain Num: 0, DB Partition Number: 0,
    Taken at 2003-12-09-18.52.39
```

3. The following is sample output from the command `db2adutl queryaccess for all`

```
Node                User                Database Name      type
-------------------------------------------------------------------
bar2                jchisan             sample             B
<all>               <all>               test               B
-------------------------------------------------------------------
Access Types: B — Backup images  L — Logs  A - both
```

**Usage Notes:**

One parameter from each group below can be used to restrict what backup images types are included in the operation:

**Granularity:**

- FULL - include only database backup images.
- TABLESPACE - include only table space backup images.

**Cumulativeness:**

- NONINCREMENTAL - include only non-incremental backup images.
- INCREMENTAL - include only incremental backup images.
- DELTA - include only incremental delta backup images.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related concepts:**

- "Cross-node recovery with the db2adutl command and the logarchopt1 and vendoropt database configuration parameters" in the *Administration Guide: Performance*

## db2advis - DB2 Design Advisor

Advises users on the creation of materialized query tables (MQTs) and indexes, the repartitioning of tables, the conversion to multidimensional clustering (MDC) tables, and the deletion of unused objects. The recommendations are based on one or more SQL statements provided by the user. A group of related SQL statements is known as a *workload*. Users can rank the importance of each statement in a workload, and specify the frequency at which each statement in the workload is to be executed. The Design Advisor outputs a DDL CLP script that includes CREATE INDEX, CREATE SUMMARY TABLE (MQT), and CREATE TABLE statements to create the recommended objects.

Structured type columns are not considered when this command is executed.

**Authorization:**

Read access to the database. Read and write access to the explain tables.

**Required connection:**

None. This command establishes a database connection.

**Command syntax:**

```
►►──db2advis──┬─-d──┬──database-name──┬────────────────────┬─────────────────►
              └─-db─┘                 ├─-w──workload-name──┤
                                      ├─-s──"statement"────┤
                                      ├─-i──filename───────┤
                                      ├─-g─────────────────┤
                                      └─-qp────────────────┘

►──┬─────────────────────────┬──┬─────────────────┬──┬────┬──┬────┬──────────►
   └─-a──userid──┬─────────┬─┘  └─-m──advise-type─┘  └─-x─┘  └─-u─┘
                 └─/passwd─┘

►──┬─────────────────┬──┬────┬──┬──────────────────────┬──┬──────────┬───────►
   └─-l──disk-limit──┘  └─-n─┘  └─-t──max-advise-time──┘  └─-k──┬─HIGH─┬─┘
                                                                ├─MED──┤
                                                                ├─LOW──┤
                                                                └─OFF──┘

►──┬────┬──┬────┬──┬──────────────────┬──┬──────────────────┬────────────────►
   └─-f─┘  └─-r─┘  └─-n──schema-name──┘  └─-q──schema-name──┘

►──┬───────────────────────┬──┬───────────────────────┬──┬────┬──┬────┬──────►
   └─-b──tablespace-name───┘  └─-c──tablespace-name───┘  └─-h─┘  └─-p─┘

►──┬──────────────┬──────────────────────────────────────────────────────────►◄
   └─-o──outfile──┘
```

**Command parameters:**

**-d** *database-name*
: Specifies the name of the database to which a connection is to be established.

**-w** *workload-name*
: Specifies the name of the workload for which indexes are to be advised.

This name is used in the ADVISE_WORKLOAD table. This option cannot be specified with the -g, -i, qp, or -s options.

**-s** *"statement"*
Specifies the text of a single SQL statement whose indexes are to be advised. The statement must be enclosed by double quotation marks. This option cannot be specified with the -g, -i, -qp, or -w options.

**-i** *filename*
Specifies the name of an input file containing one or more SQL statements. The default is standard input. Identify comment text with two hyphens at the start of each line; that is, `-- <comment>`. Statements must be delimited by semicolons.

The frequency at which each statement in the workload is to be executed can by changed by inserting the following line into the input file:

```
--#SET FREQUENCY <x>
```

The frequency can be updated any number of times in the file. This option cannot be specified with the -g, -s, -qp, or -w options.

**-g** Specifies the retrieval of the SQL statements from a dynamic SQL snapshot. If combined with the -p command parameter, the SQL statements are kept in the ADVISE_WORKLOAD table. This option cannot be specified with the -i, -s, -qp, or -w options.

**-qp** Specifies that the workload is coming from Query Patroller. This option cannot be used with the -w, -s, -i, or -g options.

**-a** *userid/passwd*
Name and password used to connect to the database. The slash (/) must be included if a password is specified. A password should not be specified if the -x option is specified.

**-m** *advise-type*
Specifies the type of recommendation the advisor will return. Any combination of I, M, C, and P can be specified. The values must be entered in upper case. For example, **db2advis -m PC** will recommend partitioning and MDC tables.

**I** Recommends new indexes. This is the default.

**M** Recommends new materialized query tables (MQTs) and indexes on the MQTs. In partitioned database environments, partitioning on MQTs is also recommended.

**C** Recommends the conversion of standard tables to multidimensional clustering (MDC) tables.

**P** Recommends the repartitioning of existing tables.

**-x** Specifies that the password will be read from the terminal or through user input.

**-u** Specifies that the advisor will consider the recommendation of deferred MQTs. Incremental MQTs will not be recommended. When this option is specified, comments in the DDL CLP script indicate which of the MQTs could be converted to immediate MQTs. If immediate MQTs are recommended in a partitioned database environment, the default partitioning key is the implied unique key for the MQT.

**-l** *disk-limit*

Specifies the number of megabytes available for all indexes in the existing schema. Specify -1 to use the maximum possible size. The default value is 20% of the total database size.

**-t** *max-advise-time*

Specifies the maximum allowable time, in minutes, to complete the operation. If no value is specified for this option, the operation will continue until it is completed. To specify an unlimited time enter a value of zero. The default is zero.

**-k**

Specifies to what degree the workload will be compressed. Compression is done to allow the advisor to reduce the complexity of the advisor's execution while achieving similar results to those the advisor could provide when the full workload is considered. HIGH indicates the advisor will concentrate on a small subset of the workload. MED indicates the advisor will concentrate on a medium-sized subset of the workload. LOW indicates the advisor will concentrate on a larger subset of the workload. OFF indicates that no compression will occur. The default is MED.

**-f**

Drops previously existing simulated catalog tables.

**-r**

Specifies that detailed statistics should be used for the virtual MQTs and for the partitioning selection. If this option is not specified, the default is to use optimizer statistics for MQTs. Note that although the detailed statistics might be more accurate, the time to derive them will be significant and will cause the **db2advis** execution time to be greater.

**-n** *schema-name*

Specifies the qualifying name of simulation catalog tables, and the qualifier for the new indexes and MQTs. The default schema name is the caller's user ID, except for catalog simulation tables where the default schema name is SYSTOOLS.

**-q** *schema-name*

Specifies the qualifying name of unqualified names in the workload. It serves as the schema name to use for CURRENT SCHEMA when **db2advis** executes. The default schema name is the user ID of the person executing the command.

**-b** *tablespace-name*

Specifies the name of a table space in which new MQTs will be created. If not specified, the advisor will select the table spaces from the set of table spaces that exist.

**-c** *tablespace-name*

Specifies the name of a table space (file name or directory) in which to create the simulation catalog table space on the catalog database partition group. The default is USERSPACE1.

It is recommended that the user create the table space employed for the simulation instead of using the default USERSPACE1. In addition, the ALTER TABLESPACE DROPPED TABLE RECOVERY OFF statement should be run on this table space to improve the performance of the **db2advis** utility. When the utility completes, turn the history back on for the table space. In a partitioned database environment, the user-created table space must be created only on the catalog partition of the database.

**-h**

Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-p** Keeps the plans that were generated while running the tool in the explain tables.

**-o** *outfile*
Saves the script to create the recommended objects in `outfile`.

**Examples:**

1. In the following example, the utility connects to database PROTOTYPE, and recommends indexes for table ADDRESSES without any constraints on the solution:

```
db2advis -d prototype -s "select * from addresses a
   where a.zip in ('93213', '98567', '93412')
   and (company like 'IBM%' or company like '%otus')"
```

2. In the following example, the utility connects to database PROTOTYPE, and recommends indexes that will not exceed 53MB for queries in table ADVISE_WORKLOAD. The workload name is equal to ″production″. The maximum allowable time for finding a solution is 20 minutes.

```
db2advis -d prototype -w production -l 53 -t 20
```

3. In the following example, the input file `db2advis.in` contains SQL statements and a specification of the frequency at which each statement is to be executed:

```
--#SET FREQUENCY 100
SELECT COUNT(*) FROM EMPLOYEE;
SELECT * FROM EMPLOYEE WHERE LASTNAME='HAAS';
--#SET FREQUENCY 1
SELECT AVG(BONUS), AVG(SALARY) FROM EMPLOYEE
   GROUP BY WORKDEPT ORDER BY WORKDEPT;
```

The utility connects to database SAMPLE, and recommends indexes for each table referenced by the queries in the input file. The maximum allowable time for finding a solution is 5 minutes:

```
db2advis -d sample -f db2advis.in -t 5
```

4. In the following example, MQTs are created in table space SPACE1 and the simulation table space is SPACE2. The qualifying name for unqualified names in the workload is SCHEMA1, and the schema name in which the new MQTs will be recommended is SCHEMA2. The workload compression being used is HIGH and the disk space is unlimited. Sample statistics are used for the MQTs. Issuing the following command will recommend MQTs and, in a partitioned database environment, indexes and partitioning will also be recommended.

```
db2advis -d prototype -w production -l -1 -m M -b space1 -c space2 -k
   HIGH -q schema1 -n schema2 -r
```

To get the recommended MQTs, as well as indexes, partitioning and MDCs on both MQT and base tabes, issue the command specifying a value of IMCP for the -m option as follows:

```
db2advis -d prototype -w production -l -1 -m IMCP -b space1 -c space2 -k
   HIGH -q schema1 -n schema2 -r
```

**Usage notes:**

Because these features must be set up before you can run the DDL CLP script, database partitioning, multi-dimensional clustering, and clustered index recommendations are commented out of the DDL CLP script that is returned.

For dynamic SQL statements, the frequency with which statements are executed can be obtained from the monitor as follows:

1. Issue

```
db2 reset monitor for database <database-alias>
```

Wait for an appropriate interval of time.

2. Issue

```
db2advis -g <other-options>
```

If the -p parameter is used with the -g parameter, the dynamic SQL statements obtained will be placed in the ADVISE_WORKLOAD table with a generated workload name that contains a timestamp.

The default frequency for each SQL statement in a workload is 1, and the default importance is also 1. The generate_unique() function assigns a unique identifier to the statement, which can be updated by the user to be a more meaningful description of that SQL statement.

**Related concepts:**

- “The Design Advisor” in the *Administration Guide: Performance*

## db2atld - Autoloader

Autoloader is a tool for partitioning and loading data in an MPP environment. This utility can:

- Transfer data from one system (MVS, for example) to an AIX system (RS/6000 or SP2)
- Partition data in parallel
- Load data simultaneously on corresponding database partitions.

**Related reference:**

- "LOAD" on page 520

# db2audit - Audit Facility Administrator Tool

DB2 provides an audit facility to assist in the detection of unknown or unanticipated access to data. The DB2 audit facility generates and permits the maintenance of an audit trail for a series of predefined database events. The records generated from this facility are kept in an audit log file. The analysis of these records can reveal usage patterns which would identify system misuse. Once identified, actions can be taken to reduce or eliminate such system misuse. The audit facility acts at an instance level, recording all instance level activities and database level activities.

Authorized users of the audit facility can control the following actions within the audit facility, using **db2audit**:

- Start recording auditable events within the DB2 instance.
- Stop recording auditable events within the DB2 instance.
- Configure the behavior of the audit facility.
- Select the categories of the auditable events to be recorded.
- Request a description of the current audit configuration.
- Flush any pending audit records from the instance and write them to the audit log.
- Extract audit records by formatting and copying them from the audit log to a flat file or ASCII delimited files. Extraction is done for one of two reasons: In preparation for analysis of log records, or in preparation for pruning of log records.
- Prune audit records from the current audit log.

# db2batch - Benchmark Tool

Reads SQL statements from either a flat file or standard input, dynamically prepares and describes the statements, and returns an answer set.

This tool can work in both a single partition database and in a multiple partition database.

Through the tool's optional parameters you are able to control the number of rows to be fetched from the answer set, the number of fetched rows to be sent to the output file or standard output, and the level of performance information to be returned.

The output default is to use standard output. You can name the output file for the results summary. When you are working in a partitioned database and you use the -r option to name the output file, the output from each database partition goes into a separate file with the same name on each database partition. The exception occurs when the file specified is on an NFS-mounted file system. When this is the case, in a multiple partitioned database, all of the results are kept in this file.

**Authorization:**

The same authority level as that required by the SQL statements to be read.

In parallel mode, users must have the authorization to run **db2_all**.

**Required connection:**

None. This command establishes a database connection.

**Command syntax:**

```
►►──db2batch──┬─────────────┬──┬────────────────┬──┬─────────────────────┬──────────►
              └─-d──dbname──┘  └─-f──file_name──┘  └─-a──userid/passwd──┘

►──┬─────────────┬──┬──────────────────────────────┬──┬─────────────────┬───────────►
   └─-t──delcol──┘  └─-r──outfile──┬──────────────┬┘  │      ┌─on──┐    │
                                   └─,outfile2────┘   └─-c───┴─off─┴────┘

►──┬───────────────────────┬──┬──────────────┬──┬─────────────────┬──┬─────────────────┬──►
   │     ┌─short─────┐     │  └─-o──options──┘  │      ┌─off─┐    │  │      ┌─on──┐    │
   └─-i──┼─long──────┼─────┘                    └─-v───┴─on──┴────┘  └─-s───┴─off─┴────┘
         └─complete──┘

►──┬──────────────────┬──┬──────────┬──┬────────────────────┬──┬───────────────────────────┬──►
   │      ┌─off─┐     │  └─-l──x────┘  └─-p──┬─s──────────┬─┘  └─-cli──┬────────────────┬─┘
   └─-q───┼─on──┼─────┘                      ├─t──table───┤            └─cache-size─────┘
          └─del─┘                            └─d──────────┘

►──┬──────┬────────────────────────────────────────────────────────────────────────►◄
   └─-h───┘
```

**Command parameters:**

**-d dbname**

An alias name for the database against which SQL statements are to be applied. If this option is not specified, the value of the **DB2DBDFT** environment variable is used.

**-f file_name**

Name of an input file containing SQL statements. The default is standard input.

Identify comment text with two hyphens at the start of each line, that is, `--` *<comment>*. If it is to be included in the output, mark the comment as follows: `--#COMMENT` *<comment>*.

A *block* is a number of SQL statements that are treated as one, that is, information is collected for all of those statements at once, instead of one at a time. Identify the beginning of a block of queries as follows: `--#BGBLK`. Identify the end of a block of queries as follows: `--#EOBLK`.

Specify one or more control options as follows: `--#SET` *<control option> <value>*. Valid control options are:

**ROWS_FETCH**

Number of rows to be fetched from the answer set. Valid values are `-1` to *n*. The default value is `-1` (all rows are to be fetched).

**ROWS_OUT**

Number of fetched rows to be sent to output. Valid values are `-1` to *n*. The default value is `-1` (all fetched rows are to be sent to output).

**PERF_DETAIL**

Specifies the level of performance information to be returned. Valid values are:

**0** No timing is to be done.

**1** Return elapsed time only.

**2** Return elapsed time and CPU time.

**3** Return a summary of monitoring information.

**4** Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed).

**Note:** The snapshot will not include hash join information.

**5** Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed). Also return a snapshot for the buffer pools, table spaces and FCM (an FCM snapshot is only available in a multi-database-partition environment).

**Note:** The snapshot will not include hash join information.

The default value is 1. A value >1 is only valid on DB2 Version 2 and DB2 UDB servers, and is not currently supported on host machines.

**DELIMITER**
A one- or two-character end-of-statement delimiter. The default value is a semicolon (;).

**SLEEP**
Number of seconds to sleep. Valid values are `1` to *n*.

**PAUSE**
Prompts the user to continue.

**TIMESTAMP**
Generates a time stamp.

**-a userid/passwd**
Name and password used to connect to the database. The slash (`/`) must be included.

**-t delcol**
Specifies a single character column separator.

**Note:** To include a tab column delimiter use `-t TAB`.

**-r outfile**
An output file that will contain the query results. An optional *outfile2* will contain a results summary. The default is standard output.

**-c** Automatically commit changes resulting from each SQL statement.

**-i** An elapsed time interval (in seconds).

**short** The time taken to open the cursor, complete the fetch, and close the cursor.

**long** The elapsed time from the start of one query to the start of the next query, including pause and sleep times, and command overhead.

**complete**
The time to prepare, execute, and fetch, expressed separately.

**-o options**
Control options. Valid options are:

**f rows_fetch**
Number of rows to be fetched from the answer set. Valid values are `-1` to *n*. The default value is `-1` (all rows are to be fetched).

**r rows_out**
Number of fetched rows to be sent to output. Valid values are `-1` to *n*. The default value is `-1` (all fetched rows are to be sent to output).

**p perf_detail**
Specifies the level of performance information to be returned. Valid values are:

**0** No timing is to be done.

**1** Return elapsed time only.

**2** Return elapsed time and CPU time.

**3** Return a summary of monitoring information.

**4** Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed).

**5** Return a snapshot for the database manager, the database, the application, and the statement (the latter is returned only if autocommit is off, and single statements, not blocks of statements, are being processed). Also return a snapshot for the bufferpools, table spaces and FCM (an FCM snapshot is only available in a multi-database-partition environment).

The default value is 1. A value >1 is only valid on DB2 Version 2 and DB2 UDB servers, and is not currently supported on host machines.

**o query_optimization_class**
Sets the query optimization class.

**e explain_mode**
Sets the explain mode under which **db2batch** runs. The explain tables must be created prior to using this option. Valid values are:

**0** Run query only (default).

**1** Populate explain tables only. This option populates the explain tables and causes explain snapshots to be taken.

**2** Populate explain tables and run query. This option populates the explain tables and causes explain snapshots to be taken.

**-v** Verbose. Send information to standard error during query processing. The default value is `off`.

**-s** Summary Table. Provide a summary table for each query or block of queries, containing elapsed time (if selected), CPU times (if selected), the rows fetched, and the rows printed. The arithmetic and geometric means for elapsed time and CPU times are provided if they were collected.

**-q** Query output. Valid values are:

**on** Print only the *non-delimited* output of the query.

**off** Print the output of the query and all associated information. This is the default.

**del** Print only the *delimited* output of the query.

**-l x** Specifies the termination character.

**-p** Parallel (ESE only). Only SELECT statements are supported in this mode. Output names must have a fully qualified path. Valid values are:

**s** Single table or collocated join query. SELECT statements cannot contain only column functions. This is a requirement of the DBPARTITIONNUM function which is added to the query. If this option is specified, the DBPARTITIONNUM function will be added to the WHERE clause of the query, and a temporary table will not be created. This option is valid only if the query contains a single table in the FROM clause, or if the tables contained in the FROM clause are collocated.

If this option is specified and the query contains a GROUP BY clause, the columns specified in GROUP BY must be a superset of the tables partitioning key.

**t table**
Specifies the name of an existing table to use as the staging table to populate with the export data. If the query contains multiple tables in the FROM clause, and the tables are not collocated, the result set is inserted into the specified table and a SELECT is issued in parallel on all partitions to generate the files with the export data.

**d**
Creates a system table in IBMDEFAULTGROUP to be used for an INSERT INTO statement. If the query contains multiple tables in the FROM clause, and the tables are not collocated, the result set is inserted into the specified table and a SELECT is issued in parallel on all partitions to generate the files with the export data.

If a *local* output file is specified (using the **-r** option), the output from each database partition will go into a separate file with the same name on each database partition. If a file that is on an NFS-mounted file system is specified, all of the output will go into this file.

**-cli**
Run **db2batch** in CLI mode. The default is to use embedded dynamic SQL. The statement memory can be set manually, using the *cache-size* parameter.

**cache-size**
Size of the statement memory, expressed as number of statements. The default value is 25. If the utility encounters an SQL statement that has already been prepared, it will reuse the old plans. This parameter can only be set when **db2batch** is run in CLI mode.

**-h**
Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Usage notes:**

1. Although SQL statements can be up to 65 535 characters in length, no text line in the input file can exceed 3 898 characters, and long statements must be divided among several lines. Statements must be terminated by a delimiter (the default is a semicolon).
2. SQL statements are executed with the repeatable read (RR) isolation level.
3. SQL queries that include LOB columns in their output are not supported.

**Related reference:**

- "db2sql92 - SQL92 Compliant SQL Statement Processor" on page 207

## db2bfd - Bind File Description Tool

Displays the contents of a bind file. This utility, which can be used to examine and to verify the SQL statements within a bind file, as well as to display the precompile options used to create the bind file, might be helpful in problem determination related to an application's bind file.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
                ┌─,──────┐
                ▼        │
►►──db2bfd──────┬─-h─┬───┴──filespec──────────────────────────────────────►◄
                ├─-b─┤
                ├─-s─┤
                └─-v─┘
```

**Command parameters:**

**-h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-b** Display the bind file header.

**-s** Display the SQL statements.

**-v** Display the host variable declarations.

**filespec**
Name of the bind file whose contents are to be displayed.

# db2cap - CLI/ODBC Static Package Binding Tool

Binds a capture file to generate one or more static packages. A capture file is generated during a static profiling session of a CLI/ODBC/JDBC application, and contains SQL statements that were captured during the application run. This utility processes the capture file so that it can be used by the CLI/ODBC/JDBC driver to execute static SQL for the application.

**Authorization:**

- Access privileges to any database objects referenced by SQL statements recorded in the capture file.
- Sufficient authority to set bind options such as OWNER and QUALIFIER if they are different from the connect ID used to invoke the **db2cap** command.
- BINDADD authority if the package is being bound for the first time; otherwise, BIND authority is required.

**Command syntax:**

```
►►──db2cap──┬─────┬──bind──capture-file───-d──database_alias──────────────────►
            ├─-h─┤
            └─-?─┘

►─┬──────────────────────────────┬───────────────────────────────────────────►◄
  └─-u──userid──┬──────────────┬─┘
                └─-p──password─┘
```

**Command parameters:**

**-h/-?** Displays help text for the command syntax.

**bind** *capture-file*
: Binds the statements from the capture file and creates one or more packages.

**-d** *database_alias*
: Specifies the database alias for the database that will contain one or more packages.

**-u** *userid*
: Specifies the user ID to be used to connect to the data source.

**Note:** If a user ID is not specified, a trusted authorization ID is obtained from the system.

**-p** *password*
: Specifies the password to be used to connect to the data source.

**Usage notes:**

This command must be entered in lowercase on UNIX platforms, but can be entered in either lowercase or uppercase on Windows operating systems.

This utility supports many user-specified bind options that can be found in the capture file. In order to change the bind options, open the capture file in a text editor.

The SQLERROR(CONTINUE) and the VALIDATE(RUN) bind options can be used to create a package.

When using this utility to create a package, static profiling must be disabled.

The number of packages created depends on the isolation levels used for the SQL statements that are recorded in the capture file. The package name consists of up to a maximum of the first seven characters of the package keyword from the capture file, and one of the following single-character suffixes:

- 0 - Uncommitted Read (UR)
- 1 - Cursor Stability (CS)
- 2 - Read Stability (RS)
- 3 - Repeatable Read (RR)
- 4 - No Commit (NC)

To obtain specific information about packages, the user can:

- Query the appropriate SYSIBM catalog tables using the COLLECTION and PACKAGE keywords found in the capture file.
- View the capture file.

## db2cc - Start Control Center

Starts the Control Center. The Control Center is a graphical interface that is used to manage database objects (such as databases, tables, and packages) and their relationship to one another.

**Authorization:**

*sysadm*

**Command syntax:**

```
►►──db2cc──┬───────┬──┬────┬──┬──────────────┬──┬─────────┬──────────────────►
           ├─-wc───┤  └─-t─┘  └─-tf──filename─┘  └─-tcomms─┘
           ├─-rc───┤
           ├─-tc───┤
           ├─-j────┤
           ├─-hc───┤
           ├─-mv───┤
           ├─-tm───┤
           └─-icc──┘

►──┬─────────────────────────┬──┬───────────────────────┬──┬─────┬──┬───────────────┬──►
   │             ┌─+──────┐  │  └─-ccf──┬────────────┬──┘  └─-ic─┘  └─-ict──seconds─┘
   └─-tfilter────▼─filter─┴──┘          └─filename───┘

►──┬──────────────────────────────────────────────────────────────────────┬────────►◄
   └─-h──system──┬───────────────────────────────────────┬──┬────────────────────┬─┘
                 └─-i──instance──┬─────────────────┬─────┘  └─-sub──subsystem────┘
                                 └─-d──database────┘
```

**Command parameters:**

**-wc** Opens the Data Warehouse Center.

**-rc** Opens the Replication Center.

**-hc** Opens the Health Center.

**-tc** Opens the Task Center.

**-j** Opens the Journal.

**-mv** Opens the Memory Visualizer.

**-tm** Opens the Identify Indoubt Transaction Manager.

**-icc** Opens the Information Catalog Manager.

**-t** Turns on Control Center Trace for an initialization code. This option has no effect on Windows operating systems.

**-tf** Turns on Control Center Trace for an initialization code and saves the output of the trace to the specified file. The output file is saved to `<DB2 install path>\sqllib\tools` on Windows and to `/home/<userid>/sqllib/tools` on UNIX-based platforms.

**-tcomms**
Limits tracing to communications events.

**-tfilter filter**
Limits tracing to entries containing the specified filter or filters.

**-ccf filename**
Opens the Command Editor. If a filename is specified, then the contents of this file are loaded into the Command Editor's Script page. Note that when specifying a file name, you must provide the absolute path to the file.

**-ic** Opens the Information Center.

**-ict seconds**
Idle Connection Timer. Closes any idle connections in the pools maintained by the Control Center after the number of seconds specified. The default timer is 30 minutes.

**-h** *system*
Opens the Control Center in the context of a system.

**-i** *instance*
Opens the Control Center in the context of an instance.

**-d** *database*
Opens the Control Center in the context of a database.

**-sub** *subsystem*
Opens the Control Center in the context of a subsystem.

**Related reference:**

- "GET ADMIN CONFIGURATION" on page 374
- "RESET ADMIN CONFIGURATION" on page 635
- "UPDATE ADMIN CONFIGURATION" on page 715

## db2cfexp - Connectivity Configuration Export Tool

Exports connectivity configuration information to an export profile, which can later be imported at another DB2 Universal Database (UDB) workstation instance of similar instance type. The resulting profile will contain only configuration information associated with the current DB2 UDB instance. This profile can be referred to as a *client* configuration profile or *instance* configuration profile.

This utility exports connectivity configuration information into a file known as a configuration profile. It is a non-interactive utility that packages all of the configuration information needed to satisfy the requirements of the export options specified. Items that can be exported are:

- Database information (including DCS and ODBC information)
- Node information
- Protocol information
- database manager configuration settings
- UDB registry settings
- Common ODBC/CLI settings.

This utility is especially useful for exporting connectivity configuration information at workstations that do not have the DB2 Configuration Assistant installed, and in situations where multiple similar remote DB2 UDB clients are to be installed, configured, and maintained (for example, cloning or making templates of client configurations).

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Command syntax:**

```
                            ┌TEMPLATE┐
►►──db2cfexp──filename──────┼BACKUP──┼─────────────────────────────────────────►◄
                            └MAINTAIN┘
```

**Command parameters:**

**filename**

Specifies the fully qualified name of the target export file. This file is known as a configuration profile.

**TEMPLATE**

Creates a configuration profile that is used as a template for other instances of the same instance type. The profile includes information about:

- All databases, including related ODBC and DCS information
- All nodes associated with the exported databases
- Common ODBC/CLI settings
- Common client settings in the database manager configuration
- Common client settings in the DB2 UDB registry.

**BACKUP**

Creates a configuration profile of the DB2 UDB instance for local backup purposes. This profile contains all of the instance configuration information, including information of a specific nature relevant only to this local instance. The profile includes information about:

- All databases including related ODBC and DCS information
- All nodes associated with the exported databases
- Common ODBC/CLI settings
- All settings in the database manager configuration
- All settings in the DB2 UDB registry
- All protocol information.

**MAINTAIN**
Creates a configuration profile containing only database- and node-related information for maintaining or updating other instances.

## db2cfimp - Connectivity Configuration Import Tool

Imports connectivity configuration information from a file known as a configuration profile. It is a non-interactive utility that will attempt to import all the information found in the configuration profile.

A configuration profile can contain connectivity items such as:

- Database information (including DB2 Connect and ODBC information)
- Node information
- Protocol information
- database manager configuration settings
- Universal Database (UDB) registry settings
- Common ODBC/CLI settings.

This utility can be used to duplicate the connectivity information from another similar instance that was configured previously. It is especially useful on workstations that do not have the DB2 Configuration Assistant (CA) installed, and in situations where multiple similar remote UDB clients are to be installed, configured, and maintained (for example, cloning or making templates of client configurations). When cloning an instance, the profile imported should always be a client configuration profile that contains configuration information about one DB2 UDB instance only.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Command syntax:**

```
►►──db2cfimp──filename──────────────────────────────────────────────►◄
```

**Command parameters:**

**filename**
Specifies the fully qualified name of the configuration profile to be imported. Valid import configuration profiles are: profiles created by any DB2 UDB or DB2 Connect connectivity configuration export method, or server access profiles.

## db2cidmg - Remote Database Migration

Supports remote unattended migration in the Configuration, Installation, and Distribution (CID) architecture environment.

**Authorization:**

One of the following:

- *sysadm*
- *dbadm*

**Command syntax:**

```
►►──db2cidmg──┬─database────┬──┬───────────────┬──┬──────┬──────────────────────────►◄
              ├─/r=respfile─┤  └─/l1=logfile───┘  └─/b───┘
              └─/e──────────┘
```

**Command parameters:**

**database**
Specifies an alias name for the database which is to be migrated. If not specified, a response file or /e must be provided for program invocation. Note that the database alias must be cataloged on the target workstation. However, it can be a local or a remote database.

**/r**
Specifies a response file to be used for CID migration. The response file is an ASCII file containing a list of databases which are to be migrated. If not specified, a database alias or /e must be provided for program invocation.

**/e**
Indicates that every single database cataloged in the system database directory is to be migrated. If /e is not specified, a database alias or a response file must be provided.

**/l1**
Specifies the path name of the file to which error log information from remote workstations can be copied after the migration process is completed. If more than one database is specified in the response file, the log information for each database migration is appended to the end of the file. Regardless of whether /l1 is specified or not, a log file with the name DB2CIDMG.LOG is generated and kept in the workstation's file system where the database migration has been performed.

**/b**
Indicates that all packages in the database are to be rebound once migration is complete.

# db2ckbkp - Check Backup

This utility can be used to test the integrity of a backup image and to determine whether or not the image can be restored. It can also be used to display the metadata stored in the backup header.

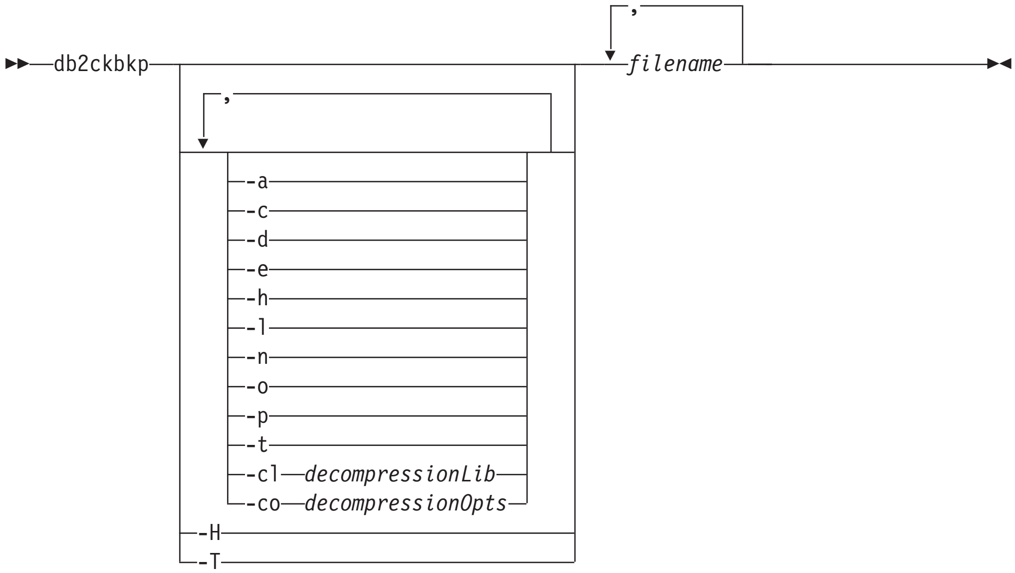
**Authorization:**

Anyone can access the utility, but users must have read permissions on image backups in order to execute this utility against them.

**Required connection:**

None

**Command syntax:**



**Command parameters:**

**-a** Displays all available information.

**-c** Displays results of checkbits and checksums.

**-cl** *decompressionLib*
: Indicates the name of the library to be used to perform the decompression. The name must be a fully qualified path referring to a file on the server. If this parameter is not specified, DB2 will attempt to use the library stored in the image. If the backup was not compressed, the value of this parameter will be ignored. If the specified library cannot be loaded, the operation will fail.

**-co** *decompressionOpts*
: Describes a block of binary data that will be passed to the initialization routine in the decompression library. DB2 will pass this string directly from the client to the server, so any issues of byte reversal or code page conversion will have to be handled by the decompression library. If the first character of the data block is '@', the remainder of the data will be interpreted by DB2 as the name of a file residing on the server. DB2 will

then replace the contents of *string* with the contents of this file and will pass this new value to the initialization routine instead. The maximum length for string is 1024 bytes.

**-d** Displays information from the headers of DMS table space data pages.

**-e** Extracts pages from an image to a file. To extract pages, you will need an input and an output file. The default input file is called extractPage.in. You can override the default input file name by setting the DB2LISTFILE environment variable to a full path. The format of the input file is as follows:

For SMS table spaces:

```
S <tbspID> <objID> <objType> <startPage> <numPages>
```

For DMS table spaces:

```
D <tbspID> <objType> <startPage> <numPages>
```

**Note:** <objType> is only needed if verifying DMS load copy images
For log files:

```
 L <log num> <startPos> <numPages>
```

For other data (for example, initial data):

```
 O <objType> <startPos> <numBytes>
```

The default output file is extractPage.out. You can override the default output file name by setting the DB2EXTRACTFILE environment variable to a full path.

**-h** Displays media header information including the name and path of the image expected by the restore utility.

**-H** Displays the same information as -h but only reads the 4K media header information from the beginning of the image. It does not validate the image.

**Note:** This option cannot be used in combination with any other options.

**-l** Displays log file header (LFH) and mirror log file header (MFH) data.

**-n** Prompt for tape mount. Assume one tape per device.

**-o** Displays detailed information from the object headers.

**-p** Displays the number of pages of each object type.

**-t** Displays table space details, including container information, for the table spaces in the image.

**-T** Displays the same information as -t but does not validate the image.

**Note:** This option cannot be used in combination with any other options.

**filename**
The name of the backup image file. One or more files can be checked at a time.

**Notes:**

1. If the complete backup consists of multiple objects, the validation will only succeed if **db2ckbkp** is used to validate all of the objects at the same time.

2. When checking multiple parts of an image, the first backup image object (.001) must be specified first.

**Examples:**

Example 1 (on UNIX platforms)

```
db2ckbkp SAMPLE.0.krodger.NODE0000.CATN0000.19990817150714.001
SAMPLE.0.krodger.NODE0000.CATN0000.19990817150714.002
SAMPLE.0.krodger.NODE0000.CATN0000.19990817150714.003

[1] Buffers processed:  ##
[2] Buffers processed:  ##
[3] Buffers processed:  ##
Image Verification Complete - successful.
```

Example 2 (on Windows platforms)

```
db2ckbkp SAMPLE.0\krodger\NODE0000\CATN0000\19990817\150714.001
SAMPLE.0\krodger\NODE0000\CATN0000\19990817\150714.002
SAMPLE.0\krodger\NODE0000\CATN0000\19990817\150714.003

[1] Buffers processed:  ##
[2] Buffers processed:  ##
[3] Buffers processed:  ##
Image Verification Complete - successful.
```

Example 3

```
db2ckbkp -h SAMPLE2.0.krodger.NODE0000.CATN0000.19990818122909.001

=====================
MEDIA HEADER REACHED:
=====================
        Server Database Name           -- SAMPLE2
        Server Database Alias          -- SAMPLE2
        Client Database Alias          -- SAMPLE2
        Timestamp                      -- 19990818122909
        Database Partition Number      -- 0
        Instance                       -- krodger
        Sequence Number                -- 1
        Release ID                     -- 900
        Database Seed                  -- 65E0B395
        DB Comment's Codepage (Volume) -- 0
        DB Comment (Volume)            --
        DB Comment's Codepage (System) -- 0
        DB Comment (System)            --
        Authentication Value           -- 255
        Backup Mode                    -- 0
        Include Logs                   -- 0
        Compression                    -- 0
        Backup Type                    -- 0
        Backup Gran.                   -- 0
        Status Flags                   -- 11
        System Cats inc                -- 1
        Catalog Database Partition No. -- 0
        DB Codeset                     -- ISO8859-1
        DB Territory                   --
        LogID                          -- 1074717952
        LogPath                       -- /home/krodger/krodger/NODE0000/
                                          SQL00001/SQLOGDIR
        Backup Buffer Size             -- 4194304
        Number of Sessions             -- 1
        Platform                       -- 0

The proper image file name would be:
SAMPLE2.0.krodger.NODE0000.CATN0000.19990818122909.001
```

```
[1] Buffers processed:  ####
Image Verification Complete - successful.
```

**Usage notes:**

1. If a backup image was created using multiple sessions, **db2ckbkp** can examine all of the files at the same time. Users are responsible for ensuring that the session with sequence number 001 is the first file specified.
2. This utility can also verify backup images that are stored on tape (except images that were created with a variable block size). This is done by preparing the tape as for a restore operation, and then invoking the utility, specifying the tape device name. For example, on UNIX based systems:

   ```
   db2ckbkp -h /dev/rmt0
   ```

   and on Windows:

   ```
   db2ckbkp -d \\.\tape1
   ```

3. If the image is on a tape device, specify the tape device path. You will be prompted to ensure it is mounted, unless option ′-n′ is given. If there are multiple tapes, the first tape must be mounted on the first device path given. (That is the tape with sequence 001 in the header).

   The default when a tape device is detected is to prompt the user to mount the tape. The user has the choice on the prompt. Here is the prompt and options: (where the device I specified is on device path /dev/rmt0)

   ```
    Please mount the source media on device /dev/rmt0.
    Continue(c), terminate only this device(d), or abort this tool(t)?
    (c/d/t)
   ```

   The user will be prompted for each device specified, and when the device reaches the end of tape.

**Related reference:**

- “db2adutl - Managing DB2 objects within TSM” on page 10

## db2ckmig - Database Pre-migration Tool

Verifies that a database can be migrated.

**Scope:**

This command only affects the database partition on which it is executed. In a partitioned database environment, run the command on each database partition.

**Authorization:**

*sysadm*

**Required connection:**

None

**Command syntax:**

```
►►──db2ckmig──┬─database─┬──-l──filename──┬──────────────┬──┬────────────────┬──────────►◄
              └─-e───────┘                 └─-u──userid───┘  └─-p──password───┘
```

**Command parameters:**

**database**
Specifies an alias name of a database to be scanned.

**-e** Specifies that all local cataloged databases are to be scanned.

**-l** Specifies a log file to keep a list of errors and warnings generated for the scanned database.

**-u** Specifies the user ID of the system administrator.

**-p** Specifies the password of the system administrator's user ID.

**Usage notes:**

On UNIX-based platforms, when an instance is migrated with **db2imigr**, **db2ckmig** is implicitly called as part of the migration. If you choose to run **db2ckmig** manually, it must be run for each instance after DB2 UDB is installed, but before the instance is migrated. It should be run from the install path. It is located in DB2DIR/`bin`, where DB2DIR represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems.

On Windows platforms, instances are migrated during installation, and the installation will prompt you to run **db2ckmig**. It is located on the DB2 UDB CD, in `db2/Windows/Utilities`.

To verify the state of a database:

1. Log on as the instance owner.
2. Issue the **db2ckmig** command.
3. Check the log file.

**Note:** The log file displays the errors that occur when the **db2ckmig** command is run. Check that the log is empty before continuing with the migration process.

**Related tasks:**

- "Verifying that your databases are ready for migration" in the *Quick Beginnings for DB2 Servers*

# db2ckrst - Check Incremental Restore Image Sequence

Queries the database history and generates a list of timestamps for the backup images that are required for an incremental restore. A simplified restore syntax for a manual incremental restore is also generated.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──db2ckrst──-d──database name──-t──timestamp──┬──────────────────────┬──────────►
                                                │    ┌─database───┐    │
                                                └─-r─┴─tablespace─┴────┘

►──┬──────────────────────────────┬──┬────┬───────────────────────────────────►◄
   │         ┌────────────────┐   │  ├─-h─┤
   │         ▼                │   │  ├─-u─┤
   └─-n──────tablespace name──┴───┘  └─-?─┘
```

**Command parameters:**

**-d** *database name*
: Specifies the alias name for the database that will be restored.

**-t** *timestamp*
: Specifies the timestamp for a backup image that will be incrementally restored.

**-r**
: Specifies the type of restore that will be executed. The default is database.

**Note:** If TABLESPACE is chosen and no table space names are given, the utility looks into the history entry of the specified image and uses the table space names listed to do the restore.

**-n** *tablespace name*
: Specifies the name of one or more table spaces that will be restored.

**Note:** If a database restore type is selected and a list of table space names is specified, the utility will continue as a table space restore using the table space names given.

**-h/-u/-?**
: Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Examples:**

```
db2ckrst -d mr -t 20001015193455 -r database
db2ckrst -d mr -t 20001015193455 -r tablespace
db2ckrst -d mr -t 20001015193455 -r tablespace -n tbsp1 tbsp2

> db2 backup db mr

Backup successful. The timestamp for this backup image is : 20001016001426
```

```
> db2 backup db mr incremental

Backup successful. The timestamp for this backup image is : 20001016001445

> db2ckrst -d mr -t 20001016001445

Suggested restore order of images using timestamp 20001016001445 for
database mr.
===================================================================
  db2 restore db mr incremental taken at 20001016001445
  db2 restore db mr incremental taken at 20001016001426
  db2 restore db mr incremental taken at 20001016001445
===================================================================

> db2ckrst -d mr -t 20001016001445 -r tablespace -n userspace1
Suggested restore order of images using timestamp 20001016001445 for
database mr.
===================================================================
  db2 restore db mr tablespace ( USERSPACE1 ) incremental taken at
  20001016001445
  db2 restore db mr tablespace ( USERSPACE1 ) incremental taken at
  20001016001426
  db2 restore db mr tablespace ( USERSPACE1 ) incremental taken at
  20001016001445
===================================================================
```

**Usage notes:**

The database history must exist in order for this utility to be used. If the database history does not exist, specify the HISTORY FILE option in the RESTORE command before using this utility.

If the FORCE option of the PRUNE HISTORY command is used, you can delete entries that are required for automatic incremental restoration of databases. Manual restores will still work correctly. Use of this command can also prevent the dbckrst utility from being able to correctly analyse the complete chain of required backup images. The default operation of the PRUNE HISTORY command prevents required entries from being deleted. It is recommended that you do not use the FORCE option of the PRUNE HISTORY command.

This utility should not be used as a replacement for keeping records of your backups.

**Related tasks:**

- "Restoring from incremental backup images" in the *Data Recovery and High Availability Guide and Reference*

**Related reference:**

- "RESTORE DATABASE" on page 647
- "PRUNE HISTORY/LOGFILE" on page 584

# db2cli - DB2 Interactive CLI

Launches the interactive Call Level Interface environment for design and prototyping in CLI. Located in the `sqllib/samples/cli/` subdirectory of the home directory of the database instance owner.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──db2cli──────────────────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Usage notes:**
DB2 Interactive CLI consists of a set of commands that can be used to design, prototype, and test CLI function calls. It is a programmers' testing tool provided for the convenience of those who want to use it, and IBM makes no guarantees about its performance. DB2 Interactive CLI is not intended for end users, and so does not have extensive error-checking capabilities.

Two types of commands are supported:

**CLI commands**
Commands that correspond to (and have the same name as) each of the function calls that is supported by IBM CLI

**Support commands**
Commands that do not have an equivalent CLI function.

Commands can be issued interactively, or from within a file. Similarly, command output can be displayed on the terminal, or written to a file. A useful feature of the CLI command driver is the ability to capture all commands that are entered during a session, and to write them to a file, thus creating a *command script* that can be rerun at a later time.

## db2cmd - Open DB2 Command Window

Opens the CLP-enabled DB2 window, and initializes the DB2 command line environment. Issuing this command is equivalent to clicking on the *DB2 Command Window* icon.

This command is only available on Windows.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──db2cmd──┬────┬──────────────────────────────────────────────────►◄
            ├─-c─┤
            ├─-w─┤
            ├─-i─┤
            └─-t─┘
```

**Command parameters:**

**-c** Execute the command, and then terminate. For example, ″db2cmd /c dir″ causes the ″dir″ command to be invoked in a command window, and then the command window closes.

**-w** Wait until the cmd.exe process ends. For example, ″db2cmd /c /w dir″ invokes the ″dir″ command, and db2cmd.exe does not end until the command window closes.

**-i** Run the command window, sharing the same console and inheriting file handles. For example, ″db2cmd /c /w /i db2 get dbm cfg > myoutput″ invokes cmd.exe to run the DB2 command and to wait for its completion. A new console is not assigned, and stdout is piped to file ″myoutput″.

**-t** Instead of using ″DB2 CLP″ as the title of the command window, inherit the title from the invoking window. This is useful if one wants, for example, to set up an icon with a different title that invokes ″db2cmd /t″.

**Note:** All switches must appear before any commands to be executed. For example: db2cmd /t db2.

**Usage notes:**

If DB21061E (″Command line environment not initialized.″) is returned when bringing up the CLP-enabled DB2 window, or running CLP commands on Windows 98, the operating system may be running out of environment space. Check the `config.sys` file for the SHELL environment setup parameter, and increase its value accordingly. For example:

```
SHELL=C:\COMMAND.COM C:\ /P /E:32768
```

## db2dart - Database Analysis and Reporting Tool

Examines databases for architectural correctness and reports any encountered errors.

**Authorization:**

*sysadm*

**Required connection:**

None. **db2dart** must be run with no users connected to the database.

**Command syntax:**

```
►►──db2dart──database-name──┬─────────────────────┬─────────────────►◄
                            └─action──┬─────────┬─┘
                                      └─options─┘
```

**Command parameters:**
**Inspection actions**

**/DB** Inspects the entire database. This is the default option.

**/T** Inspects a single table. Requires two input values: a table space ID, and the table object ID or the table name.

**/TSF** Inspects only table space files and containers.

**/TSC** Inspects a table space's constructs, but not its tables. Requires one input value: table space ID.

**/TS** Inspects a single table space and its tables. Requires one input value: table space ID.

**/ATSC** Inspects constructs of all table spaces, but not their tables.

**Data formatting actions**

**/DD** Dumps formatted table data. Requires five input values: either a table object ID or table name, table space ID, page number to start with, number of pages, and verbose choice.

**/DI** Dumps formatted index data. Requires five input values: either a table object ID or table name, table space ID, page number to start with, number of pages, and verbose choice.

**/DM** Dumps formatted block map data. Requires five input values: either a table object ID or table name, table space ID, page number to start with, number of pages, and verbose choice.

**/DP** Dumps pages in hex format. Requires three input values: DMS table space ID, page number to start with, and number of pages.

**/DTSF** Dumps formatted table space file information.

**/DEMP**
Dumps formatted EMP information for a DMS table. Requires two input values: table space ID and the table object ID or table name.

**/DDEL**
Dumps formatted table data in delimited ASCII format. Requires four

input values: either a table object ID or table name, table space ID, page number to start with, and number of pages.

**/DHWM**
Dumps high water mark information. Requires one input value: table space ID.

**/LHWM**
Suggests ways of lowering the high water mark. Requires two input values: table space ID and number of pages.

### Repair actions

**/ETS** Extends the table limit in a 4 KB table space (DMS only), if possible. Requires one input value: table space ID.

**/MI** Marks index as invalid. When specifying this parameter the database must be offline. Requires two input values: table space ID and table object ID

**/MT** Marks table with drop-pending state. When specifying this parameter the database must be offline. Requires three input values: table space ID, either table object ID or table name, and password.

**/IP** Initializes the data page of a table as empty. When specifying this parameter the database must be offline. Requires five input values: table name or table object ID, table space ID, page number to start with, number of pages, and password.

### Change state actions

**/CHST**
Change the state of a database. When specifying this parameter the database must be offline. Requires one input value: database backup pending state.

### Help

**/H** Displays help information.

### Input value options

**/OI** *object-id*
Specifies the object ID.

**/TN** *table-name*
Specifies the table name.

**/TSI** *tablespace-id*
Specifies the table space ID.

**/ROW** *sum*
Identifies whether long field descriptors, LOB descriptors, and control information should be checked. You can specify just one option or add the values to specify more than one option.

**1** Checks control information in rows.

**2** Checks long field and LOB descriptors.

**/PW** *password*
Password required to execute the **db2dart** action. Contact DB2 Service for a valid password.

**/RPT** *path*
Optional path for the report output file.

**/RPTN** *file-name*
Optional name for the report output file.

**/PS** *number*
Specifies the page number to start with.

**Note:** The page number must be suffixed with `p` for pool relative.

**/NP** *number*
Specifies the number of pages.

**/V** *option*
Specifies whether or not the verbose option should be implemented. Valid values are:

**Y** Specifies that the verbose option should be implemented.

**N** Specifies that the verbose option should not be implemented.

**/SCR** *option*
Specifies type of screen output, if any. Valid values are:

**Y** Normal screen output is produced.

**M** Minimized screen output is produced.

**N** No screen output is produced.

**/RPTF** *option*
Specifies type of report file output, if any. Valid values are:

**Y** Normal report file output is produced.

**E** Only error information is produced to the report file.

**N** No report file output is produced.

**/ERR** *option*
Specifies type of log to produce in `DART.INF`, if any. Valid values are:

**Y** Produces normal log in `DART.INF` file.

**N** Minimizes output to log `DART.INF` file.

**E** Minimizes `DART.INF` file and screen output. Only error information is sent to the report file.

**/WHAT DBBP** *option*
Specifies the database backup pending state. Valid values are:

**OFF** Off state.

**ON** On state.

**Usage notes:**

When invoking the **db2dart** command, you can specify only one action. An action can support a varying number of options.

If you do not specify all the required input values when you invoke the **db2dart** command, you will be prompted for the values. For the /DDEL and /IP actions, the options cannot be specified from the command line, and must be entered when prompted by **db2dart**.

The /ROW, /RPT, /RPTN, /SCR, /RPTF, /ERR, and /WHAT DBBP options can all be invoked in addition to the action. They are not required by any of the actions.

**Related reference:**

- "rah and db2_all command descriptions" in the *Administration Guide: Implementation*

## db2dclgn - Declaration Generator

Generates declarations for a specified database table, eliminating the need to look up those declarations in the documentation. The generated declarations can be modified as necessary. The supported host languages are C/C++, COBOL, JAVA, and FORTRAN.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
                                              ┌──────────────┐
                                              ▼              │
►►──db2dclgn──-d──database-name──-t──table-name──┬──────────┬┴──────────────►◄
                                                 └─option───┘
```

**Command parameters:**

**-d database-name**
Specifies the name of the database to which a connection is to be established.

**-t table-name**
Specifies the name of the table from which column information is to be retrieved to generate declarations.

**option** One or more of the following:

**-a action**
Specifies whether declarations are to be added or replaced. Valid values are `ADD` and `REPLACE`. The default value is `ADD`.

**-b lob-var-type**
Specifies the type of variable to be generated for a LOB column. Valid values are:

**LOB (default)**
For example, in C, SQL TYPE is CLOB(5K) x.

**LOCATOR**
For example, in C, SQL TYPE is CLOB_LOCATOR x.

**FILE** For example, in C, SQL TYPE is CLOB_FILE x.

**-c** Specifies whether the column name is to be used as a suffix in the field name when a prefix (**-n**) is specified. If no prefix is specified, this option is ignored. The default behavior is to not use the column name as a suffix, but instead to use the column number, which starts at 1.

**-i** Specifies whether indicator variables are to be generated. Since host structures are supported in C and COBOL, an indicator table of size equal to the number of columns is generated, whereas for JAVA and FORTRAN, individual indicator variables are generated for each column. The names of the indicator table and the variable are the same as the table name and the column name, respectively, prefixed by ″IND-″ (for COBOL) or ″ind_″ (for the other languages). The default behavior is to not generate indicator variables.

**-l language**
Specifies the host language in which the declarations are to be generated. Valid values are `C`, `COBOL`, `JAVA`, and `FORTRAN`. The default behavior is to generate C declarations, which are also valid for C++.

**-n name**
Specifies a prefix for each of the field names. A prefix must be specified if the **-c** option is used. If it is not specified, the column name is used as the field name.

**-o output-file**
Specifies the name of the output file for the declarations. The default behavior is to use the table name as the base file name, with an extension that reflects the generated host language:

```
.h for C
.cbl for COBOL
.java for JAVA
.f for FORTRAN (UNIX)
.for for FORTRAN (INTEL)
```

**-p password**
Specifies the password to be used to connect to the database. It must be specified if a user ID is specified. The default behavior is to provide no password when establishing a connection.

**-r remarks**
Specifies whether column remarks, if available, are to be used as comments in the declarations, to provide more detailed descriptions of the fields.

**-s structure-name**
Specifies the structure name that is to be generated to group all the fields in the declarations. The default behavior is to use the unqualified table name.

**-u userid**
Specifies the user ID to be used to connect to the database. It must be specified if a password is specified. The default behavior is to provide no user ID when establishing a connection.

**-v**
Specifies whether the status (for example, the connection status) of the utility is to be displayed. The default behavior is to display only error messages.

**-w DBCS-var-type**
Specifies whether sqldbchar or wchar_t is to be used for a GRAPHIC/VARGRAPHIC/DBCLOB column in C.

**-y DBCS-symbol**
Specifies whether G or N is to be used as the DBCS symbol in COBOL.

**-z encoding**
Specifies the encoding the coding convention in accordance to the

particular server. Encoding can be either UDB or OS390. If OS390 is specified, the generated file would look identical to a file generated by OS390.

# db2demigdbd - Demigrate Database Directory Files

Converts DB2 UDB Version 8.2 database directory files to Version 8.1 database directory file format.

In DB2 UDB Version 8.2. two new fields were added to the database entry structure, therefore the database directory file structure has changed. When migrating from Version 8.1 to Version 8.2 the database directory files will be migrated automatically. However, to demigrate the database directory files from Version 8.2 to Version 8.1, **db2demigdbd** is required to convert the current database directory files to the Version 8.1 format.

**Authorization:**

On UNIX-based platforms, one of the following:

- instance owner
- member of the primary group

On Windows operating systems:

- member of the DB2 Administrators group

**Required connection:**

None

**Command syntax:**

```
►►──db2demigdbd──path──┬─1─┬──────────────────────────────────►◄
                       └─2─┘
```

**Command parameters:**

*path* Specifies the path name of the system or local database directories.

If demigrating only the system database directory, the path is $HOME/sqllib on Unix-based systems and $DB2PATH\instance on Windows operating systems.

If demigrating only the local database directory, the path is `$DBPATH/NODExxxx` on Unix-based systems and `$HOME\$INSTANCE\NODExxxx` on Windows operating systems.

If demigrating both the system and local database directories, the path is $HOME/sqllib on Unix-based systems and $DB2PATH\instance on Windows operating systems.

**1** Specifies that the database directory files located in *path* will be demigrated. Use this option to demigrate a local database directory when its path does not exist in a database entry of the system database directory file.

**2** Specifies that the system and all local database directory files in the instance located in *path* will be demigrated.

**Examples:**

To demigrate system database directory files on AIX:

```
db2demigdbd $HOME/sqllib 1
```

To demigrate system database directory files on Windows platforms:

```
db2demigdbd  d:\sqllib\db2  1
```

where db2 is the current instance.

To demigrate the local system database directory files on AIX:

```
db2demigdbd  ~/user/NODE0000  1
```

To demigrate the local system database directory files on Windows platforms:

```
db2demigdbd  d:\db2\NODE0000  1
```

To demigrate the system and all local database directory files in the instance, on AIX:

```
db2demigdbd  $HOME/sqllib  2
```

To de-migrate the system and all local database directory files in the instance, on Windows platforms:

```
db2demigdbd  d:\sqllib\db2 2
```

**Usage notes:**

You can run the utility before or after you have fallen back from Version 8.2 to Version 8.1.

If you fall back from Version 8.2 to Version 8.1 and do not demigrate your database directory files with the db2demigdbd utility, you will receive SQL10004 when you try to access the database.

# db2diag - db2diag.log analysis tool

Filters and formats the `db2diag.log` file.

**Authorization:**

None.

**Required connection:**

None.

**Command syntax:**

```
                                       ┌───────────────┐
                                       ▼               │
►►──db2diag──┬──────────────────────┬───┬────────────┬─┴──────────────────►
             ├─-h──-optionList──────┤   └─filename───┘
             ├─?──-optionList───────┤
             └─-help──-optionList───┘

►──┬──────────────────────────────┬──┬─────────────────────┬───────────────►
   ├─-g──fieldPatternList─────────┤  └─-pid──processIDList─┘
   ├─-filter──fieldPatternList────┤
   ├─-gi──fieldPatternList────────┤
   ├─-gv──fieldPatternList────────┤
   ├─-giv──fieldPatternList───────┤
   └─-gvi──fieldPatternList───────┘

►──┬──────────────────────┬──┬─────────────────────┬──┬──────────────────────┬──►
   └─-tid──threadIDList───┘  └─┬─-n────┬─nodeList─┘  └─┬─-e─────┬─errorList─┘
                               └─-node─┘                └─-error─┘

►──┬─────────────────────────┬──┬────────────┬──┬──────────────┬──┬──────┬──►
   └─┬─-l─────┬─levelList────┘  └─┬─-c─────┬─┘  └─┬─-V───────┬─┘  └─-cbe─┘
     └─-level─┘                   └─-count─┘      └─-verbose─┘

►──┬────────────┬──┬────────┬──┬─────────┬──┬───────────────┬──────────────►
   ├─-v─────────┤  └─-exist─┘  └─-strict─┘  └─-rc──┬─rcList─┤
   └─-invert────┘                                  └─switch─┘

►──┬──────────────────────┬──┬──────────────────────┬──────────────────────►
   └─-fmt──formatString───┘  └─┬─-o──────┬─pathName─┘
                               └─-output─┘
```

```
►──┬───────────────────────────────────────────────────┬──────────────────►
   └─┬─-f───────┬──┬─startTime────────────────────┬─┘
     └─-follow──┘  ├─:──sleepInterval─────────────┤
                   └─startTime──:──sleepInterval──┘
```

```
►──┬──────────────────────────────────────────────────────┬───────────────►
   └─┬─-H────────┬──┬─historyPeriod─────────────────────┬─┘
     └─-history──┘  ├─:──historyBegin───────────────────┤
                    └─historyPeriod──:──historyBegin────┘
```

```
►──┬───────────────────────────────────────┬──┬──────────────────────────┬────►◄
   └─┬─-t─────┬──┬─startTime────────────┬─┘  └─┬─-A────────┬──┬─────────┬─┘
     └─-time──┘  ├─:──endTime───────────┤      └─-archive──┘  └─dirName─┘
                 └─startTime──:──endTime┘
```

**Command parameters:**

*filename*
: Specifies one or more space-separated path names of DB2 diagnostic logs to be processed. If the file name is omitted, the `db2diag.log` file from the current directory is processed. If the file is not found, a directory set by the DIAGPATH variable is searched.

**-h/-help/?**
: Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed. If a list of options, *optionList*, containing one or more comma separated command parameters is omitted, a list of all available options with short descriptions is displayed. For each option specified in the *optionList*, more detailed information and usage examples are displayed. Help output can be modified by using one of the following switches in place of the *optionList* argument to display more information about the tool and its usage:

**brief** Displays help information for all options without examples.

**examples**
Displays a few typical examples to assist in using the tool.

**tutorial**
Displays examples that describe advanced features.

**notes** Displays usage notes and restrictions.

**all** Displays complete information about all options, including usage examples for each option.

**-fmt** *formatString*
: Formats the **db2diag** output using a format string, *formatString*, containing record fields in the form %field, %{field}, @field or@{field}. The%{field} and @ {field} are used to separate a field name from the alphanumeric (or any other allowed character) that may follow the field name. All field names are case-insensitive. Field names can be shortened to the several first characters that are necessary to recognize a field name without ambiguity. In addition, aliases can be used for fields with long names. A prefix before

a field name, %, or @, specifies whether a text preceding the field will be displayed (%) or not (@) if the field is empty.

The following fields are currently available (the same list is valid for the fields with the prefix @:

**%timestamp/%ts**
Time stamp. This field can be divided into its constituent fields: %tsyear,%tsmonth, %tsday, %tshour, %tsmin (minute),%tssec (second),%tsmsec (microsecond for UNIX-based systems, millisecond for Windows operating systems).

**%timezone/%tz**
Number of minutes difference from UTC (Universal Coordinated Time). For example, -300 is Eastern Time.

**%recordid/%recid**
Unique record ID.

**%audience**
Intended audience for a logged message. 'E' indicates external users (IBM customers, service analysts, and developers). 'I' indicates internal users (service analysts and developers). 'D' indicates debugging information for developers.

**%level**
Severity level of a message: `Info`, `Warning`, `Error`, `Severe`, `Critical`, or `Event`.

**%source**
Location from which the logged error originated: `Origin`, `OS`, `Received`, or `Sent`.

**%instance/%inst**
Instance name.

**%node**
Database partition server number.

**%database/%db**
Database name.

**%pid** Process ID.

**%tid** Thread ID.

**%process**
Name associated with the process ID, in double quotation marks. For example, `"db2sysc.exe"`.

**%product**
Product name. For example, `DB2 UDB` or `DB2 COMMON`.

**%component**
Component name.

**%funcname**
Function name.

**%probe**
Probe number.

**%function**
Full function description: `%prod`, `%comp`, `%funcname`, `probe:%probe`.

**%appid**
Application ID.

**%coordnode**
Coordinator partition.

**%coordindex**
Coordinator index.

**%apphdl**
Application handle: `%coordnode - %coordindex`.

**%message/%msg**
Error message.

**%calledprod**
Product name of the function that returned an error.

**%calledcomp**
Component name of the function that returned an error.

**%calledfunc**
Name of the function that returned an error.

**%called**
Full description of the function that returned an error: `%calledprod`, `%calledcomp`, `%calledfunc`.

**%rcval**
Return code value (32 bytes).

**%rcdesc**
Error description.

**%retcode/%rc**
Return code returned by the function called: `%rcval %rcdesc`.

**%errno**
System error number.

**%errname**
System-specific error name.

**%oserror**
Operating system error returned by a system call: %errno %errname.

**%callstack**
Call stack.

**%datadesc**
Data description.

**%dataobject**
Data object.

**%data** Full data section of a message: %datadesc %dataobject.

**%argdesc**
Argument description.

**%argobject**
Argument object.

**%arg** Arguments of a function call that returned an error: %argdesc %argobject.

**%startevent**
Start event description.

**%stopevent**
Stop event description.

**%changeevent**
Change event description.

To always display the text preceding a field name (for example, for the required fields), the % field prefix should be used. To display the text preceding a field name when this field contains some data, the@ prefix should be used. Any combination of required and optional fields with the corresponding text descriptions is allowed.

The following special characters are recognized within a format string: `\n`, `\r`, `\f`, `\v`, and `\t`.

In contrast to other fields, the data and argument fields can contain several sections. To output a specific section, add the `[`*n*`]` after the field name where *n* is a section number (1≤**n**≤64). For example, to output the first data object and the second data description sections, use `%{dataobj}[1]` and `%{datadesc}[2]`. When `[`*n*`]` is not used, all sections logged are output using pre-formatted logged data exactly as appears in a log message, so there is no need to add the applicable text description and separating newline before each data field, argument field, or section.

**-g** *fieldPatternList*
*fieldPatternList* is a comma-separated list of field-pattern pairs in the following format: `fieldName operator searchPattern`.

The operator can be one of the following:

**=** Selects only those records that contain matches that form whole words. (Word search.)

**:=** Selects those records that contain matches in which a search pattern can be part of a larger expression.

**!=** Selects only non-matching lines. (Invert word match.)

**!:=** Selects only non-matching lines in which the search pattern can be part of a larger expression.

**^=** Selects records for which the field value starts with the search pattern specified.

**!^=** Selects records for which the field value does not start with the search pattern specified.

The same fields are available as described for the -fmt option, except that the% and @ prefixes are not used for this option

**-gi** *fieldPatternList*
Same as -g, but case-insensitive.

**-gv** *fieldPatternList*
Searches for messages that do not match the specified pattern.

**-gvi/-giv** *fieldPatternList*
Same as -gv, but case-insensitive.

**-pid** *processIDList*
Displays only log messages with the process IDs listed.

**-tid** *threadIDList*
Displays only log messages with the thread IDs listed.

**-n/-node** *nodeList*
Displays only log messages with the database partition numbers listed.

**-e/-error** *errorList*
Displays only log messages with the error numbers listed.

**-l/-level** *levelList*
Displays only log messages with the severity levels indicated.

**-c/-count**
Displays the number of records found.

**-v/-invert**
Inverts the pattern matching to select all records that do not match the specified pattern

**-strict**
Displays records using only one *field: value* pair per line. All empty fields are skipped.

**-V/-verbose**
Outputs all fields, including empty fields.

**-exist**
Defines how fields in a record are processed when a search is requested. If this option is specified, a field must exist in order to be processed.

**-cbe**
Common Base Event (CBE) Canonical Situation Data

**-o/-output** *pathName*
Saves the output to a file specified by a fully qualified *pathName*.

**-f/follow**
If the input file is a regular file, specifies that the tool will not terminate after the last record of the input file has been processed. Instead, it sleeps for a specified interval of time (*sleepInterval*), and then attemps to read and process further records from the input file as they become available.

This is option can be used when monitoring records being written to a file by another process. The *startTime* option, can be specified to show all the records logged after this time. The *startTime* option is specified using the following format: `YYYY-MM-DD-hh.mm.ss.nnnnnn`, where

*YYYY* Specifies a year.

*MM* Specifies a month of a year (01 through 12).

*DD* Specifies a day of a month (01 through 31).

*hh* Specifies an hour of a day (00 through 23).

*mm* Specifies a minute of an hour (00 through 59).

*ss* Specifies a second of a minute (00 through 59).

*nnnnnn*
Specifies microseconds on UNIX-based systems, or milliseconds on Windows operating systems.

Some or all of the fields that follow the year field can be omitted. If they are omitted, the default values will be used. The default values are 1 for the month and day, and 0 for all other fields.

If an exact match for the record time stamp does not exist in the diagnostic log file, the time closest to the time stamp specified will be used.

The *sleepInterval* option specifies a sleep interval in seconds. If a smaller time unit is required, it can be specified as a floating point value. The default value is 2 seconds

**-H/-history**

Displays the history of logged messages for the specified time interval. This option can be specified with the following options:

*historyPeriod*

Specifies that logged messages are displayed starting from the most recent logged record, for the duration specified by *historyPeriod*. The *historyPeriod* option is specified using the following format: `Number timeUnit`, where `Number` is the number of time units and `timeUnit` indicates the type of time unit: M (month), d (day), h (hour), m (minute), and s (second). The default value for `Number` is 30, and for *timeUnit* is `m`.

*historyPeriod*:*historyBegin*

Specifies that logged messages are displayed starting from the time specified by *historyBegin*, for the duration specified by *historyPeriod*.

The format is *YYYY-MM-DD-hh.mm.ss.nnnnnn*, where:

*YYYY* Specifies a year.

*MM* Specifies a month of a year (01 through 12).

*DD* Specifies a day of a month (01 through 31).

*hh* Specifies an hour of a day (00 through 23).

*mm* Specifies a minute of an hour (00 through 59).

*ss* Specifies a second of a minute (00 through 59).

*nnnnnn*

Specifies microseconds (UNIX-based platforms) or milliseconds (Windows operating systems).

**-t/-time**

Specifies a time stamp value. This option can be specified with one or both of the following options:

*startTime*

Displays all messages logged after *startTime*.

**:***endTime*

Displays all messages logged before *endTime*.

To display messages logged between *startTime* and *endTime*, specify -t *startTime*:*endTime*.

The format is *YYYY-MM-DD-hh.mm.ss.nnnnnn*, where:

*YYYY* Specifies a year.

*MM* Specifies a month of a year (01 through 12).

*DD* Specifies a day of a month (01 through 31).

*hh* Specifies an hour of a day (00 through 23).

*mm* Specifies a minute of an hour (00 through 59).

*ss* Specifies a second of a minute (00 through 59).

*nnnnnn*
Specifies microseconds (UNIX-based platforms) or milliseconds (Windows operating systems).

Some or all of the fields that follow the year field can be omitted. If they are omitted, the default values will be used. The default values are 1 for the month and day, and 0 for all other fields.

If an exact match for the record time stamp does not exist in the diagnostic log file, the time closest to the time stamp specified will be used.

**-A/-archive** *dirName*
Archives a diagnostic log file. When this option is specified, all other options are ignored. If one or more file names are specified, each file is processed individually. A timestamp, in the format `YYYY-MM-DD-hh.mm.ss, is appended to the file name.`

You can specify the name of the file and directory where it is to be archived. If the directory is not specified, the file is archived in the directory where the file is located and the directory name is extracted from the file name.

If you specify a directory but no file name, the current directory is searched for the db2diag.log file. If found, the file will be archived in the specified directory. If the file is not found, the directory specified by the DIAGPATH configuration parameter is searched for the db2diag.log file. If found, it is archived in the directory specified.

If you do not specify a file or a directory, the current directory is searched for the db2diag.log file. If found, it is archived in the current directory. If the file is not found, the directory specified by the DIAGPATH configuration parameter is searched for the db2diag.log file. If found, it is archived in the directory specified by the DIAGPATH configuration parameter.

**-rc** *rcList***/***switch*
Displays descriptions of DB2 internal error return codes for a space separated list, rcList, of the particular ZRC or ECF hexadecimal or negative decimal return codes. A full list of ZRC or ECF return codes can be displayed by specifying one of the following switches:

**zrc** Displays short descriptions of DB2 ZRC return codes.

**ecf** Displays short descriptions of DB2 ECF return codes.

**html** Displays short descriptions of DB2 ZRC return codes in the HTML format.

When this option is specified, all other options are ignored and output is directed to a display.

**Examples:**

To display all severe error messages produced by the process with the process ID (PID) 52356 and on node 1, 2 or 3, enter:

```
db2diag -g level=Severe,pid=952356 -n 1,2,3
```

To display all messages containing database SAMPLE and instance aabrashk, enter:

```
db2diag -g db=SAMPLE,instance=aabrashk
```

To display all severe error messages containing the database field, enter:

```
db2diag -g db:= -gi level=severe
```

To display all error messages containing the DB2 ZRC return code 0x87040055, and the application ID G916625D.NA8C.068149162729, enter:

```
db2diag -g msg:=0x87040055 -l Error | db2diag -gi appid^=G916625D.NA
```

To display all messages not containing the LOADID data, enter:

```
db2diag -gv data:=LOADID
```

To display only logged records not containing the LOCAL pattern in the application ID field, enter:

```
db2diag -gi appid!:=local  or  db2diag -g appid!:=LOCAL
```

All records that don’t match will be displayed. To output only messages that have the application ID field, enter:

```
db2diag -gvi appid:=local -exist
```

To display all messages logged after the one with timestamp 2003-03-03-12.16.26.230520 inclusively, enter:

```
db2diag -time 2003-03-03-12.16.26.230520
```

To display severe errors logged for the last three days, enter:

```
db2diag -gi "level=severe" -H 3d
```

To display all log messages not matching the pdLog pattern for the funcname field, enter:

```
db2diag -g 'funcname!=pdLog'  or  db2diag -gv 'funcn=pdLog'
```

To display all severe error messages containing component name starting from the ″base sys, enter:

```
db2diag -l severe | db2diag -g "comp^=base sys"
```

To view the growth of the db2diag.log file, enter: db2diag -f db2diag.log This displays all records written to the db2diag.log file in the current directory. Records are displayed as they added to the file The display continues until you press Ctrl-C.

To write the context of the db2diag.log into the db2diag_123.log file located in the /home/user/Logs directory, enter:

```
db2diag -o /home/user/Logs/db2diag_123.log
```

**Usage notes:**

1. Each option can appear only once. They can be specified in any order and can have optional parameters. Short options can not be included together. For example, use -l -eand not -le.
2. By default, **db2diag** looks for the db2diag.log file in the current directory. If the file is not found, the directory set by the DIAGPATH registry variable is searched next. If the db2diag.log file is not found, **db2diag** returns an error and exits.

3. Filtering and formatting options can be combined on a single command line to perform complex searches using pipes. The formatting options -fmt, -strict, -cbe, and -verbose should be used only after all filtering is done to ensure that only original logged messages with standard fields will be filtered, not those fields either defined or omitted by the user. Note that it is not necessary to use - when using pipes.
4. When pipes are used and one or more files names are specified on the command line, the **db2diag** input is processed differently depending on whether the - has been specified or not. If the - is omitted, input is taken from the specified files . In contrast, when the - option is specified, file names (even if present on the command line) are ignored and input from a terminal is used. When a pipe is used and a file name is not specified, the **db2diag** input is processed exactly the same way with or without the - specified on the command line.
5. The -exist option overwrites the default **db2diag** behavior for invert match searches when all records that do not match a pattern are output independent of whether they contain the proper fields or not. When the -exist option is specified, only the records containing fields requested are processed and output.
6. If the -fmt (format) option is not specified, all messages (filtered or not) are output exactly as they are written in the diagnostic log file. Output record format can be changed by using the -strict, -cbe, and -verbose options.
7. The -fmt option overwrites the -strict, -cbe and -verbose options.
8. Some restrictions apply when the -cbe option is specified and the `db2diag.log` file has been transferred over a network from the original computer. The **db2diag** tool collects information about DB2 and the computer host name locally, meaning that the DB2 version and the source or reporter componentID location field for the local system can be different from the corresponding values that were used on the original computer.
9. Ordinarily, the exit status is `0` if matches were found, and `1` if no matches were found. The exit status is `2` if there are syntax errors in the input data and patterns, the input files are inaccessible, or other errors are found.

# db2dlm_upd_hostname - Data Links Update Host Name

Registers changes in the host name of the DB2 or Data Links File Manager servers in a Data Links Manager environment. Host name changes must be registered if any of the DB2 or DLFM servers change, for example because of a change in the host name of the servers or because of an HACMP failover resulting in a change in the host names. If the host name of the DB2 server changes, running this utility on the DLFM server will update the DLFM_DB database with the new host name of the DB2 server. If the host name of the DLFM server changes, running this utility on the DB2 server will result in the update of the `datalink.cfg` file located in each of the database directories on the DB2 server.

On UNIX systems, **db2dlm_upd_hostname** is located in the INSTHOME/sqllib/bin directory, where INSTHOME is the home directory of the instance owner. On Windows systems, **db2dlm_upd_hostname** is located in the x:\sqllib\bin directory, where x: is the drive where you installed DB2 Data Links Manager.

**Authorization:**

When updating the DLFM server:

- *sysadm* of the DLFM_DB (user ID that created DLFM_DB)

When updating the DB2 server:

- *sysadm*

On a DLFM server, **db2dlm_upd_hostname** has to be run while attached to the DLFM instance. On a DB2 server, the utility has to be run while attached to the instance where the host databases reside.

**Command syntax:**

```
►►──db2dlm_upd_hostname──────────────────────────────────────────────────────────►

►──┬server──dlfm──-oldhost──name──-newhost──name─┬──────────────────────────────┬─┬──────────────────┬─┬──►◄
   │                                             └─-dbname──name──-dbinst──instance─┘ └─-skipbkup──yes─┘ │
   └server──db2──-oldhost──name──-newhost──name──┬────────────────────┬───────────────────────────────┘
                                                 └─-newport──port─────┘
```

**Command parameters:**
**For DLFM servers:**

**-server** *dlfm*
: Specifies that the utility is being run on the DLFM server.

**-oldhost** *name*
: Specifies the old host name of the DB2 server.

**-newhost** *name*
: specifies the new host name of the DB2 server.

**-dbname** *name*
: Specifies the database name entry in the dfm_dbid table of DLFM_DB database for which the host name needs to be changed. To be used only along with the -dbinst option. If this option along with -dbinst option is not provided, then all the matching host name entries in the dfm_dbid table will be updated.

**-dbinst** *instance*
: Specifies the database instance entry in the dfm_dbid table of DLFM_DB

database for which the host name needs to be changed. This option can only be specified with the -dbname option.

**-skipbkup** *yes*
: Specifies that DLFM_DB will not be backed up. If this option is not specified, the DLFM_DB database is backed up. This option is not recommended.

**For DB2 servers:**

**-server** *db2*
: Specifies that the utility is being run on the DB2 server.

**-oldhost** *name*
: Specifies the old host name of the DLFM server.

**-newhost** *name*
: Specifies the new host name of the DLFM server.

**-newport** *port*
: Specifies the port number to be used for communication between DB2 and DLFM.

**Examples:**

**On a DB2 server:**

```
db2dlm_upd_hostname -server dlfm -oldhost dlfscl.in.ibm.com
   -newhost  dlfstst.in.ibm.com
```

```
db2dlm_upd_hostname -server dlfm -oldhost dlfssrv.in.ibm.com
   -newhost dlfstst.in.ibm.com -dbname tstdb -dbinst regress
```

```
db2dlm_upd_hostname -server dlfm -oldhost dlfscl.in.ibm.com
   -newhost  dlfstst.in.ibm.com -skipbkup yes
```

**On a DLFM server:**

```
db2dlm_upd_hostname-server db2 -oldhost dlfsv.in.ibm.com
   -newhost dlfscln.in.ibm.com
```

```
db2dlm_upd_hostname-server db2 -oldhost dlfsv.in.ibm.com
   -newhost dlfscln.in.ibm.com -newport 5000
```

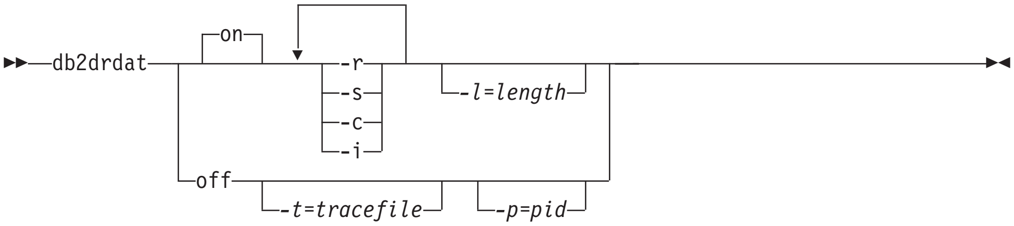
# db2drdat - DRDA Trace

Allows the user to capture the DRDA data stream exchanged between a DRDA Application Requestor (AR) and the DB2 UDB DRDA Application Server (AS). Although this tool is most often used for problem determination, by determining how many sends and receives are required to execute an application, it can also be used for performance tuning in a client/server environment.

**Authorization:**

None

**Command syntax:**



**Command parameters:**

**on** Turns on AS trace events (all if none specified).

**off** Turns off AS trace events.

**-r** Traces DRDA requests received from the DRDA AR.

**-s** Traces DRDA replies sent to the DRDA AR.

**-c** Traces the SQLCA received from the DRDA server on the host system. This is a formatted, easy-to-read version of *not null* SQLCAs.

**-i** Includes time stamps in the trace information.

**-l** Specifies the size of the buffer used to store the trace information.

**-p** Traces events only for this process. If `-p` is not specified, all agents with incoming DRDA connections on the server are traced.

**Note:** The *pid* to be traced can be found in the *agent* field returned by the LIST APPLICATIONS command.

**-t** Specifies the destination for the trace. If a file name is specified without a complete path, missing information is taken from the current path.

**Note:** If *tracefile* is not specified, messages are directed to `db2drdat.dmp` in the current directory.

**Usage notes:**

Do not issue **db2trc** commands while **db2drdat** is active.

**db2drdat** writes the following information to *tracefile*:

1. -r
   - Type of DRDA request
   - Receive buffer.

2. -s
   - Type of DRDA reply/object
   - Send buffer.
3. CPI-C error information
   - Severity
   - Protocol used
   - API used
   - Local LU name
   - Failed CPI-C function
   - CPI-C return code.

The command returns an exit code. A zero value indicates that the command completed successfully, and a nonzero value indicates that the command was not successful.

**Note:** If **db2drdat** sends the output to a file that already exists, the old file will be erased unless the permissions on the file do not allow it to be erased, in which case the operating system will return an error.

**Related reference:**

- "LIST APPLICATIONS" on page 480

# db2drvmp - DB2 Database Drive Map

Maps a database drive for Microsoft Cluster Server (MSCS). This command is available only on Windows platforms.

**Authorization:**

Read/write access to the Windows registry and the cluster registry.

**Required connection:**

Instance. The application creates a default instance attachment if one is not present.

**Command syntax:**

```
►►──db2drvmp───┬─add───────┬──dbpartition_number──from_drive──to_drive─────────────►◄
               ├─drop──────┤
               ├─query─────┤
               └─reconcile─┘
```

**Command parameters:**

**add** Assigns a new database drive map.

**drop** Removes an existing database drive map.

**query** Queries a database map.

**reconcile**
Reapplies the database drive mapping to the registry when the registry contents are damaged or dropped accidentally.

*dbpartition_number*
The database partition number. This parameter is required for add and drop operations. If this parameter is not specified for a reconcile operation, **db2drvmp** reconciles the mapping for all database partitions.

*from_drive*
The drive letter from which to map. This parameter is required for add and drop operations. If this parameter is not specified for a reconcile operation, **db2drvmp** reconciles the mapping for all drives.

*to_drive*
The drive letter to which to map. This parameter is required for add operations. It is not applicable to other operations.

**Examples:**

To set up database drive mapping from F: to E: for NODE0, issue the following command:

```
db2drvmp add 0 F E
```

To set up database drive mapping from E: to F: for NODE1, issue the following command:

```
db2drvmp add 1 E F
```

**Usage notes:**

1. Database drive mapping does not apply to table spaces, containers, or any other database storage objects.

2. Any setup of or change to the database drive mapping does not take effect immediately. To activate the database drive mapping, use the Microsoft Cluster Administrator tool to bring the DB2 resource offline, then online.
3. Using the TARGET_DRVMAP_DISK keyword in the `DB2MSCS.CFG` file will enable drive mapping to be done automatically.

# db2empfa - Enable Multipage File Allocation

Enables the use of multipage file allocation for a database. With multipage file allocation enabled for SMS table spaces, disk space is allocated one extent at a time rather than one page at a time.

**Scope:**

This command only affects the database partition on which it is executed.

**Authorization:**

*sysadm*

**Required connection:**

None. This command establishes a database connection.

**Command syntax:**

```
►►──db2empfa──database-alias──────────────────────────────────────────►◄
```

**Command parameters:**

**database-alias**
Specifies the alias of the database for which multipage file allocation is to be enabled.

**Usage notes:**

This utility:
- Connects to the database partition (where applicable) in exclusive mode
- In all SMS table spaces, allocates empty pages to fill up the last extent in all data and index files which are larger than one extent
- Changes the value of the database configuration parameter *multipage_alloc* to YES
- Disconnects.

Since **db2empfa** connects to the database partition in exclusive mode, it cannot be run concurrently on the catalog database partition, or on any other database partition.

Multipage file allocation can be enabled using **db2empfa** for databases that are created after the registry variable DB2_NO_MPFA_FOR_NEW_DB has been set.

**Related concepts:**
- “SMS table spaces” in the *Administration Guide: Performance*

## db2eva - Event Analyzer

Starts the event analyzer, allowing the user to trace performance data produced by DB2 event monitors that have their data directed to tables.

**Authorization:**

The Event Analyzer reads data from event monitor tables stored with the database. For this reason, you must have the following authorization to access this data:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

Database connection

**Command syntax:**

```
►►──db2eva──┬─────────────────────────┬──┬───────────────────────┬──────────────────────────►◄
            └─-db──-database-alias─┘  └─-evm──evmon-name─┘
```

**Command parameters:**

**Note:** The *db2eva* parameters are optional. If you do not specify parameters, the Open Event Analyzer dialog box appears to prompt you for the database and event monitor name.

**-db** *database-alias*
Specifies the name of the database defined for the event monitor.

**-evm** *evmon-name*
Specifies the name of the event monitor whose traces are to be analyzed.

**Usage notes:**

Without the required access, the user cannot retrieve any event monitor data.

There are two methods for retrieving event monitor traces:

1. The user can enter `db2eva` from the command line and the Open Event Analyzer Dialog box opens to let the user choose the database and event monitor names from the drop-down lists before clicking OK to open the Event Analyzer dialog box.
2. The user can specify the `-db` and `-evm` parameters from the command line and the Event Analyzer dialog opens on the specified database.

The Event Analyzer connects to the database, and issues a select target from SYSIBM.SYSEVENTTABLES to get the event monitor tables. The connection is then released after the required data has been retrieved.

**Note:** The event analyzer can be used to analyze the data produced by an active event monitor. However, event monitor captured after the event analyzer has been invoked might not be shown. Turn off the event monitor before invoking the Event Analyzer to ensure data are properly displayed.

## db2evmon - Event Monitor Productivity Tool

Formats event monitor file and named pipe output, and writes it to standard output.

**Authorization:**

None, unless connecting to the database (`-db -evm`); then, one of the following is required:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

None

**Command syntax:**

```
►►──db2evmon──┬─────────────────────────────────────────────┬───────────────►◄
              ├─-db──database-alias──-evm──event-monitor-name─┤
              └─-path──event-monitor-target─────────────────┘
```

**Command parameters:**

**-db database-alias**
: Specifies the database whose data is to be displayed. This parameter is case sensitive.

**-evm event-monitor-name**
: The one-part name of the event monitor. An ordinary or delimited SQL identifier. This parameter is case sensitive.

**-path event-monitor-target**
: Specifies the directory containing the event monitor trace files.

**Usage notes:**

If the instance is not already started when **db2evmon** is issued with the -db and -evm options, the command will start the instance.

If the instance is not already started when **db2evmon** is issued with the -path option, the command will not start the instance.

If the data is being written to files, the tool formats the files for display using standard output. In this case, the monitor is turned on first, and any event data in the files is displayed by the tool. To view any data written to files after the tool has been run, reissue **db2evmon**.

If the data is being written to a pipe, the tool formats the output for display using standard output as events occur. In this case, the tool is started *before* the monitor is turned on.

## db2evtbl - Generate Event Monitor Target Table Definitions

Generates sample CREATE EVENT MONITOR SQL statements that can be used when defining event monitors that write to SQL tables.

**Authorization:**

None.

**Required connection:**

None.

**Command syntax:**

```
>>--db2evtbl--+-----------------+--+--------------+---evm--evmName------------>
              '--schema--name---'  '--partitioned-'

     .-,----------.
     V            |
>-----event type--+-----------------------------------------------------------><
```

**Command parameters:**

**-schema**
Schema name. If not specified, the table names are unqualified.

**-partitioned**
If specified, elements that are only applicable for a partitioned environment are also generated.

**-evm** The name of the event monitor.

**event type**
Any of the event types available on the CREATE EVENT MONITOR statement, for example, DATABASE, TABLES, TRANSACTIONS.

**Examples:**

```
db2evtbl -schema smith -evm foo database, tables, tablespaces, bufferpools
```

**Usage notes:**

Output is written to standard output.

Defining WRITE TO TABLE event monitors is more straightforward when using the **db2evtbl** tool. For example, the following steps can be followed to define and activate an event monitor.

1. Use **db2evtbl** to generate the CREATE EVENT MONITOR statement.
2. Edit the SQL statement, removing any unwanted columns.
3. Use the CLP to process the SQL statement. (When the CREATE EVENT MONITOR statement is executing, target tables are created.)
4. Issue SET EVENT MONITOR STATE to activate the new event monitor.

Since all events other than deadlock event monitors can be flushed, creating more than one record per event, users who do not use the FLUSH EVENT MONITOR statement can leave the element evmon_flushes out of any target tables.

**Related concepts:**

- "Event monitors" in the *System Monitor Guide and Reference*

**Related reference:**

- "CREATE EVENT MONITOR statement" in the *SQL Reference, Volume 2*
- "SET EVENT MONITOR STATE statement" in the *SQL Reference, Volume 2*

## db2exfmt - Explain Table Format

You use the db2exfmt tool to format the contents of the explain tables. This tool is located in the `misc` subdirectory of the instance `sqllib` directory.

To use the tool, you require read access to the explain tables being formatted.

**Command syntax:**

```
>>──db2exfmt──┬───────────┬──┬───────────┬──┬────────┬──┬───────────────────┬──────>
              └─-d──dbname┘  └─-e──schema┘  └─-f──O──┘  │       ┌─────────┐ │
                                                        └─-g────▼─────────┴─┘
                                                               ├─O─┤
                                                               ├─T─┤
                                                               ├─I─┤
                                                               └─C─┘

>──┬────┬──┬─────────┬──┬───────────┬──┬─-o──outfile─┬──────────────────────────────>
   └─-l─┘  └─-n──name┘  └─-s──schema┘  │ ┌─-t─┐      │
                                       └─┴────┴──────┘

>──┬──────────────────────┬──┬──────────────┬──┬────────────┬──┬────┬──────────────><
   └─-u──userID──password─┘  └─-w──timestamp┘  └─-#──sectnbr─┘  └─-h─┘
```

**Command parameters:**

**-d** *dbname*
: Name of the database containing packages.

**-e** *schema*
: Explain table schema.

**-f**
: Formatting flags. In this release, the only supported value is `O` (operator summary).

**-g**
: Graph plan. If only `-g` is specified, a graph, followed by formatted information for all of the tables, is generated. Otherwise, any combination of the following valid values can be specified:

  **O** Generate a graph only. Do not format the table contents.
  **T** Include total cost under each operator in the graph.
  **I** Include I/O cost under each operator in the graph.
  **C** Include the expected output cardinality (number of tuples) of each operator in the graph.

**-l**
: Respect case when processing package names.

**-n** *name*
: Name of the source of the explain request (SOURCE_NAME).

**-s** *schema*
: Schema or qualifier of the source of the explain request (SOURCE_SCHEMA).

**-o** *outfile*
: Output file name.

**-t**
: Direct the output to the terminal.

**-u** *userID password*
: When connecting to a database, use the provided user ID and password.

  Both the user ID and password must be valid according to naming conventions and be recognized by the database.

**-w** *timestamp*
: Explain time stamp. Specify `-1` to obtain the latest explain request.

**-#** *sectnbr*
: Section number in the source. To request all sections, specify zero.

**-h**
: Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Usage notes:**

You will be prompted for any parameter values that are not supplied, or that are incompletely specified, except in the case of the `-h` and the `-l` options.

If an explain table schema is not provided, the value of the environment variable **USER** is used as the default. If this variable is not found, the user is prompted for an explain table schema.

Source name, source schema, and explain time stamp can be supplied in LIKE predicate form, which allows the percent sign (%) and the underscore (_) to be used as pattern matching characters to select multiple sources with one invocation. For the latest explained statement, the explain time can be specified as `-1`.

If `-o` is specified without a file name, and `-t` is not specified, the user is prompted for a file name (the default name is `db2exfmt.out`). If neither `-o` nor `-t` is specified, the user is prompted for a file name (the default option is terminal output). If `-o` and `-t` are both specified, the output is directed to the terminal.

**Related concepts:**

- "Explain tools" in the *Administration Guide: Performance*
- "Guidelines for using explain information" in the *Administration Guide: Performance*
- "Guidelines for capturing explain information" in the *Administration Guide: Performance*

## db2expln - SQL Explain

**Command syntax:**

```
►►──db2expln──┬──────────────────────┬──┬──────────────────┬──────────────────────►
              └─┤ connection-options ├─┘  └─┤ output-options ├─┘

►──┬───────────────────┬──┬───────────────────┬──┬───────────────────┬────────────►
   └─┤ package-options ├─┘  └─┤ dynamic-options ├─┘  └─┤ explain-options ├─┘

►──┬───────┬─────────────────────────────────────────────────────────────────────►◄
   └─-help─┘
```

**connection-options:**

```
├──-database──database-name──┬────────────────────────────┬────────────────────┤
                             └─-user──user-id──password─┘
```

**output-options:**

```
├──┬───────────────────────┬──┬────────────┬──────────────────────────────────┤
   └─ -output──output-file─┘  └─ -terminal─┘
```

**package-options:**

```
├── -schema──schema-name──-package──package-name──────────────────────────────►

►──┬─────────────────────────────────┬──┬──────────────────────────────┬──────►
   └─-version──version-identifier─┘  └─-escape──escape-character─┘

►──┬──────────┬──┬──────────────────────────┬─────────────────────────────────┤
   └─-noupper─┘  └─-section──section-number─┘
```

**dynamic-options:**

```
├──┬─────────────────────────────┬──┬─────────────────────────────────┬───────►
   └─-statement──sql-statement─┘  └─-stmtfile──sql-statement-file─┘

►──┬────────────────────────────────────────┬──┬────────┬─────────────────────┤
   └─-terminator──termination-character─┘  └─-noenv─┘
```

**explain-options:**

```
├──┬────────┬──┬────────┬─────────────────────────────────────────────────────┤
   └─-graph─┘  └─-opids─┘
```

**Command parameters:**
The options can be specified in any order.

**connection-options:**

These options specify the database to connect to and any options necessary to make the connection. The connection options are required except when the **-help** option is specified.

**-database** *database-name*
The name of the database that contains the packages to be explained.

For backward compatibility, you can use **-d** instead of **-database**.

**-user** *user-id password*
The authorization ID and password to use when establishing the database connection. Both *user-id* and *password* must be valid according to DB2® naming conventions and must be recognized by the database.

For backward compatibility, you can use **-u** instead of **-user**.

**output-options:**

These options specify where the db2expln output should be directed. Except when the **-help** option is specified, you must specify at least one output option. If you specify both options, output is sent to a file as well as to the terminal.

**-output** *output-file*
The output of db2expln is written to the file that you specify.

For backward compatibility, you can use **-o** instead of **-output**.

**-terminal**
The db2expln output is directed to the terminal.

For backward compatibility, you can use **-t** instead of **-terminal**.

**package-options:**

These options specify one or more packages and sections to be explained. Only static SQL in the packages and sections is explained.

**Note:** As in a LIKE predicate, you can use the pattern matching characters, which are percent sign (%) and underscore (_), to specify the *schema-name*, *package-name*, and *version-identifier*.

**-schema** *schema-name*
The schema of the package or packages to be explained.

For backward compatibility, you can use **-c** instead of **-schema**.

**-package** *package-name*
The name of the package or packages to be explained.

For backward compatibility, you can use **-p** instead of **-package**.

**-version** *version-identifier*
The version identifier of the package or packages to be explained. The default version is the empty string.

**-escape** *escape-character*
The character, *escape-character* to be used as the escape character for pattern matching in the *schema-name*, *package-name*, and *version-identifier*.

For example, the db2expln command to explain the package TESTID.CALC% is as follows:

```
db2expln -schema TESTID -package CALC% ....
```

However, this command would also explain any other plans that start with CALC. To explain only the TESTID.CALC% package, you must use an escape character. If you specify the exclamation point (!) as the escape character, you can change the command to read: `db2expln -schema TESTID -escape ! -package CALC!% ...` . Then the ! character is used as an escape character and thus !% is interpreted as the % character and not as the ″match anything″ pattern. There is no default escape character.

For backward compatibility, you can use **-e** instead of **-escape**.

**Note:** To avoid problems, do not specify the operating system escape character as the `db2expln` escape character.

**-noupper**
Specifies that the *schema-name*, *package-name*, and *version-identifier*, should not be converted to upper case before searching for matching packages.

By default, these variables are converted to upper case before searching for packages. This option indicates that these values should be used exactly as typed.

For backward compatibility, you can use **-l**, which is a lowercase L and not the number 1, instead of **-noupper**.

**-section** *section-number*
The section number to explain within the selected package or packages.

To explain all the sections in each package, use the number zero (0). This is the default behavior. If you do not specify this option, or if *schema-name*, *package-name*, or *version-identifier* contain a pattern-matching character, all sections are displayed.

To find section numbers, query the system catalog view SYSCAT.STATEMENTS. Refer to the *SQL Reference* for a description of the system catalog views.

For backward compatibility, you can use **-s** instead of **-section**.

**dynamic-options:**

These options specify one or more dynamic SQL statements to be explained.

**-statement** *sql-statement*
An SQL statement to be dynamically prepared and explained. To explain more than one statement, either use the **-stmtfile** option to provide a file containing the SQL statements to explain, or use the **-terminator** option to define a termination character that can be used to separate statements in the **-statement** option.

For compatibility with `dynexpln`, you can use **-q** instead of **-statement**.

**-stmtfile** *sql-statement-file*
A file that contains one or more SQL statements to be dynamically prepared and explained. By default, each line of the file is assumed to be a distinct SQL statement. If statements must span lines, use the **-terminator** option to specify the character that marks the end of an SQL statement.

For compatibility with `dynexpln`, you can use **-f** instead of **-stmtfile**.

**-terminator** *termination-character*
The character that indicates the end of dynamic SQL statements. By default, the **-statement** option provides a single SQL statement and each

line of the file in the **-stmtfile** is treated as a separate SQL statement. The termination character that you specify can be used to provide multiple SQL statements with **-statement** or to have statements span lines in the **-stmtfile** file.

For compatibility with dynexpln, you can use **-z** instead of **-terminator**.

**-noenv**
Specifies that dynamic statements that alter the compilation environment should not be executed after they have been explained.

By default, db2expln will execute any of the following statements after they have been explained:

```
SET CURRENT DEFAULT TRANSFORM GROUP
SET CURRENT DEGREE
SET CURRENT MAINTAINED TABLE TYPES FOR OPTIMIZATION
SET CURRENT QUERY OPTIMIZATION
SET CURRENT REFRESH AGE
SET PATH
SET SCHEMA
```

These statements make it possible to alter the plan chosen for subsequent dynamic SQL statements processed by db2expln.

If you specify **-noenv**, then these statement are explained, but not executed.

It is necessary to specify either **-statement** or **-stmtfile** to explain dynamic SQL. Both options can be specified in a single invocation of db2expln.

**explain-options:**

These options determine what additional information is provided in the explained plans.

**-graph** Show optimizer plan graphs. Each section is examined, and the original optimizer plan graph is constructed as presented by Visual Explain. Note that the generated graph might not match the Visual Explain graph exactly.

For backward compatibility, you can specify **-g** instead of **-graph**.

**-opids** Display operator ID numbers in the explained plan.

The operator ID numbers allow the output from db2expln to be matched to the output from the explain facility. Note that not all operators have an ID number and that some ID numbers that appear in the explain facility output do not appear in the db2expln output.

For backward compatibility, you can specify **-i** instead of **-opids**.

**-help** Shows the help text for db2expln. If this option is specified no packages are explained.

Most of the command line is processed in the *db2exsrv* stored procedure. To get help on all the available options, it is necessary to provide **connection-options** along with **-help**. For example, use:

```
db2expln -help -database SAMPLE
```

For backward compatibility, you can specify **-h** or **-?**.

**Usage notes:**

Unless you specify the **-help** option, you must specify either package-options or dynamic-options. You can explain both packages and dynamic SQL with a single invocation of db2expln.

Some of the option flags above might have special meaning to your operating system and, as a result, might not be interpreted correctly in the `db2expln` command line. However, you might be able to enter these characters by preceding them with an operating system escape character. For more information, see your operating system documentation. Make sure that you do not inadvertently specify the operating system escape character as the `db2expln` escape character.

Help and initial status messages, produced by `db2expln`, are written to standard output. All prompts and other status messages produced by the explain tool are written to standard error. Explain text is written to standard output or to a file depending on the output option chosen.

**Examples:**

To explain multiple plans with one invocation of `db2expln`, use the **-package**, **-schema**, and **-version** option and specify string constants for packages and creators with LIKE patterns. That is, the underscore (_) can be used to represent a single character, and the percent sign (%) can be used to represent the occurrence of zero or more characters.

To explain all sections for all packages in a database named SAMPLE, with the results being written to the file **my.exp** , enter

```
db2expln -database SAMPLE -schema % -package %  -output my.exp
```

As another example, suppose a user has a CLP script file called ″statements.db2″ and wants to explain the statements in the file. The file contains the following statements:

```
SET PATH=SYSIBM, SYSFUN, DEPT01, DEPT93@
SELECT EMPNO, TITLE(JOBID) FROM EMPLOYEE@
```

To explain these statements, enter the following command:

```
db2expln -database DEPTDATA -stmtfile statements.db2 -terminator @ -terminal
```

**Related concepts:**

- “SQL explain tools” in the *Administration Guide: Performance*
- “Description of db2expln and dynexpln output” in the *Administration Guide: Performance*
- “Examples of db2expln and dynexpln output” in the *Administration Guide: Performance*

# db2flsn - Find Log Sequence Number

Returns the name of the file that contains the log record identified by a specified log sequence number (LSN).

**Authorization:**

None

**Command syntax:**

```
►►──db2flsn──┬────┬──input_LSN────────────────────────────────────────►◄
             └─-q─┘
```

**Command parameters:**

**-q**
Specifies that only the log file name be printed. No error or warning messages will be printed, and status can only be determined through the return code. Valid error codes are:
- -100 Invalid input
- -101 Cannot open LFH file
- -102 Failed to read LFH file
- -103 Invalid LFH
- -104 Database is not recoverable
- -105 LSN too big
- -500 Logical error.

Other valid return codes are:
- 0 Successful execution
- 99 Warning: the result is based on the last known log file size.

**input_LSN**
A 12-character string that represents the internal (6-byte) hexadecimal value with leading zeros.

**Examples:**

```
db2flsn 000000BF0030
   Given LSN is contained in log file S0000002.LOG

db2flsn -q 000000BF0030
   S0000002.LOG

db2flsn 000000BE0030
   Warning: the result is based on the last known log file size.
   The last known log file size is 23 4K pages starting from log extent 2.

   Given LSN is contained in log file S0000001.LOG

db2flsn -q 000000BE0030
   S0000001.LOG
```

**Usage notes:**

The log header control file `SQLOGCTL.LFH` must reside in the current directory. Since this file is located in the database directory, the tool can be run from the database directory, or the control file can be copied to the directory from which the tool will be run.

The tool uses the *logfilsiz* database configuration parameter. DB2 records the three most recent values for this parameter, and the first log file that is created with each *logfilsiz* value; this enables the tool to work correctly when *logfilsiz* changes. If the specified LSN predates the earliest recorded value of *logfilsiz*, the tool uses this value, and returns a warning. The tool can be used with database managers prior to UDB Version 5.2; in this case, the warning is returned even with a correct result (obtained if the value of *logfilsiz* remains unchanged).

This tool can only be used with recoverable databases. A database is recoverable if it is configured with the *logarchmeth1* or *logarchmeth2* configruation parameters set to a value other than `OFF`.

# db2fm - DB2 Fault Monitor

Controls the DB2 fault monitor daemon. You can use **db2fm** to configure the fault monitor.

This command is only available on UNIX-based platforms.

**Authorization:**

Authorization over the instance against which you are running the command.

**Required connection:**

None.

**Command syntax:**

```
►►──db2fm──┬─-t──service──┬──-m──module path──┬──────────────┬──────────────────►◄
           └─-i──instance─┘                   ├─-u───────────┤
                                              ├─-d───────────┤
                                              ├─-s───────────┤
                                              ├─-k───────────┤
                                              ├─-U───────────┤
                                              ├─-D───────────┤
                                              ├─-S───────────┤
                                              ├─-K───────────┤
                                              ├─-f───┬─on──┬─┤
                                              │      └─off─┘ │
                                              ├─-a───┬─on──┬─┤
                                              │      └─off─┘ │
                                              ├─-T──T1/T2────┤
                                              ├─-l──I1/I2────┤
                                              ├─-R──R1/R2────┤
                                              ├─-n──email────┤
                                              ├─-h───────────┤
                                              └─-?───────────┘
```

**Command parameters:**

**-m** *module-path*
: Defines the full path of the fault monitor shared library for the product being monitored. The default is $INSTANCEHOME/sqllib/lib/libdb2gcf.

**-t** *service*
: Gives the unique text descriptor for a service.

**-i** *instance*
: Defines the instance of the service.

**-u** Brings the service up.

**-U** Brings the fault monitor daemon up.

**-d** Brings the service down.

**-D** Brings the fault monitor daemon down.

**-k** Kills the service.

**-K** Kills the fault monitor daemon.

**-s** Returns the status of the service.

**-S** Returns the status of the fault monitor daemon.

**Note:** the status of the service or fault monitor can be one of the following

- Not properly installed,
- INSTALLED PROPERLY but NOT ALIVE,
- ALIVE but NOT AVAILABLE (maintenance),
- AVAILABLE, or
- UNKNOWN

**-f** *on|off*
Turns fault monitor on or off.

**Note:** If this option is set off, the fault monitor daemon will not be started, or the daemon will exit if it was running.

**-a** *on|off*
Activates or deactivate fault monitoring.

**Note:** If this option if set off, the fault monitor will not be actively monitoring, which means if the service goes down it will not try to bring it back.

**-T** *T1/T2*
Overwrites the start and stop time-out.

For example:

- -T 15/10 updates the two time-outs respectively
- -T 15 updates the start time-out to 15 secs
- -T /10 updates the stop time-out to 10 secs

**-I** *I1/I2*
Sets the status interval and time-out respectively.

**-R** *R1/R2*
Sets the number of retries for the status method and action before giving up.

**-n** *email*
Sets the email address for notification of events.

**-h** Prints usage.

**-?** Prints usage.

# db2fs - First Steps

Launches the First Steps interface which contains links to the functions users need to begin learning about and using DB2.

On UNIX-based systems, `db2fs` is located in the `sqllib/bin` directory. On the Windows operating system, `db2fs.bat` is located in the `$DB2PATH\bin` directory.

**Authorization:**

*sysadm*

**Command syntax:**

```
►►──db2fs──────────────────────────────────────────────────────────────►◄
```

**Command parameters:**
None

## db2gcf - Control DB2 Instance

Starts, stops, or monitors a DB2 instance, usually from an automated script, such as in an HA (high availability) cluster.

On UNIX-based systems, this command is located in `INSTHOME/sqllib/bin`, where INSTHOME is the home directory of the instance owner. On Windows systems, this command is located in the `sqllib\bin` subdirectory.

**Authorization:**

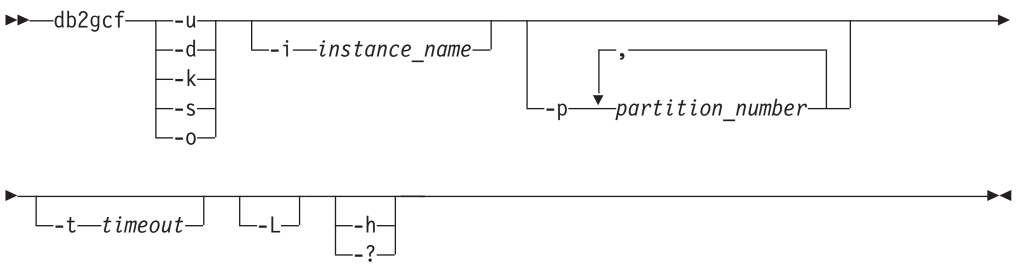
One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

None

**Command syntax:**



**Command parameters:**

**-u** Starts specified partition for specified instance on current database partition server (node).

**-d** Stops specified partition for specified instance.

**-k** Removes all processes associated with the specified instance.

**-s** Returns status of the specified partition and the specified instance. The possible states are:

- *Available*: The specified partition for the specified instance is available for processing.
- *Operable*: The instance is installed but not currently available.
- *Not operable*: The instance will be unable to be brought to available state.

**-o** Returns the default timeouts for each of the possible actions; you can override all these defaults by specifying a value for the -t parameter.

**-i** *instance_name*
Instance name to perform action against. If no instance name is specified, the value of DB2INSTANCE is used. If no instance name is specified and DB2INSTANCE is not set, the following error is returned:

```
db2gcf Error: Neither DB2INSTANCE is set nor instance passed.
```

**-p** *partition_number*
: In a partitioned database environment, specifies partition number to perform action against. If no value is specified, the default is 0. This value is ignored in a single-partition database environment.

**-t** *timeout*
: Timeout in seconds. The **db2gcf** command will return unsuccessfully if processing does not complete within the specified period of time. There are default timeouts for each of the possible actions; you can override all these defaults by specifying a value for the -t parameter.

**-L** Enables error logging. Instance-specific information will be logged to db2diag.log in the instance log directory. Non-instance specific information will be logged to system log files.

**-h/-?** Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Examples:**

1. The following example starts the instance *stevera* on partition 0:

   ```
   db2gcf -u -p 0 -i stevera
   ```

   The following output is returned:

   ```
   Instance  : stevera
   DB2 Start : Success
   Partition 0 : Success
   ```
2. The following example returns the status of the instance *stevera* on partition 0:

   ```
   db2gcf -s -p 0 -i stevera
   ```

   The following output is returned:

   ```
   Instance  : stevera
   DB2 State
   Partition 0 :  Available
   ```
3. The following example stops the instance *stevera* on partition 0:

   ```
   db2gcf -d -p 0 -i stevera
   ```

   The following output is returned:

   ```
   Instance  : stevera
   DB2 Stop : Success
   Partition 0 : Success
   ```

## db2gov - DB2 Governor

Monitors and changes the behavior of applications that run against a database. By default, a daemon is started on every database partition, but the front-end utility can be used to start a single daemon at a specific database partition.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

In an environment with an instance that has a `db2nodes.cfg` file defined, you might also require the authorization to invoke the **db2_all** command. Environments with a `db2nodes.cfg` file defined include partitioned database environments as well as single-partition database environments that have a database partition defined in `db2nodes.cfg`.

**Command syntax:**

```
>>--db2gov------------------------------------------------------------------------------->

>--+-START--datadase--+-------------------------------------+--config-file--log-file-+--------><
   |                  '-DBPARTITIONNUM--db-partition-number-'                        |
   '-STOP--database--+-------------------------------------+-------------------------'
                     '-DBPARTITIONNUM--db-partition-number-'
```

**Command parameters:**

**START database**
Starts the governor daemon to monitor the specified database. Either the database name or the database alias can be specified. The name specified must be the same as the one specified in the governor configuration file.

**Note:** One daemon runs for each database that is being monitored. In a partitioned database environment, one daemon runs for each database partition. If the governor is running for more than one database, there will be more than one daemon running at that database server.

**DBPARTITIONNUM db-partition-number**
Specifies the database partition on which to start or stop the governor daemon. The number specified must be the same as the one specified in the database partition configuration file.

**config-file**
Specifies the configuration file to use when monitoring the database. The default location for the configuration file is the `sqllib` directory. If the specified file is not there, the front-end assumes that the specified name is the full name of the file.

**log-file**
Specifies the base name of the file to which the governor writes log records. The log file is stored in the log subdirectory of the `sqllib` directory. The number of database partitions on which the governor is running is automatically appended to the log file name. For example, `mylog.0, mylog.1, mylog.2`.

**STOP database**

Stops the governor daemon that is monitoring the specified database. In a partitioned database environment, the front-end utility stops the governor on all database partitions by reading the database partition configuration file `db2nodes.cfg`.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODENUM can be substituted for DBPARTITIONNUM.

## db2govlg - DB2 Governor Log Query

Extracts records of specified type from the governor log files. The DB2 governor monitors and changes the behavior of applications that run against a database.

**Authorization:**

None

**Command syntax:**

```
►►──db2govlg──log-file──┬─────────────────────────────────────┬──────────►
                         └─dbpartitionnum──db-partition-number─┘

►──┬────────────────────────┬─────────────────────────────────────────────►◄
   └─rectype──record-type─┘
```

**Command parameters:**

**log-file**
The base name of one or more log files that are to be queried.

**dbpartitionnum db-partition-number**
Number of the database partition on which the governor is running.

**rectype record-type**
The type of record that is to be queried. Valid record types are:

- START
- FORCE
- NICE
- ERROR
- WARNING
- READCFG
- STOP
- ACCOUNT

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword nodenum can be substituted for dbpartitionnum.

**Related reference:**

- "db2gov - DB2 Governor" on page 92

## db2gpmap - Get Partitioning Map

If a database is already set up and database partition groups defined for it, **db2gpmap** gets the partitioning map for the database table or the database partition group from the catalog partitioned database server.

**Authorization:**

Both of the following:

- Read access to the system catalog tables
- BIND and EXECUTE package privileges on `db2gpmap.bnd`

**Required connection:**

Before using **db2gpmap** the database manager must be started and `db2gpmap.bnd` must be bound to the database. If not already bound **db2gpmap** will attempt to bind the file.

**Command syntax:**

```
►►──db2gpmap──┬─────────────────────────┬──┬───────────────────────┬──────────────►
              └─-d──┬─────────────────┬─┘  └─-m──┬───────────────┬─┘
                    └─database-name───┘          └─map-file-name─┘

►──┬───────────────────────────────────────────┬──┬───────────────────┬──┬──────┬──────►◄
   └─-g──┬────────────────────────────────────┬┘  └─-t──table-name────┘  └─-h───┘
         └─database-partition-group-name──────┘
```

**Command parameters:**

**-d** Specifies the name of the database for which to generate a partitioning map. If no database name is specified, the value of the DB2DBDFT environment variable is used. If DB2DBDFT is not set, the default is the SAMPLE database.

**-m** Specifies the fully qualified file name where the partitioning map will be saved. The default is `db2split.map`.

**-g** Specifies the name of the database partition group for which to generate a partitioning map. The default is IBMDEFAULTGROUP.

**-t** Specifies the table name.

**-h** Displays usage information.

**Examples:**

The following example extracts the partitioning map for a table ZURBIE.SALES in database SAMPLE into a file called C:\pmaps\zurbie_sales.map:

```
db2gpmap -d SAMPLE -m C:\pmaps\zurbie_sales.map -t ZURBIE.SALES
```

**Related concepts:**

- "Partitioning maps" in the *Administration Guide: Planning*

## db2hc - Start Health Center

Starts the Health Center. The Health Center is a graphical interface that is used to view the overall health of database systems. Using the Health Center, you can view details and recommendations for alerts on health indicators and take the recommended actions to resolve the alerts.

**Authorization:**

No special authority is required for viewing the information. Appropriate authority is required for taking actions.

**Required Connection:**

Instance

**Command Syntax:**

```
►►──db2hc──┬────┬──┬─────────┬──┬─────────────────────┬──────────────►◄
           └─-t─┘  └─-tcomms─┘  │           ┌──────┐  │
                                │           ▼      │  │
                                └─-tfilter────filter┴─┘
```

**Command Parameters:**

**-t**
: Turns on NavTrace for initialization code. You should use this option only when instructed to do so by DB2 Support.

**-tcomms**
: Limits tracing to communication events. You should use this option only when instructed to do so by DB2 Support.

**-tfilter** *filter*
: Limits tracing to entries containing the specified filter or filters. You should use this option only when instructed to do so by DB2 Support.

# db2iauto - Auto-start Instance

Enables or disables the auto-start of an instance after each system restart. This command is available on UNIX-based systems only.

**Authorization:**

One of the following:

- Root authority
- *sysadm*

**Required connection:**

None

**Command syntax:**

```
►►──db2iauto──┬─-on──┬──instance-name─────────────────────────────────────►◄
              └─-off─┘
```

**Command parameters:**

**-on** Enables auto-start for the specified instance.

**-off** Disables auto-start for the specified instance.

*instance-name*
The login name of the instance.

# db2iclus - Microsoft Cluster Server

Allows users to add, drop, migrate and unmigrate instances and DB2 administration servers (DAS) in a Microsoft Cluster Server (MSCS) environment. This command is only available on Windows platforms.

**Authorization:**

Local administrator authority is required on the machine where the task will be performed. If adding a remote machine to an instance or removing a remote machine from an instance, local administrator authority is required on the target machine.

**Required connection:**

None.

**Command syntax:**

```
>>--db2iclus--+-ADD /u:--username,password--+-------------------+-+------------>
              |                             '-/m:--machine name-' |
              +-DROP--+-------------------+-----------------------+
              |       '-/m:--machine name-'                       |
              +-MIGRATE /p:--InstProfPath-------------------------+
              '-UNMIGRATE-----------------------------------------'

>--+--------------------+--+---------------+--+-------------------+----------><
   '-/i:--instance name-'  '-/DAS:--DAS name-'  '-/c:--cluster name-'
```

**Command parameters:**

**ADD** Adds an MSCS node to a DB2 MSCS instance.

**DROP** Removes an MSCS node from a DB2 MSCS instance.

**MIGRATE**
Migrates a non-MSCS instance to an MSCS instance.

**UNMIGRATE**
Undoes the MSCS migration.

**/DAS:***DAS name*
Specifies the DAS name. This option is required when performing the cluster operation against the DB2 administration server.

**/c:***cluster name*
Specifies the MSCS cluster name if different from the default/current cluster.

**/p:***instance profile path*
Specifies the instance profile path. This path must reside on a cluster disk so it is accessible when DB2 is active on any machine in the MSCS cluster. This option is required when migrating a non-MSCS instance to an MSCS instance.

**/u:***username,password*
Specifies the account name and password for the DB2 service. This option is required when adding another MSCS node to the DB2 MSCS partitioned database instance.

**/m:***machine name*
: Specifies the remote computer name for adding or removing an MSCS node.

**/i:***instance name*
: Specifies the instance name if different from the default/current instance.

**Examples:**

This example shows the use of the **db2iclus** command to manually configure the DB2 instance to run in a hot standby configuration that consists of two machines, WA26 and WA27.

1. To start, MSCS and DB2 UDB Enterprise Server Edition must be installed on both machines.
2. Create a new instance called DB2 on machine WA26:
   ```
   db2icrt DB2
   ```
3. From the Windows Services dialog box, ensure that the instance is configured to start manually.
4. If the DB2 instance is running, stop it with the DB2STOP command.
5. Install the DB2 resource type from WA26:
   ```
   c:>db2wolfi i
   ok
   ```

   If the **db2wolfi** command returns ″Error : 183″, then it is already installed. To confirm, the resource type can be dropped and added again. Also, the resource type will not show up in Cluster Administrator if it does not exist.
   ```
   c:>db2wolfi u
   ok
   c:>db2wolfi i
   ok
   ```
6. From WA26, use the **db2iclus** command to transfrom the DB2 instance into a clustered instance.
   ```
   c:\>db2iclus migrate /i:db2 /c:mycluster /m:wa26 /p:p:\db2profs

   DBI1912I The DB2 Cluster command was successful.
   Explanation:  The user request was successfully processed.
   User Response:  No action required.
   ```

   **Note:** The directory `p:\db2profs` should be on a clustered drive and must already exist. This drive should also be currently owned by machine WA26.
7. From WA26, use the **db2iclus** command to add other machines to the DB2 cluster list:
   ```
   c:\>db2iclus add /i:db2 /c:mycluster /m:wa27

   DBI1912I The DB2 Cluster command was successful.
   Explanation:  The user request was successfully processed.
   User Response:  No action required.
   ```

   This command should be executed for each subsequent machine in the cluster.
8. From Cluster Administrator, create a new group called ″DB2 Group″.
9. From Cluster Administrator, move the Physical Disk resources Disk O and Disk P into DB2 Group.
10. From Cluster Administrator, create a new resource type of type ″IP Address″ called ″mscs5″ that resides on the Public Network. This resource should also

belong to DB2 Group. This will be a highly available IP address, and this address should not correspond to any machine on the network. Bring the IP Address resource type online and ensure that the address can be pinged from a remote machine.

11. From Cluster Administrator, create a new resource of type ″DB2″ that will belong to DB2 Group. The name of this resource must be exactly identical to the instance name, so it is called DB2 for this case. When Cluster Administrator prompts for dependencies associated with the DB2 resource, ensure it is dependent on Disk O, Disk P and mscs5.
12. Configure DB2 Group for fallback, if desired, via Cluster Administrator and using the DB2_FALLBACK profile variable.
13. Create or restore all databases putting all data on Disk O and Disk P.
14. Test the failover configuration.

**Usage notes:**

To migrate an instance to run in an MSCS failover environment, you need to migrate the instance on the current machine first, then add other MSCS nodes to the instance using the **db2iclus** with the ADD option.

To revert an MSCS instance back to a regular instance, you first need to remove all other MSCS nodes from the instance by using the **db2iclus** with the DROP option. Next, you should undo the migration for the instance on the current machine.

# db2icons - Add DB2 icons

Adds DB2 icons and folders to a Linux desktop. This command is only available on Gnome and KDE desktops for supported Intel-based Linux distributions. It is located in the `DB2DIR/bin` directory, where DB2DIR represents `/opt/IBM/db2/V8.1`. It is also located in `/sqllib/bin` in the `home` directory of the instance owner.

**Authorization:**

One of the following:

- To invoke the command for other users: root authority or authority to write to the `home` directories of the specified users
- To invoke the command for your own desktop: none

**Required connection:**

None

**Command syntax:**

```
               .-----------.
               V           |
>>--db2icons------user_name-+-----------------------------------------><
```

**Command parameters:**

*user_name*
User ID for which you want to add desktop icons.

**Usage notes:**

If icons are generated while a Gnome or KDE desktop environment is running, the user might need to force a manual desktop refresh to see the new icons.

## db2icrt - Create Instance

Creates DB2 instances.

On Windows operating systems, the db2icrt utility is located in the `\sqllib\bin` subdirectory.

On UNIX-based systems, the db2icrt utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems. If you have a FixPak or modification level installed in an alternate path, the `DB2DIR` directory is `usr/opt/db2_08_FPn` on AIX and `opt/IBM/db2/V8.FPn` on all other UNIX-based systems, where `n` represents the number of the FixPak or modification level. The db2icrt utility creates an instance on the directory from which you invoke it.

**Authorization:**

Root access on UNIX-based systems or Local Administrator authority on Windows operating systems.

**Command syntax:**

**For UNIX-based systems**

```
►►──db2icrt──┬──────┬──┬─────┬──┬─────────────────┬──┬─────────────────┬──────────────────►
             ├──-h──┤  └──-d─┘  └──-a──AuthType──┘  └──-p──PortName──┘
             └──-?──┘

►──┬─────────────────┬──┬──────────────────┬──┬─────────────────┬──InstName──────────────►◄
   └──-s──InstType──┘  └──-w──WordWidth──┘  └──-u──FencedID──┘
```

**For Windows operating systems**

```
►►──db2icrt──┬─────────────────┬──┬──────────────────────────────┬──────────────────────────►
             └──-s──InstType──┘  └──-u──UserName, Password──┘

►──┬─────────────────────┬──┬─────────────────┬──┬──────────────────┬──InstName──────────►◄
   └──-p──InstProfPath──┘  └──-h──HostName──┘  └──-r──PortRange──┘
```

**Command parameters:**
**For UNIX-based systems**

**-h or -?**
Displays the usage information.

**-d**
Turns debug mode on. Use this option only when instructed by DB2 Support.

**-a** *AuthType*
Specifies the authentication type (SERVER, CLIENT or SERVER_ENCRYPT) for the instance. The default is SERVER.

**-p** *PortName*
Specifies the port name or number used by the instance. This option does not apply to client instances.

**-s** *InstType*
Specifies the type of instance to create. Use the -s option only when you are creating an instance other than the default for your system. Valid values are:

**CLIENT**
Used to create an instance for a client.

**ESE** Used to create an instance for a database server with local and remote clients.

**Note:** Specify this option if you are creating an instance for a PE database system, a single-partition ESE database system, or DB2 Connect.

**WSE** Used to create an instance for a Workgroup Server Edition server.

**-w** *WordWidth*
Specifies the width, in bits, of the instance to be created (31, 32 or 64). You must have the requisite version of DB2 installed (31-bit, 32-bit, or 64-bit) to be able to select the appropriate width. The default value is the lowest bit width supported, and depends on the installed version of DB2 UDB, the platform it is operating on, and the instance type. This parameter is only valid on AIX 5L, HP-UX, and the Solaris Operating Environment.

**-u** *Fenced ID*
Specifies the name of the user ID under which fenced user-defined functions and fenced stored procedures will run. The -u option is requred if you are creating a server instance.

*InstName*
Specifies the name of the instance.

**For Windows operating systems**

**-s** *InstType*
Specifies the type of instance to create. Valid values are:

**Client** Used to create an instance for a client.

**Note:** Use this value if you are using DB2 Connect Personal Edition.

**Standalone**
Used to create an instance for a database server with local clients.

**ESE** Used to create an instance for a database server with local and remote clients.

**Note:** Specify this option if you are creating an instance for a PE database system, a single partition ESE database system, or DB2 Connect.

**WSE** Used to create an instance for a Workgroup Server Edition server.

**-u** *Username, Password*
Specifies the account name and password for the DB2 service. This option is required when creating a partitioned database instance.

**-p** *InstProfPath*
Specifies the instance profile path.

**-h** *HostName*
Overrides the default TCP/IP host name if there is more than one for the current machine. The TCP/IP host name is used when creating the default database partition (database partition 0). This option is only valid for partitioned database instances.

**-r** *PortRange*
Specifies a range of TCP/IP ports to be used by the partitioned database instance when running in MPP mode. The services file of the local machine will be updated with the following entries if this option is specified:

```
DB2_InstName         baseport/tcp
DB2_InstName_END     endport/tcp
```

*InstName*
Specifies the name of the instance.

**Examples:**

**Example 1:**

On an AIX machine, to create an instance called "db2inst1" on the directory `/u/db2inst1/sqllib/bin`, issue the following command from that directory:

On a client machine:

```
usr/opt/db2_08_01/instance/db2icrt db2inst1
```

On a server machine:

```
usr/opt/db2_08_01/instance/db2icrt -u db2fenc1 db2inst1
```

where db2fenc1 is the user ID under which fenced user-defined functions and fenced stored procedures will run.

**Example 2:**

On an AIX machine, if you have Alternate FixPak 1 installed, run the following command to create an instance running FixPak 1 code from the Alternate FixPak 1 install path:

```
/usr/opt/db2_08_FP1/instance/db2icrt -u db2fenc1 db2inst1
```

**Usage notes:**

The -s option is intended for situations in which you want to create an instance that does not use the full functionality of the system. For example, if you are using Enterprise Server Edition (ESE), but do not want partition capabilities, you could create a Workgroup Server Edition (WSE) instance, using the option `-s WSE`.

To create a DB2 instance that supports Microsoft Cluster Server, first create an instance, then use the **db2iclus** command to migrate it to run in a MSCS instance.

**Related reference:**
- "db2iclus - Microsoft Cluster Server" on page 98

# db2idrop - Remove Instance

Removes a DB2 instance that was created by db2icrt. Removes the instance entry from the list of instances.

On Windows operating systems, the db2idrop utility is located in the `\sqllib\bin` subdirectory. On UNIX-based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems. If you have a FixPak or modification level installed in an alternate path, the `DB2DIR` directory is `usr/opt/db2_08_FPn` on AIX and `opt/IBM/db2/V8.FPn` on all other UNIX-based systems, where `n` represents the number of the FixPak or modification level.

If you have a FixPak or modification level installed in an alternate path, you can drop any instance by running the db2idrop utility from an installation path; to do this, the install code must still be located in the installation path of the instance that you are dropping. If you remove install code from an installation path, and then try to drop the instance in that path by invoking the db2idrop utility from a different installation path, you will not be able to drop the instance.

For example, consider a setup in which both DB2 Version 8 and DB2 Version 8 Alternate FixPak 1 are installed. An instance `db2inst1` has been created in the Version 8 install path and an instance `db2inst2` has been created in the Alternate FixPak 1 (AFP1) path.

- If no installation code has been removed, then running the db2idrop utility from the AFP1 path will allow you to drop both instances, even though `db2inst1` is created to run against code in the Version 8 path.
- If installation code in the Version 8 path has been removed, you will not be able to drop `db2inst1` by invoking db2idrop from the AFP1 path. However, it will still be possible to drop `db2inst2`.

**Authorization:**

Root access on UNIX based systems or Local Administrator on Windows operating systems.

**Command syntax:**

**For UNIX Based Systems**

```
►►──db2idrop──┬──────┬──InstName─────────────────────────────────────►◄
              ├──-h──┤
              └──-?──┘
```

**For Windows Operating Systems**

```
►►──db2idrop──┬──────┬──InstName──────────────────────────────────────►◄
              └──-f──┘
```

**Command parameters:**
**For UNIX Based Systems**

**-h or -?**
Displays the usage information.

**InstName**
Specifies the name of the instance.

**For Windows Operating Systems**

**-f** Specifies the force applications flag. If this flag is specified all the applications using the instance will be forced to terminate.

**InstName**
Specifies the name of the instance.

**Example:**

On an AIX machine, an instance named ″db2inst1″ is running Version 8 code in the Version 8 install path. An instance named ″db2inst2″ is running Version 8 FixPak 1 code in the Alternate FixPak 1 install path. The command to drop db2inst1 can be issued from the Alternate FixPak 1 install path:

```
/usr/opt/db2_08_FP1/instance/db2idrop db2inst1
```

**Usage notes:**

In a partitioned database environment, if more than one database partition belongs to the instance that is being dropped, the **db2idrop** command has to be run on each database partition so that the DB2 registry on each database partition is updated.

**Related reference:**

- "db2icrt - Create Instance" on page 102

# db2ilist - List Instances

Lists all the instances that are available on a system.

On Windows operating systems, this utility is located in the `\sqllib\bin` subdirectory. On UNIX-based systems, this utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems. If you have a FixPak or modification level installed in an alternate path, the `DB2DIR` directory is `usr/opt/db2_08_FPn` on AIX and `opt/IBM/db2/V8.FPn` on all other UNIX-based systems, where `n` represents the number of the FixPak or modification level.

You can issue this command from any code path (for example an Alternate FixPak code path) with identical results.

**Authorization:**

None

**Command syntax:**

**For Windows operating systems**

```
►►──db2ilist──────────────────────────────────────────────────────►◄
```

**For UNIX-based systems**

```
►►──db2ilist──┬──────────────┬──┬────┬──┬────┬──┬───────────┬──────────────►◄
              └─-w──┬─31─┬───┘  └─-p─┘  └─-a─┘  └─inst_name─┘
                    ├─32─┤
                    └─64─┘
```

**Command parameters:**

**-w** Lists the 31-, 32-, or 64-bit instances. The -w option can be specified with the -p option, and is superseded by the -a option.

**-p** Lists the DB2 install path that an instance is running from. The -p option can be used with the -a option, and is superseded by the -a option.

**-a** Lists information including the DB2 install path associated with an instance, as well as its bit width (32 or 64). The returned information for 32-bit indicates 31-bit for DB2 on Linux (on S/390 and zSeries).

**-inst_name**
Lists the information for the specified instance. If no instance is named, **db2ilist** lists information about all instances of the current DB2 release.

**-h** Displays usage information.

**Examples:**

Consider an AIX system with four instances:

- 32-bit instance called ″db2inst1″ installed on the Version 8 code path
- 64-bit instance called ″db2inst2″ installed on the Version 8 code path

- 32-bit instance called ″db2inst3″ installed on the Version 8, Alternate FixPak 1 code path
- 64-bit instance called ″db2inst4″ installed on the Version 8, Alternate FixPak 1 code path

Issuing the db2ilist command will produce different output, depending on the parameters of the command:

- db2list

```
db2inst1
db2inst2
db2inst3
db2inst4
```

- db2ilist -a

```
db2inst1  32  /usr/opt/db2_08_01
db2inst2  64  /usr/opt/db2_08_01
db2inst3  32  /usr/opt/db2_08_FP1
db2inst4  64  /usr/opt/db2_08_FP1
```

- db2ilist -w 64 -p

```
db2inst2  /usr/opt/db2_08_01
db2inst4  /usr/opt/db2_08_FP1
```

## db2imigr - Migrate Instance

Migrates an existing instance following installation of the database manager. This command is available on UNIX-based systems only. On Windows, instance migration is done implicitly during migration.

This utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems.

**Note:** Migration of an instance from a DB2 Version 7 to a Version 8 FixPak or modification level installed in an alternate path is only supported for Enterprise Server Edition. If your Version 7 instance includes other products, such as Warehouse Manager or Data Links Manager, you cannot migrate those products to a FixPak or modification level installed in an alternate path.

**Authorization:**

Root authority on UNIX-based systems.

**Command syntax:**

```
►►──db2imigr──┬────┬──┬─────────────┬──┬────────────────┬──InstName──────────►◄
              └─-d─┘  └─-a──AuthType─┘  ├─-u──FencedID───┤
                                        └─-g──dlfmxgrpid─┘
```

**Command parameters:**

**-d**
: Turns debug mode on. Use this option only when instructed by DB2 Support.

**-a** *AuthType*
: Specifies the authentication type (SERVER, CLIENT or SERVER_ENCRYPT) for the instance. The default is SERVER.

**-u** *FencedID*
: Specifies the name of the user ID under which fenced user-defined functions and fenced stored procedures will run. This option is not required if only a DB2 client is installed.

**-g** *dlfmxgrpid*
: Specifies the dlfmxgrp ID. This option must only be used if migrating a Data Links File Manager instance Version 7 or earlier. The system group ID specified here is exclusively for use with the Data Links File Manager. The DLFM database instance owner, which is `dlfm` by default, will be the only system user ID defined as a member of this group.

*InstName*
: Specifies the name of the instance.

**Usage notes:**

If they exist, **db2imigr** removes the symbolic links in `/usr/lib` and `/usr/include` to the version you are migrating from. If you have applications that load `libdb2` directly from `/usr/lib` rather than using the operating system's library environment variable to find it, your applications might fail to execute properly after you have run **db2imigr**.

**Related concepts:**

- "Before you install DB2 Data Links Manager (AIX)" in the *Quick Beginnings for Data Links Manager*
- "Before you install DB2 Data Links Manager (Solaris Operating Environment)" in the *Quick Beginnings for Data Links Manager*

# db2inidb - Initialize a Mirrored Database

Initializes a mirrored database in a split mirror environment. The mirrored database can be initialized as a clone of the primary database, placed in roll forward pending state, or used as a backup image to restore the primary database. This command can only be run against a split mirror database, and it must be run before the split mirror can be used.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

None

**Command syntax:**

```
►►──db2inidb──database_alias──AS──┬─SNAPSHOT─┬──┬─────────────────────────────┬──►◄
                                  ├─STANDBY──┤  └─RELOCATE USING──configFile─┘
                                  └─MIRROR───┘
```

**Command parameters:**

**database_alias**
Specifies the alias of the database to be initialized.

**SNAPSHOT**
Specifies that the mirrored database will be initialized as a clone of the primary database.

**STANDBY**
Specifies that the database will be placed in roll forward pending state.

**Note:** New logs from the primary database can be fetched and applied to the standby database. The standby database can then be used in place of the primary database if it goes down.

**MIRROR**
Specifies that the mirrored database is to be used as a backup image which can be used to restore the primary database.

**RELOCATE USING** *configFile*
Specifies that the database files are to be relocated based on the information listed in the specified *configFile* prior to initializing the database as a snapshot, standby, or mirror. The format of *configFile* is described in db2relocatedb - Relocate Database Command.

**Usage notes:**

In a partitioned database environment, **db2inidb** must be run on every partition before the split mirror from any of the partitions can be used. **db2inidb** can be run on all partitions simultaneously using the **db2_all** command.

If; however, you are using the RELOCATE USING option, you cannot use the **db2_all** command to run **db2inidb** on all of the partitions simultaneously. A separate configuration file must be supplied for each partition, that includes the NODENUM value of the partition being changed. For example, if the name of a database is being changed, every partition will be affected and the **db2relocatedb** command must be run with a separate configuration file on each partition. If containers belonging to a single database partition are being moved, the **db2relocatedb** command only needs to be run once on that partition.

If the RELOCATE USING *configFile* parameter is specified and the database is relocated successfully, the specified *configFile* will be copied into the database directory and renamed to `db2path.cfg`. During a subsequent crash recovery or rollforward recovery, this file will be used to rename container paths as log files are being processed.

If a clone database is being initialized, the specified *configFile* will be automatically removed from the database directory after a crash recovery is completed.

If a standby database or mirrored database is being initialized, the specified *configFile* will be automatically removed from the database directory after a rollforward recovery is completed or canceled. New container paths can be added to the `db2path.cfg` file after **db2inidb** has been run. This would be necessary when CREATE or ALTER TABLESPACE operations are done on the original database and different paths must be used on the standby database.

**Related tasks:**

- "Using a split mirror to clone a database" in the *Data Recovery and High Availability Guide and Reference*
- "Using a split mirror as a standby database" in the *Data Recovery and High Availability Guide and Reference*
- "Using a split mirror as a backup image" in the *Data Recovery and High Availability Guide and Reference*

**Related reference:**

- "db2relocatedb - Relocate Database" on page 194
- "rah and db2_all command descriptions" in the *Administration Guide: Implementation*

# db2inspf - Format inspect results

This utility formats the data from INSPECT CHECK results into ASCII format. Use this utility to see details of the inspection. The formatting by the db2inspf utility can be format for a table only, or a table space only, or for errors only, or warnings only, or summary only.

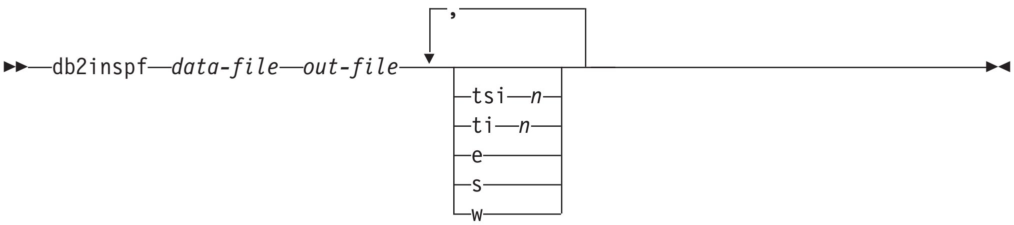
**Authorization:**

Anyone can access the utility, but users must have read permission on the results file in order to execute this utility against them.

**Required connection:**

None

**Command syntax:**



**Command Parameters:**

**data-file**
The unformatted inspection results file to format.

**out-file**
The output file for the formatted output.

**-tsi** *n* Table space ID. Format out only for tables in this table space.

**-ti** *n* Table ID. Format out only for table with this ID, table space ID must also be provided.

**-e** Format out errors only.

**-s** Summary only.

**-w** Warnings only.

# db2isetup - Start Instance Creation Interface

Starts the DB2 Instance Setup wizard, a graphical tool for creating instances and for configuring new functionality on existing instances. For instance, if you create an instance and then install more products such as DB2 Spatial Extender, issuing this command will start the graphical interface used to configure the DB2 Spatial Extender functionality on your existing instance.

**Authorization:**

Root authority on the system where the command is issued.

**Required connection:**

None.

**Command syntax:**

```
►►──db2isetup──┬────────────────┬──┬──────────────┬──┬─────────────────────┬──────────►
               └─-t──tracefile──┘  └─-l──logfile──┘  └─-i──language-code──┘

►──┬──────┬─────────────────────────────────────────────────────────────────────────►◄
   └─-?───┘
```

**Command parameters:**

**-t** *tracefile*
: The full path and name of trace file specified by *tracefile*.

**-l** *logfile*
: Full path and name of the log file. If no name is specified, the path and filename default to `/tmp/db2isetup.log`

**-i** *language-code*
: Two letter code for the preferred language in which to run the install. If unspecified, this parameter will default to the locale of the current user.

**-?, -h** Output usage information.

**Usage notes:**

1. This instance setup wizard provides a subset of the functionality provided by the DB2 Setup wizard. The DB2 Setup wizard (which runs from the installation media) allows you to install DB2 components, do system setup tasks such as DAS creation/configuration, and set up instances. The DB2 Instance Setup wizard only provides the functionality pertaining to instance setup.
2. The executable file for this command is located in the `/product install dir/instance` directory, along with other instance scripts such as `db2icrt` and `db2iupdt`. Like these other instance scripts, it requires root authority, and like these other instance scripts, it is not part of the DB2 instance on UNIX.
3. db2isetup runs on all supported UNIX platforms.

## db2iupdt - Update Instances

On Windows operating systems, this command updates single-partition instances for use in a partitioned database system. It is located in the `\sqllib\bin` subdirectory.

On UNIX-based systems, this command updates a specified DB2 instance to enable acquisition of a new system configuration or access to function associated with the installation or removal of certain product options, FixPaks, or modification levels. This utility is located in the `DB2DIR/instance` directory, where `DB2DIR` represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems. If you have a FixPak or modification level installed in an alternate path, the `DB2DIR` directory is `usr/opt/db2_08_FPn` on AIX and `opt/IBM/db2/V8.FPn` on all other UNIX-based systems, where `n` represents the number of the FixPak or modification level.

If you have a FixPak or modification level installed in an alternate path, you can update any instance by running the **db2iupdt** utility from an installation path. The **db2iupdt** command will update an instance to run against the code installed in the same path from which the command was issued.

**Authorization:**

Root access on UNIX based systems or Local Administrator on Windows.

**Command syntax:**

**For UNIX-based systems**

```
►►─db2iupdt──┬─────┬──┬────┬──┬────┬──┬────┬──┬────┬──┬──────────────┬──────────►
             ├─-h──┤  └─-d─┘  └─-k─┘  └─-D─┘  └─-s─┘  └─-a──AuthType─┘
             └─-?──┘

►──┬──────────────┬──┬─────────────┬──┬─InstName─┬──────────────────────────►◄
   └─-w──WordWidth┘  └─-u──FencedID┘  └─-e───────┘
```

**For Windows**

```
►►─db2iupdt──InstName──/u:──username,password──┬──────────────────────────────┬──►
                                               └─/p:──instance profile path───┘

►──┬──────────────────────────┬──┬──────────────┬────────────────────────────►◄
   └─/r:──baseport,endport────┘  └─/h:──hostname┘
```

**Command parameters:**
**For UNIX-based systems**

**-h or -?**
Displays the usage information.

**-d** Turns debug mode on.

**-k** Keeps the current instance type during the update.

**-D** Moves an instance from a higher code level on one path to a lower code level installed on another path.

**-s** Ignores the existing SPM log directory.

**-a AuthType**
Specifies the authentication type (SERVER, SERVER_ENCRYPT or CLIENT) for the instance. The default is SERVER.

**-w WordWidth**
Specifies the width, in bits, of the instance to be created. Valid values are 31, 32, and 64. This parameter is only valid on AIX, HP-UX, Linux for AMD64, and the Solaris Operating Environment. The requisite version of DB2 must be installed (31-bit, 32-bit, or 64-bit). The default value is the bit width of the instance that is being updated.

**-u Fenced ID**
Specifies the name of the user ID under which fenced user defined functions and fenced stored procedures will run.

**InstName**
Specifies the name of the instance.

**-e** Updates every instance.

**For Windows**

**InstName**
Specifies the name of the instance.

**/u:username,password**
Specifies the account name and password for the DB2 service.

**/p:instance profile path**
Specifies the new instance profile path for the updated instance.

**/r:baseport,endport**
Specifies the range of TCP/IP ports to be used by the partitioned database instance when running in MPP mode. When this option is specified, the services file on the local machine will be updated with the following entries:

```
DB2_InstName       baseport/tcp
DB2_InstName_END   endport/tcp
```

**/h:hostname**
Overrides the default TCP/IP host name if there are more than one TCP/IP host names for the current machine.

**Examples (UNIX):**

1. An instance, ″db2inst1″, is running Version 8, FixPak 1 code in the Version 8 install path. If Version 8.1.2 is installed in the Version 8 install path, the following command, invoked from the Version 8 install path, will update db2inst1 to Version 8.1.2:

   ```
   db2iupdt db2inst1
   ```

2. An instance, ″db2inst2″, is running Version 8.1.2 code in the Version 8 install path. If you then install Version 8.1.2 in an alternate install path, the following command, invoked from the Version 8.1.2 alternate install path, will update db2inst2 to Version 8.1.2, running from the alternate install path:

   ```
   db2iupdt db2inst2
   ```

3. An instance, ″db2inst3″, is running Version 8.1.2 code in an alternate install path. If FixPak 1 is installed in another alternate install path, the following command, invoked from the FixPak 1 alternate install path, will update db2inst3 to FixPak 1, running from the FixPak 1 alternate install path:

```
db2iupdt -D db2inst3
```

**Usage notes:**

If you are switching the install path that an instance will run from, and you are updating the instance from a higher code level to a lower code level (for example, switching among multiple levels of DB2), you will have to use the **db2iupdt** command with the -D option. This situation is illustrated in the third example.

## db2jdbcbind - DB2 JDBC Package Binder

This utility is used to bind or rebind the JDBC packages to a DB2 database. DB2 Version 8 databases already have the JDBC packages preinstalled, therefore, this command is usually necessary only for downlevel servers.

**Note:** JDBC and CLI share the same packages. If the CLI packages have already been bound to a database, then it is not necessary to run this utility and vice versa.

**Authorization:**

One of the following:

- *sysadm*
- *dbadm*
- BINDADD privilege if a package does not exist, and one of:
  - IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
  - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists

**Required connection:**

This command establishes a database connection.

**Command syntax:**

```
►►──db2jdbcbind──┬──────┬──-url jdbc:db2://server:port/dbname──-user──username────►
                 └─help─┘

►──-password──password──┬─────────────────────────────────┬───────────────────────►
                        └─-collection──collection ID──────┘

►──┬────────────────────────────────┬──────────────────────────────────────────────►
   └─-size──number of packages──────┘
```

```
                 ┌─,──────────────────────────────┐
                 ▼                                │
►►─-tracelevel───┬─TRACE_ALL────────────────────┬─┴──────────────►◄
                 ├─TRACE_CONNECTION_CALLS───────┤
                 ├─TRACE_CONNECTS───────────────┤
                 ├─TRACE_DIAGNOSTICS────────────┤
                 ├─TRACE_DRDA_FLOWS─────────────┤
                 ├─TRACE_DRIVER_CONFIGURATION───┤
                 ├─TRACE_NONE───────────────────┤
                 ├─TRACE_PARAMETER_META_DATA────┤
                 ├─TRACE_RESULT_SET_CALLS───────┤
                 ├─TRACE_RESULT_SET_META_DATA───┤
                 └─TRACE_STATEMENT_CALLS────────┘
```

**Command parameters:**

**-help** Displays help information, all other options are ignored.

**-url jdbc:db2://server:port/dbname**
Specifies a JDBC URL for establishing the database connection. The DB2 JDBC type 4 driver is used to establish the connection.

**-user** *username*
Specifies the name used when connecting to a database.

**-password** *password*
Specifies the password for the user name.

**-collection** *collection ID*
The collection identifier (CURRENT PACKAGESET), to use for the packages. The default is NULLID. Use this to create multiple instances of the package set. This option can only be used in conjunction with the Connection or DataSource property currentPackageSet.

**-size** *number of packages*
The number of internal packages to bind for each DB2 transaction isolation level and holdability setting. The default is 3. Since there are four DB2 isolation levels and two cursor holdability settings, there will be 4x2=8 times as many dynamic packages bound as are specified by this option. In addition, a single static package is always bound for internal use.

**-tracelevel**
Identifies the level of tracing, only required for troubleshooting.

## db2ldcfg - Configure LDAP Environment

Configures the Lightweight Directory Access Protocol (LDAP) user distinguished name (DN) and password for the current logon user in an LDAP environment using an IBM LDAP client.

**Authorization:**

*none*

**Required connection:**

None

**Command syntax:**

```
►►──db2ldcfg──┬─-u──user's Distinguished Name ──-w──password─┬──────────────────►◄
              └─-r───────────────────────────────────────────┘
```

**Command parameters:**

**-u user's Distinguished Name**
Specifies the LDAP user's Distinguished Name to be used when accessing the LDAP directory. As shown in the example below, the Distinguished name has several parts: the user ID, such as `jdoe`, the domain and organization names and the suffix, such as `com` or `org`.

**-w password**
Specifies the password.

**-r**
Removes the user's DN and password from the machine environment.

**Example:**

```
db2ldcfg -u "uid=jdoe,dc=mydomain,dc=myorg,dc=com" -w password
```

**Usage notes:**

In an LDAP environment using an IBM LDAP client, the default LDAP user's DN and password can be configured for the current logon user. Once configured, the LDAP user's DN and password are saved in the user's environment and used whenever DB2 accesses the LDAP directory. This eliminates the need to specify the LDAP user's DN and password when issuing the LDAP command or API. However, if the LDAP user's DN and password are specified when the command or API is issued, the default settings will be overridden.

This command can only be run when using an IBM LDAP client. On a Microsoft LDAP client, the current logon user's credentials will be used.

## db2level - Show DB2 Service Level

Shows the current Version and Service Level of the installed DB2 product. Output from this command goes to the console by default.

**Authorization:**

None.

**Required Connection:**

None.

**Command Syntax:**

```
►►──db2level──────────────────────────────────────────────────────────────►◄
```

**Examples:**
A typical result of running the db2level command on a Windows system would be:

```
DB21085I  Instance "kirton" uses DB2 code release "SQL08010" with level
identifier "01010106" and informational tokens "DB2 v8.1.0", "n020320" and "".
```

The information output by the command includes Release, Level, and various informational tokens.

## db2licm - License Management Tool

Performs basic license functions in the absence of the Control Center. Adds, removes, lists, and modifies licenses and policies installed on the local system.

**Authorization:**

On UNIX-based systems, root authority is required only to remove a license key. On Windows operating systems, no authorization is required.

**Required connection:**

None

**Command syntax:**

```
►►──db2licm──────────────────────────────────────────────────────────────────►

►──┬─────────────────────────────────────────────────────────────┬──────────►◄
   ├─-a──filename────────────────────────────────────────────────┤
   ├─-l──────────────────────────────────────────────────────────┤
   ├─-p──prod-password──┬────────────┬──┬────────────┬──┬──────────┬──┤
   │                    └─REGISTERED─┘  └─CONCURRENT─┘  ├─INTERNET─┤  │
   │                                                    └─MEASURED─┘  │
   ├─-r──prod-password───────────────────────────────────────────┤
   ├─-u──prod-password──num-users────────────────────────────────┤
   ├─-n──prod-password──num-processors───────────────────────────┤
   ├─-e──prod-password──┬─HARD─┬─────────────────────────────────┤
   │                    └─SOFT─┘                                 │
   ├─-v──────────────────────────────────────────────────────────┤
   └──┬─-h─┬─────────────────────────────────────────────────────┘
      └─-?─┘
```

**Command parameters:**

**-a filename**
: Adds a license for a product. Specify a file name containing valid license information.

**-l**
: Lists all the products with available license information.

**-p prod-password** *keyword*
: Updates the license policy type to use on the system. The keywords CONCURRENT, REGISTERED, or CONCURRENT REGISTERED can be specified. In addition, you can specify INTERNET for DB2 UDB Workgroup Server products, or MEASURED for DB2 Connect Unlimited products.

**-r prod-password**
: Removes the license for a product. After the license is removed, the product functions in "Try & Buy" mode. To get the password for a specific product, invoke the **db2licm** command with the -l option.

**-u prod-password num-users**
: Updates the number of user licenses that the customer has purchased. Specifies the number of users and the password of the product for which the licenses were purchased.

**-n prod-password num-processors**
Updates the number of processors on which the customer is licensed to use DB2.

**-e prod-password**
Updates the enforcement policy on the system. Valid values are: HARD and SOFT. HARD specifies that unlicensed requests will not be allowed. SOFT specifies that unlicensed requests will be logged but not restricted.

**-v** Displays version information.

**-h/-?** Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Examples:**

```
db2licm -a db2ese.lic
db2licm -p db2wse registered concurrent
db2licm -r db2ese
db2licm -u db2wse 10
db2licm -n db2ese 8
```

**Related tasks:**

- "Registering the DB2 product license key using the db2licm command" in the *Installation and Configuration Supplement*

# db2logsforrfwd - List Logs Required for Rollforward Recovery

Parses the `DB2TSCHG.HIS` file. This utility allows a user to find out which log files are required for a table space rollforward operation. This utility is located in `sqllib/bin`.

**Authorization:**

**Required connection:**

None.

**Command syntax:**

```
►►──db2logsforrfwd──path──┬──────┬──────────────────────────────────────────►◄
                          └─-all─┘
```

**Command parameters:**

*path* Full path and name of the `DB2TSCHG.HIS` file.

**-all** Displays more detailed information.

**Examples:**

```
db2logsForRfwd /home/ofer/ofer/NODE0000/S0000001/DB2TSCHG.HIS
db2logsForRfwd DB2TSCHG.HIS -all
```

## db2look - DB2 Statistics and DDL Extraction Tool

Extracts the required DDL (data definition language) statements to reproduce the database objects of a production database on a test database. **db2look** generates the DDL statements by object type.

This tool can generate the required UPDATE statements used to replicate the statistics on the objects in a test database. It can also be used to generate UPDATE DATABASE CONFIGURATION and UPDATE DATABASE MANAGER CONFIGURATION commands and **db2set** commands so that query optimizer-related configuration parameters and registry variables on the test database match those of the production database.

It is often advantageous to have a test system contain a subset of the production system's data. However, access plans selected for such a test system are not necessarily the same as those that would be selected for the production system. Both the catalog statistics and the configuration parameters for the test system must be updated to match those of the production system. Using this tool makes it possible to create a test database where access plans are similar to those that would be used on the production system.

**Note:** The DDL generated might not exactly reproduce all characteristics of the original SQL objects. Check the DDL generated by **db2look**.

**Authorization:**

SELECT privilege on the system catalog tables.

In some cases, such as generating table space container DDL (which calls the APIs sqlbotcq, sqlbftcq, and sqlbctcq), you will require one of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

None. This command establishes a database connection.

**Command syntax:**

```
►►──db2look──-d──DBname──┬────┬──┬────────────┬──┬───────────┬──────────────────►
                         └-e──┘  └-u──Creator─┘  └-z──schema─┘

►──┬─────────────────────────────────────────┬──┬──────────────────┬──┬────┬─────►
   │                                         │  │  ┌────────────┐  │  └-h──┘
   │                        └-tw──Tname──┘   │  │  ▼            │  │
   │     ┌────────────┐                      │  └-v─────Vname───┴──┘
   │     ▼            │  │
   └-t─────Tname──────┴──┘

►──┬──────────┬──┬────┬──┬────────────────────┬──┬────┬──┬────┬──┬─────┬─────────►
   └-o──Fname─┘  └-a──┘  └-m──┬────┬──┬────┬──┘  └-l──┘  └-x──┘  └-xd──┘
                              └-c──┘  └-r──┘
```

```
►──┬────┬──┬──────────────────┬──┬────┬──┬────┬──┬────┬──┬─────────┬──────────────────►
   └─-f─┘  └─-td──delimiter───┘  └─-p─┘  └─-s─┘  └─-g─┘  └─-noview─┘

►──┬────────────────────────────┬──┬──────────────────────┬──┬────────┬─────────────►◄
   └─-i──userid──-w──password───┘  ├─-wrapper──Wname──────┤  └─-nofed─┘
                                   └─-server──Sname───────┘
```

**Command parameters:**

**-d** *DBname*
: Alias name of the production database that is to be queried. *DBname* can be the name of a DB2 UDB for UNIX, Windows, or DB2 Universal Database for z/OS and OS/390 database. If the *DBname* is a DB2 Universal Database for z/OS and OS/390 database, the **db2look** utility will extract the DDL and UPDATE statistics statements for OS/390 and z/OS objects. These DDL and UPDATE statistics statements are statements applicable to a DB2 UDB database and not to a DB2 Universal Database for z/OS and OS/390 database. This is useful for users who want to extract OS/390 and z/OS objects and recreate them in a DB2 UDB database.

If *DBname* is a DB2 Universal Database for z/OS and OS/390 database, the **db2look** output is limited to the following:

- Generate DDL for tables, indexes, views, and user-defined distinct types
- Generate UPDATE statistics statements for tables, columns, column distributions and indexes

**-e**
: Extract DDL statements for database objects. DDL for the following database objects are extracted when using the `-e` option:

- Tables
- Views
- Automatic summary tables (AST)
- Aliases
- Indexes
- Triggers
- Sequences
- User-defined distinct types
- Primary key, referential integrity, and check constraints
- User-defined structured types
- User-defined functions
- User-defined methods
- User-defined transforms
- Wrappers
- Servers
- User mappings
- Nicknames
- Type mappings
- Function templates
- Function mappings
- Index specifications
- Stored procedures

**Note:** The DDL generated by **db2look** can be used to recreate user-defined functions successfully. However, the user source code that a particular user-defined function references (the EXTERNAL NAME clause, for example) must be available in order for the user-defined function to be usable.

**-u** *Creator*
Creator ID. Limits output to objects with this creator ID. If option `-a` is specified, this parameter is ignored. If neither `-u` nor `-a` is specified, the environment variable USER is used.

**-z** *schema*
Schema name. Limits output to objects with this schema name. If option -a is specified, this parameter is ignored. If this parameter is not specified, objects with all schema names are extracted. This option is ignored for the federated DDL.

**-t** *Tname1 Tname2 ... TnameN*
Table name list. Limits the output to particular tables in the table list. The maximum number of tables is 30. Table names are separated by a blank space. Case-sensitive names must be enclosed inside a backward slash and double quotation delimiter, for example, \″ MyTabLe \″. For multiple-word table names, the delimiters must be placed within quotation marks (for example, ″\″My Table\″″) to prevent the pairing from being evaluated word-by-word by the command line processor. If a multiple-word table name is not enclosed by the backward slash and double delimiter (for example, ″My Table″), all words will be converted into uppercase and **db2look** will look for an uppercase table (for example, ″MY TABLE″).

**-tw** *Tname*
Generates DDL for table names that match the pattern criteria specified by *Tname*. Also generates the DDL for all dependent objects of all returned tables. *Tname* can be a single value only. The underscore character (_) in *Tname* represents any single character. The percent sign (%) represents a string of zero or more characters. Any other character in *Tname* only represents itself. When -tw is specified, the -t option is ignored.

**-v** *Vname1 Vname2 ... VnameN*
Generates DDL for the specified views. The maximum number of views is 30. If the -t option is specified, the -v option is ignored.

**-h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-o** *Fname*
If using LaTeX format, write the output to *filename* `.tex`. If using plain text format, write the output to *filename*`.txt`. Otherwise, write the output to *filename*`.sql`. If this option is not specified, output is written to standard output. If a filename is specified with an extension, the output will be written into that file.

**-a** When this option is specified the output is not limited to the objects created under a particular creator ID. All objects created by all users are considered. For example, if this option is specified with the `-e` option, DDL statements are extracted for all objects in the database. If this option is specified with the `-m` option, UPDATE statistics statements are extracted for all user created tables and indexes in the database.

**Note:** If neither -u nor -a is specified, the environment variable USER is used. On UNIX-based systems, this variable does not have to be explicitly set; on Windows systems, however, there is no default value for the USER environment variable: a user variable in the SYSTEM variables must be set, or a set USER=<username> must be issued for the session.

**-m** Generate the required UPDATE statements to replicate the statistics on tables, columns and indexes. The -p, -g, and -s options are ignored when the -m option is specified. The -c, and -r options are optionally used with -m.

**-c** When this option is specified in conjunction with the -m option, **db2look** does not generate COMMIT, CONNECT and CONNECT RESET statements. The default action is to generate these statements. The -c option is ignored if the -m option is not specified.

**-r** When this option is specified in conjunction with the -m option, **db2look** does not generate the RUNSTATS command. The default action is to generate the RUNSTATS command. The -r option is ignored if the -m option is not specified.

**-l** If this option is specified, then the **db2look** utility will generate DDL for user defined table spaces, database partition groups and buffer pools. DDL for the following database objects is extracted when using the -l option:

- User-defined table spaces
- User-defined database partition groups
- User-defined buffer pools

**-x** If this option is specified, the **db2look** utility will generate authorization DDL (GRANT statement, for example).

The authorizations supported by db2look include:

- Table: ALTER, SELECT, INSERT, DELETE, UPDATE, INDEX, REFERENCE, CONTROL
- View: SELECT, INSERT, DELETE, UPDATE, CONTROL
- Index: CONTROL
- Schema: CREATEIN, DROPIN, ALTERIN
- Database: CREATEDB, BINDADD, CONNECT, CREWATE_NOT_FENCED, IMPLICIT_SCHEMA
- User-defined function (UDF): EXECUTE
- User-defined method: EXECUTE
- Stored procedure: EXECUTE
- Package: CONTROL, BIND, EXECUTE
- Column: UPDATE, REFERENCES
- Tablespace: USE
- Sequence: USAGE, ALTER

**-xd** If this option is specified, the db2look utility will generate all authorization DDL including authorization DDL for objects whose authorizations were granted by SYSIBM at object creation time.

**-f** Use this option to extract the configuration parameters and registry variables that affect the query optimizer.

The **db2look** utility generates an update command for the following configuration parameters:

- Database manager configuration parameters
  - cpuspeed
  - intra_parallel
  - comm_bandwidth
  - nodetype
  - federated
  - fed_noauth
- Database configuration parameters
  - locklist
  - dft_degree
  - maxlocks
  - avg_appls
  - stmtheap
  - dft_queryopt

The **db2look** utility generates the **db2set** command for the following DB2 registry variables:

- DB2_PRED_FACTORIZE
- DB2_CORRELATED_PREDICATES
- DB2_LIKE_VARCHAR
- DB2_SORT_AFTER_TQ
- DB2_HASH_JOIN
- DB2_ORDERED_NLJN
- DB2_NEW_CORR_SQ_FF
- DB2_PART_INNER_JOIN
- DB2_INTERESTING_KEYS

**-td** *delimiter*

Specifies the statement delimiter for SQL statements generated by **db2look**. If this option is not specified, the default is the semicolon (;). It is recommended that this option be used if the `-e` option is specified. In this case, the extracted objects might contain triggers or SQL routines.

**-p** Use plain text format.

**-s** Generate a PostScript file.

**Notes:**

1. This option removes all LaTeX and `.tmp` PostScript files.
2. Required non-IBM software: LaTeX, dvips.
3. The `psfig.tex` file must be in the LaTeX input path.

**-g** Use a graph to show fetch page pairs for indexes.

**Notes:**

1. This option generates a *filename*.`ps` file, as well as the LaTeX file.
2. Required non-IBM software: Gnuplot.
3. The `psfig.tex` file must be in the LaTeX input path.

**-noview**
If this option is specified, CREATE VIEW DDL statements will not be extracted.

**-i** *userid*
Use this option when working with a remote database.

**-w** *password*
Used with the `-i` option, this parameter allows the user to run **db2look** against a database that resides on a remote system. The user ID and the password are used by **db2look** to log on to the remote system.

**Note:** If working with remote databases, the remote database must be the same version as the local database. The **db2look** utility does not have down-level or up-level support.

**-wrapper** *Wname*
Generates DDL statements for federated objects that apply to this wrapper. The federated DDL statements that might be generated include: CREATE WRAPPER, CREATE SERVER, CREATE USER MAPPING, CREATE NICKNAME, CREATE TYPE MAPPING, CREATE FUNCTION ... AS TEMPLATE, CREATE FUNCTION MAPPING, CREATE INDEX SPECIFICATION, and GRANT (privileges to nicknames, servers, indexes). Only one wrapper name is supported; an error is returned if less than one or more than one is specified. This option cannot be used if the `-server` option is used.

**-server** *Sname*
Generates DDL statements for federated objects that apply to this server. The federated DDL statements that might be generated include: CREATE WRAPPER, CREATE SERVER, CREATE USER MAPPING, CREATE NICKNAME, CREATE TYPE MAPPING, CREATE FUNCTION ... AS TEMPLATE, CREATE FUNCTION MAPPING, CREATE INDEX SPECIFICATION, and GRANT (privileges to nicknames, servers, indexes). Only one server name is supported; an error is returned if less than one or more than one is specified. This option cannot be used if the `-wrapper` option is used.

**-nofed** Specifies that no federated DDL statements will be generated. When this option is specified, the -wrapper and -server options are ignored.

**Examples:**

- Generate the DDL statements for objects created by user *walid* in database DEPARTMENT. The **db2look** output is sent to file `db2look.sql`:

```
db2look -d department -u walid -e -o db2look.sql
```

- Generate the DDL statements for objects that have schema name *ianhe*, created by user *walid*, in database DEPARTMENT. The **db2look** output is sent to file `db2look.sql`:

```
db2look -d department -u walid -z ianhe -e -o db2look.sql
```

- Generate the UPDATE statements to replicate the statistics for the tables and indexes created by user *walid* in database DEPARTMENT. The output is sent to file `db2look.sql`:

```
db2look -d department -u walid -m -o db2look.sql
```

- Generate both the DDL statements for the objects created by user *walid* and the UPDATE statements to replicate the statistics on the tables and indexes created by the same user. The **db2look** output is sent to file `db2look.sql`:

```
db2look -d department -u walid -e -m -o db2look.sql
```

- Generate the DDL statements for objects created by all users in the database DEPARTMENT. The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -a -e -o db2look.sql
  ```
- Generate the DDL statements for all user-defined database partition groups, buffer pools and table spaces. The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -l -o db2look.sql
  ```
- Generate the UPDATE statements for optimizer-related database and database manager configuration parameters, as well as the **db2set** statements for optimizer-related registry variables in database DEPARTMENT. The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -f -o db2look.sql
  ```
- Generate the DDL for all objects in database DEPARTMENT, the UPDATE statements to replicate the statistics on all tables and indexes in database DEPARTMENT, the GRANT authorization statements, the UPDATE statements for optimizer-related database and database manager configuration parameters, the **db2set** statements for optimizer-related registry variables, and the DDL for all user-defined database partition groups, buffer pools and table spaces in database DEPARTMENT. The output is sent to file `db2look.sql`.

  ```
  db2look -d department -a -e -m -l -x -f -o db2look.sql
  ```
- Generate all authorization DDL statements for all objects in database DEPARTMENT, including the objects created by the original creator. (In this case, the authorizations were granted by SYSIBM at object creation time.) The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -xd -o db2look.sql
  ```
- Generate the DDL statements for objects created by all users in the database DEPARTMENT. The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -a -e -td % -o db2look.sql
  ```

  The output can then be read by the CLP:

  ```
  db2 -td% -f db2look.sql
  ```
- Generate the DDL statements for objects in database DEPARTMENT, excluding the CREATE VIEW statements. The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -e -noview -o db2look.sql
  ```
- Generate the DDL statements for objects in database DEPARTMENT related to specified tables. The **db2look** output is sent to file `db2look.sql`:

  ```
  db2look -d department -e -t tab1 \"My TaBlE2\" -o db2look.sql
  ```
- Generate the DDL statements for all objects (federated and non-federated) in the federated database FEDDEPART. For federated DDL statements, only those that apply to the specified wrapper, FEDWRAP, are generated. The **db2look** output is sent to standard output:

  ```
  db2look -d feddepart -e -wrapper fedwrap
  ```
- Generate a script file that includes only non-federated DDL statements. The following system command can be run against a federated database (FEDDEPART) and yet only produce output like that found when run against a database which is not federated. The **db2look** output is sent to a file `out.sql`:

  ```
  db2look -d feddepart -e -nofed -o out
  ```

**Usage notes:**

On Windows systems, **db2look** must be run from a DB2 command window.

**db2look** command line options can be specified in any order. All command line options are optional except the -d option which is mandatory and must be followed by a valid database alias name.

Several of the existing options support a federated environment. The following **db2look** command line options are used in a federated environment:

- `-e`

  When used, federated DDL statements are generated.
- `-x`

  When used, GRANT statements are generated to grant privileges to the federated objects.
- `-xd`

  When used, federated DDL statements are generated to add system-granted privileges to the federated objects.
- `-f`

  When used, federated-related information is extracted from the database manager configuration.
- `-m`

  When used, statistics for nicknames are extracted.

The ability to use federated systems needs to be enabled in the database manager configuration in order to create federated DDL statements. After the **db2look** command generates the script file, you must set the *federated* configuration parameter to YES before running the script.

You need to modify the output script to add the remote passwords for the CREATE USER MAPPING statements.

You need to modify the **db2look** command output script by adding AUTHORIZATION and PASSWORD to those CREATE SERVER statements that are used to define a DB2 family instance as a data source.

Usage of the -tw option is as follows:

- To both generate the DDL statements for objects in the DEPARTMENT database associated with tables that have names beginning with `abc` and send the output to the `db2look.sql` file:

  ```
  db2look -d department -e -tw abc% -o db2look.sql
  ```
- To generate the DDL statements for objects in the DEPARTMENT database associated with tables that have a `d` as the second character of the name and to send the output to the `db2look.sql` file:

  ```
  db2look -d department -e -tw _d% -o db2look.sql
  ```
- **db2look** uses the LIKE predicate when evaluating which table names match the pattern specified by the *Tname* argument. Because the LIKE predicate is used, if either the _ character or the % character is part of the table name, the backslash (\) escape character must be used immediately before the _ or the %. In this situation, neither the _ nor the % can be used as a wildcard character in *Tname*. For example, to generate the DDL statements for objects in the DEPARTMENT database associated with tables that have a percent sign in the neither the first nor the last position of the name:

  ```
  db2look -d department -e -tw string\%string
  ```
- Case-sensitive and multi-word table names must be enclosed by both a backslash and double quotation marks. For example:

```
\"My TabLe\"
```

.

- The -tw option can be used with the -x option (to generate GRANT privileges), the -m option (to return table and column statistics), and the -l option (to generate the DDL for user-defined table spaces, database partition groups, and buffer pools). If the -t option is specified with the -tw option, the -t option (and its associated *Tname* argument) is ignored.
- The -tw option accepts only one *Tname* argument.
- The -tw option cannot be used to generate the DDL for tables (and their associated objects) that reside on federated data sources, or on DB2 Universal Database for z/OS and OS/390, DB2 Universal Database for iSeries, or DB2 Server for VSE & VM.
- The -tw option is only supported via the CLP.

**Related reference:**

- "LIKE predicate" in the *SQL Reference, Volume 1*

## db2move - Database Movement Tool

This tool facilitates the movement of large numbers of tables between DB2 databases located on workstations. The tool queries the system catalog tables for a particular database and compiles a list of all user tables. It then exports these tables in PC/IXF format. The PC/IXF files can be imported or loaded to another local DB2 database on the same system, or can be transferred to another workstation platform and imported or loaded to a DB2 database on that platform.

**Note:** Tables with structured type columns are not moved when this tool is used.

**Authorization:**

This tool calls the DB2 export, import, and load APIs, depending on the action requested by the user. Therefore, the requesting user ID must have the correct authorization required by those APIs, or the request will fail.

**Command syntax:**

```
                                  .-----------------------.
                                  V                       |
>>--db2move--dbname--action-------+-----------------------+---------------><
                                  +--tc--table-creators---+
                                  +--tn--table-names------+
                                  +--sn--schema-names-----+
                                  +--ts--tablespace-names-+
                                  +--tf--filename---------+
                                  +--io--import-option----+
                                  +--lo--load-option------+
                                  +--l--lobpaths----------+
                                  +--u--userid------------+
                                  +--p--password----------+
                                  '--aw-------------------'
```

**Command parameters:**

**dbname**
Name of the database.

**action** Must be one of: EXPORT, IMPORT, or LOAD.

**-tc** table-creators. The default is all creators.

This is an EXPORT action only. If specified, only those tables created by the creators listed with this option are exported. If not specified, the default is to use all creators. When specifying multiple creators, they must be separated by commas; no blanks are allowed between creator IDs. The maximum number of creators that can be specified is 10. This option can be used with the "-tn" table-names option to select the tables for export.

An asterisk (*) can be used as a wildcard character that can be placed anywhere in the string.

**-tn** table-names. The default is all user tables.

This is an EXPORT action only. If specified, only those tables whose names match exactly those in the specified string are exported. If not specified, the default is to use all user tables. When specifying multiple table names, they must be separated by commas; no blanks are allowed between table names. The maximum number of table names that can be specified is 10.

This option can be used with the “-tc” table-creators option to select the tables for export. **db2move** will only export those tables whose names match the specified table names and whose creators match the specified table creators.

An asterisk (*) can be used as a wildcard character that can be placed anywhere in the string.

**-sn** schema-names. The default is all schemas.

If specified, only those tables whose schema names match exactly will be exported. If the asterisk wildcard character (*) is used in the schema names, it will be changed to a percent sign (%) and the table name (with percent sign) will be used in the LIKE predicate of the WHERE clause. If not specified, the default is to use all schemas. If multiple schema names are specified, they must be separated by commas; no blanks are allowed between schema names. The maximum number of schema names that can be specified is 10. If used with the -tn or -tc option, **db2move** will export only those tables whose schemas match the specified schema names and whose creators match the specified creators.

**Note:** Schema names of less than 8 character are padded to 8 characters in length. For example, a schema name ′fred′ has to be specified ″-sn fr*d*″ instead of ″-sn fr*d″ when using an asterisk.

**-ts** tablespace-names. The default is all table spaces.

This is an EXPORT action only. If this option is specified, only those tables that reside in the specified table space will be exported. If the asterisk wildcard character (*) is used in the table space name, it will be changed to a percent sign (%) and the table name (with percent sign) will be used in the LIKE predicate in the WHERE clause. If the -ts option is not specified, the default is to use all table spaces. If multiple table space names are specified, they must be separated by commas; no blanks are allowed between table space names. The maximum number of table space names that can be specified is 10.

**Note:** Table space names less than 8 characters are padded to 8 characters in length. For example, a table space name ′mytb′ has to be specified ″-ts my*b*″ instead of ″-sn my*b″ when using the asterisk.

**-tf** filename

This is an EXPORT action only. If specified, only the tables listed in the given file will be exported. The tables should be listed one per line, and each table should be fully qualified. Here is an example of the contents of a file:

```
"SCHEMA1"."TABLE NAME1"
"SCHEMA NAME77"."TABLE155"
```

**-io** import-option. The default is REPLACE_CREATE.

Valid options are: INSERT, INSERT_UPDATE, REPLACE, CREATE, and REPLACE_CREATE.

**-lo** load-option. The default is INSERT.

Valid options are: INSERT and REPLACE.

**-l** lobpaths. The default is the current directory.

This option specifies the absolute path names where LOB files are created (as part of EXPORT) or searched for (as part of IMPORT or LOAD). When specifying multiple LOB paths, each must be separated by commas; no blanks are allowed between LOB paths. If the first path runs out of space (during EXPORT), or the files are not found in the path (during IMPORT or LOAD), the second path will be used, and so on.

If the action is EXPORT, and LOB paths are specified, all files in the LOB path directories are deleted, the directories are removed, and new directories are created. If not specified, the current directory is used for the LOB path.

**-u** userid. The default is the logged on user ID.

Both user ID and password are optional. However, if one is specified, the other must be specified. If the command is run on a client connecting to a remote server, user ID and password should be specified.

**-p** password. The default is the logged on password.

Both user ID and password are optional. However, if one is specified, the other must be specified. If the command is run on a client connecting to a remote server, user ID and password should be specified.

**-aw** Allow Warnings. When '-aw' is not specified, tables that experience warnings during export are not included in the `db2move.lst` file (although that table's .ixf file and .msg file are still generated). In some scenarios (such as data truncation) the user might wish to allow such tables to be included in the `db2move.lst` file. Specifing this option allows tables which receive warnings during export to be included in the .lst file.

**Examples:**

- `db2move sample export`

  This will export all tables in the SAMPLE database; default values are used for all options.
- `db2move sample export -tc userid1,us*rid2 -tn tbname1,*tbname2`

  This will export all tables created by "userid1" or user IDs LIKE "us%rid2", and with the name "tbname1" or table names LIKE "%tbname2".
- `db2move sample import -l D:\LOBPATH1,C:\LOBPATH2`

  This example is applicable to the Windows operating system only. The command will import all tables in the SAMPLE database; LOB paths "D:\LOBPATH1" and "C:\LOBPATH2" are to be searched for LOB files.
- `db2move sample load -l /home/userid/lobpath,/tmp`

  This example is applicable to UNIX based systems only. The command will load all tables in the SAMPLE database; both the `/home/userid/lobpath` subdirectory and the `tmp` subdirectory are to be searched for LOB files.
- `db2move sample import -io replace -u userid -p password`

  This will import all tables in the SAMPLE database in REPLACE mode; the specified user ID and password will be used.

**Usage notes:**

This tool exports, imports, or loads user-created tables. If a database is to be duplicated from one operating system to another operating system, **db2move** facilitates the movement of the tables. It is also necessary to move all other objects associated with the tables, such as aliases, views, triggers, user-defined functions, and so on. If the import utility with the REPLACE_CREATE option is used to create the tables on the target database, then the limitations outlined in Using import to recreate an exported table are imposed. If unexpected errors are encountered during the **db2move** import phase when the REPLACE_CREATE option is used, examine the appropriate `tabnnn.msg` message file and consider that the errors might be the result of the limitations on table creation.

When export, import, or load APIs are called by **db2move**, the `FileTypeMod` parameter is set to `lobsinfile`. That is, LOB data is kept in separate files from PC/IXF files. There are 26 000 file names available for LOB files.

The LOAD action must be run locally on the machine where the database and the data file reside. When the load API is called by **db2move**, the `CopyTargetList` parameter is set to NULL; that is, no copying is done. If *logretain* is on, the load operation cannot be rolled forward later. The table space where the loaded tables reside is placed in backup pending state, and is not accessible. A full database backup, or a table space backup, is required to take the table space out of backup pending state.

**Note:** 'db2move import' performance can be improved by altering default buffer pool, IBMDEFAULTBP; and by updating the configuration parameters *sortheap*, *util_heap_sz*, *logfilsz*, and *logprimary*.

**Files Required/Generated When Using EXPORT:**

- Input: None.
- Output:

**EXPORT.out** The summarized result of the EXPORT action.

**db2move.lst** The list of original table names, their corresponding PC/IXF file names (tabnnn.ixf), and message file names (tabnnn.msg). This list, the exported PC/IXF files, and LOB files (tabnnnc.yyy) are used as input to the **db2move** IMPORT or LOAD action.

**tabnnn.ixf** The exported PC/IXF file of a specific table.

**tabnnn.msg** The export message file of the corresponding table.

**tabnnnc.yyy** The exported LOB files of a specific table.

"nnn" is the table number. "c" is a letter of the alphabet. "yyy" is a number ranging from 001 to 999.

These files are created only if the table being exported contains LOB data. If created, these LOB files are placed in the "lobpath" directories. There are a total of 26 000 possible names for the LOB files.

**system.msg** The message file containing system messages for creating or deleting file or directory commands. This is only used if the action is EXPORT, and a LOB path is specified.

**Files Required/Generated When Using IMPORT:**

- Input:

**db2move.lst** An output file from the EXPORT action.

**tabnnn.ixf** An output file from the EXPORT action.

**tabnnnc.yyy** An output file from the EXPORT action.

- Output:

  **IMPORT.out** The summarized result of the IMPORT action.

  **tabnnn.msg** The import message file of the corresponding table.

**Files Required/Generated When Using LOAD:**

- Input:

  **db2move.lst** An output file from the EXPORT action.

  **tabnnn.ixf** An output file from the EXPORT action.

  **tabnnnc.yyy** An output file from the EXPORT action.
- Output:

  **LOAD.out** The summarized result of the LOAD action.

  **tabnnn.msg** The LOAD message file of the corresponding table.

**Related reference:**

- "db2look - DB2 Statistics and DDL Extraction Tool" on page 125

# db2mqlsn - MQ Listener

Invokes the asynchronous MQListener to monitor a set of WebSphere MQ message queues, passing messages that arrive on them to configured DB2 stored procedures. It can also perform associated administrative and configuration tasks. MQListener configuration information is stored in a DB2 database and consists of a set of named configurations, including a default. Each configuration is composed of a set of tasks. MQListener tasks are defined by the message queue from which to retrieve messages and the stored procedure to which they will be passed. The message queue description must include the name of the message queue and its queue manager, if it is not the default. Information about the stored procedure must include the database in which it is defined, a user name and password with which to access the database, and the procedure name and schema.

On Windows operating systems, **db2mqlsn** is located in the `sqllib\bin` subdirectory. On UNIX-based systems, this command is located in the DB2DIR/instance directory, where DB2DIR represents `/usr/opt/db2_08_01` on AIX, and `/opt/IBM/db2/V8.1` on all other UNIX-based systems.

For more information about controlling access to WebSphere MQ objects, refer to the *WebSphere MQ System Administration Guide* (SC34-6068-00).

**Authorization:**

- All options except **db2mqlsn admin** access the MQListener configuration in the configDB database. The connection is made as configUser or, if no user is specified, an implicit connection is attempted. The user in whose name the connection is made must have EXECUTE privilege on package mqlConfi.
- To access MQ objects with the **db2mqlsn run** and **db2mqlsn admin** options, the user who executes the program must be able to open the appropriate MQ objects.
- To execute the **db2mqlsn run** option successfully, the dbUser specified in the **db2mqlsn add** option that created the task must have EXECUTE privilege on the specified stored procedure, and must have EXECUTE privilege on the package mqlRun in the dbName database.

**Command syntax:**

```
►►──db2mqlsn──┬─help──┬─────────┬────────────────────────────────┬──────►◄
              │       └─command─┘                                │
              ├─run──┤ configuration ├──┤ run parameters ├───────┤
              ├─add──┤ configuration ├──┤ add parameters ├───────┤
              ├─remove──┤ configuration ├──┤ remove parameters ├─┤
              ├─show──┤ configuration ├──────────────────────────┤
              └─admin──┤ admin parameters ├──────────────────────┘
```

### configuration:

```
├──-configDB──configuration database name─────────────────────────────►

►──┬──────────────────────────────────────────────┬───────────────────►
   └─-configUser──user ID──-configPwd──password───┘
```

```
►─┬──────────────────────────────┬──────────────────────────────────────────────────┤
  └─-config──configuration name──┘
```

**run parameters:**

```
├──┬──────────────────────────────────────────────────────────────────────┬────────┤
   └─-adminQueue──admin queue name──┬──────────────────────────────────┬──┘
                                    └─-adminQMgr──admin queue manager──┘
```

**add parameters:**

```
├──-inputQueue──input queue name──┬──────────────────────────────────────┬──────────►
                                  └─-queueManager──queue manager name──┘

►──-procSchema──stored procedure schema──-procName──stored procedure name───────────►

►──-dbName──stored procedure database──┬────────────────────────────────────┬───────►
                                       └─-dbUser──user ID──-dbPwd──password─┘

►─┬─────────────────┬──┬──────────────────────────────────────────────┬─────────────┤
  └─-mqCoordinated──┘  └─-numInstances──number of instances to run──┘
```

**remove parameters:**

```
├──-inputQueue──input queue name──┬──────────────────────────────────────┬─────────┤
                                  └─-queueManager──queue manager name──┘
```

**admin parameters:**

```
├──┬─-adminQueue──admin queue name─────────────────────┬──┬──────────────────────────────────┬──►
   └─-adminQueueList──namelist of admin queue names──┘  └─-adminQMgr──admin queue manager──┘

►──-adminCommand──┬─shutdown─┬─────────────────────────────────────────────────────────┤
                  └─restart──┘
```

**Command parameters:**

**help** *command*
: Supplies detailed information about a particular command. If you do not give a command name, then a general help message is displayed.

**–configDB** *configuration database*
: Name of the database that contains the configuration information.

**–configUser** *user ID* **–configPwd** *password*
: Authorization information with which to access the configuration database.

**–config** *configuration name*
: You can group individual tasks into a configuration. By doing this you can run a group of tasks together. If you do not specify a configuration name, then the utility runs the default configuration.

**run**

**–adminQueue** *admin queue name* **–adminQMgr** *admin queue manager*
: This is the queue on which the MQListener listens for administration commands. If you do not specify a queue manager, then the utility uses the configured default queue manager. If you do not specify an adminQueue, then the application does not

receive any administration commands (such as shut down or restart) through the message queue.

**add**

**–inputQueue** *input queue name* **–queueManager** *queue manager name*
This is the queue on which the MQListener listens for messages for this task. If you do not specify a queue manager, the utility uses the default queue manager configured in WebSphere MQ.

**–procSchema** *stored procedure schema* **–procName** *stored procedure name*
The stored procedure to which MQListener passes the message when it arrives.

**–dbName** *stored procedure database*
MQListener passes the message to a stored procedure. This is the database in which the stored procedure is defined.

**–dbUser** *user ID* **–dbPwd** *password*
The user on whose behalf the stored procedure is invoked.

**–mqCoordinated**
This indicates that reading and writing to the WebSphere MQ message queue should be integrated into a transaction together with the DB2 stored procedure call. The entire transaction is coordinated by the WebSphere MQ coordinator. (Note that the queue manager must also be configured to coordinate a transaction in this way. See the WebSphere MQ documentation for more information.) By default, the message queue operations are not part of the transaction in which the stored procedure is invoked.

**–numInstances** *number of instances to run*
The number of duplicate instances of this task to run in this configuration. If you do not specify a value, then only one instance is run.

**remove**

**–inputQueue** *input queue name* **–queueManager** *queue manager name*
This is the queue and queue manager that define the task that will be removed from the configuration. The combination of input queue and queue manager is unique within a configuration.

**admin**

**–adminQueue** *admin queue name* **–adminQueueList** *namelist of admin queue names* **–adminQMgr** *admin queue manager*
The queue or namelist of queue names on which to send the admin command. If you do not specify a queue manager, the utility uses the default queue manager that is configured in WebSphere MQ.

**–adminCommand** *admin command*
Submits a command. The command can be either shutdown or restart. Shutdown causes a running MQListener to exit when the listener finishes processing the current message. Restart performs a shutdown, and then reads the configuration again and restarts.

**Examples:**

```
db2mqlsn show -configDB sampleDB -config nightlies
```

```
db2mqlsn add -configDB sampleDB -config nightlies -inputQueue app3
-procSchema imauser -procName proc3 -dbName aDB -dbUser imauser -dbPwd aSecret
```

```
db2mqlsn run -configDB -config nightlies
```

**Related concepts:**

- "Asynchronous messaging in DB2 Information Integrator" in the *IBM DB2 Information Integrator Application Developer's Guide*
- "How to use WebSphere MQ functions within DB2" in the *IBM DB2 Information Integrator Application Developer's Guide*

## db2mscs - Set up Windows Failover Utility

Creates the infrastructure for DB2 failover support on Windows using Microsoft Cluster Server (MSCS). This utility can be used to enable failover in both single-partition and partitioned database environments.

**Authorization:**

The user must be logged on to a domain user account that belongs to the Administrators group of each machine in the MSCS cluster.

**Command syntax:**

```
►►──db2mscs──┬────────────────────┬──────────────────────────────────────►◄
             ├─-f:──input_file────┤
             └─-u:──instance_name─┘
```

**Command parameters:**

**-f:input_file**
: Specifies the `DB2MSCS.CFG` input file to be used by the MSCS utility. If this parameter is not specified, the DB2MSCS utility reads the `DB2MSCS.CFG` file that is in the current directory.

**-u:instance_name**
: This option allows you to undo the db2mscs operation and revert the instance back to the non-MSCS instance specified by `instance_name`.

**Usage notes:**

The DB2MSCS utility is a standalone command line utility used to transform a non-MSCS instance into an MSCS instance. The utility will create all MSCS groups, resources, and resource dependencies. It will also copy all DB2 information stored in the Windows registry to the cluster portion of the registry as well as moving the instance directory to a shared cluster disk. The DB2MSCS utility takes as input a configuration file provided by the user specifying how the cluster should be set up. The DB2MSCS.CFG file is an ASCII text file that contains parameters that are read by the DB2MSCS utility. You specify each input parameter on a separate line using the following format: `PARAMETER_KEYWORD=parameter_value`. For example:

```
CLUSTER_NAME=FINANCE
GROUP_NAME=DB2 Group
IP_ADDRESS=9.21.22.89
```

Two example configuration files can be found in the `CFG` subdirectory under the DB2 install directory. The first, `DB2MSCS.EE`, is an example for single-partition database environments. The second, `DB2MSCS.EEE`, is an example for partitioned database environments.

The parameters for the DB2MSCS.CFG file are as follows:

**DB2_INSTANCE**
: The name of the DB2 instance. This parameter has a global scope and should be specified only once in the DB2MSCS.CFG file.

**DAS_INSTANCE**
: The name of the DB2 Admin Server instance. Specify this parameter to

migrate the DB2 Admin Server to run in the MSCS environment. This parameter has a global scope and should be specified only once in the DB2MSCS.CFG file.

**CLUSTER_NAME**
The name of the MSCS cluster. All the resources specified following this line are created in this cluster until another CLUSTER_NAME parameter is specified.

**DB2_LOGON_USERNAME**
The user name of the domain account for the DB2 service (specified as *domain\user*). This parameter has a global scope and should be specified only once in the DB2MSCS.CFG file.

**DB2_LOGON_PASSWORD**
The password of the domain account for the DB2 service. This parameter has a global scope and should be specified only once in the DB2MSCS.CFG file.

**GROUP_NAME**
The name of the MSCS group. If this parameter is specified, a new MSCS group is created if it does not exist. If the group already exists, it is used as the target group. Any MSCS resource specified after this parameter is created in this group or moved into this group until another GROUP_NAME parameter is specified. Specify this parameter once for each group.

**DB2_NODE**
The partition number of the database partition server (or database partition) to be included in the current MSCS group. If multiple logical database partitions exist on the same machine, each database partition requires a separate DB2_NODE parameter. Specify this parameter after the GROUP_NAME parameter so that the DB2 resources are created in the correct MSCS group. This parameter is required for a multi-partitioned database system.

**IP_NAME**
The name of the IP Address resource. The value for the IP_NAME is arbitrary, but it must be unique in the cluster. When this parameter is specified, an MSCS resource of type IP Address is created. This parameter is required for remote TCP/IP connections. This parameter is optional in a single partition environment. A recommended name is the hostname that corresponds to the IP address.

**IP_ADDRESS**
The TCP/IP address for the IP resource specified by the preceding IP_NAME parameter. This parameter is required if the IP_NAME parameter is specified. This is a new IP address that is not used by any machine in the network.

**IP_SUBNET**
The TCP/IP subnet mask for the IP resource specified by the preceding IP_NAME parameter. This parameter is required if the IP_NAME parameter is specified.

**IP_NETWORK**
The name of the MSCS network to which the preceding IP Address resource belongs. This parameter is optional. If it is not specified, the first

MSCS network detected by the system is used. The name of the MSCS network must be entered exactly as seen under the Networks branch in Cluster Administrator.

**Note:** The previous four IP keywords are used to create an IP Address resource.

**NETNAME_NAME**
The name of the Network Name resource. Specify this parameter to create the Network Name resource. This parameter is optional for single partition database environment. You must specify this parameter for the instance owning machine in a partitioned database environment.

**NETNAME_VALUE**
The value for the Network Name resource. This parameter must be specified if the NETNAME_NAME parameter is specified.

**NETNAME_DEPENDENCY**
The name for the IP resource that the Network Name resource depends on. Each Network Name resource must have a dependency on an IP Address resource. This parameter is optional. If it is not specified, the Network Name resource has a dependency on the first IP resource in the group.

**SERVICE_DISPLAY_NAME**
The display name of the Generic Service resource. Specify this parameter if you want to create a Generic Service resource.

**SERVICE_NAME**
The service name of the Generic Service resource. This parameter must be specified if the SERVICE_DISPLAY_NAME parameter is specified.

**SERVICE_STARTUP**
Optional startup parameter for the Generic Resource service.

**DISK_NAME**
The name of the physical disk resource to be moved to the current group. Specify as many disk resources as you need. The disk resources must already exist. When the DB2MSCS utility configures the DB2 instance for failover support, the instance directory is copied to the first MSCS disk in the group. To specify a different MSCS disk for the instance directory, use the INSTPROF_DISK parameter. The disk name used should be entered exactly as seen in Cluster Administrator.

**INSTPROF_DISK**
An optional parameter to specify an MSCS disk to contain the DB2 instance directory. If this parameter is not specified the DB2MSCS utility uses the first disk that belongs to the same group.

**INSTPROF_PATH**
An optional parameter to specify the exact path where the instance directory will be copied. This parameter *must* be specified when using IPSHAdisks, a ServerRAID Netfinity disk resource (for example, INSTPROF_PATH=p:\db2profs). INSTPROF_PATH will take precedence over INSTPROF_DISK if both are specified.

**TARGET_DRVMAP_DISK**
An optional parameter to specify the target MSCS disk for database drive mapping for a the multi-partitioned database system. This parameter will specify the disk the database will be created on by mapping it from the

drive the create database command specifies. If this parameter is not specified, the database drive mapping must be manually registered using the DB2DRVMP utility.

**DB2_FALLBACK**
An optional parameter to control whether or not the applications should be forced off when the DB2 resource is brought offline. If not specified, then the setting for DB2_FALLBACK will beYES. If you do not want the applications to be forced off, then set DB2_FALLBACK to NO.

## db2mtrk - Memory Tracker

Provide complete report of memory status, for instances, databases and agents. This command outputs the following memory pool allocation information:

- Current size
- Maximum size (hard limit)
- Largest size (high water mark)
- Type (identifier indicating function for which memory will be used)
- Agent who allocated pool (only if the pool is private)

The same information is also available from the Snapshot monitor.

**Scope**

In a partitioned database environment, this command can be invoked from any database partition defined in the db2nodes.cfg file. It returns information only for that partition. This command does not return information for remote servers.

**Authorization:**

One of the following:

- sysadm
- sysctrl
- sysmaint

**Required Connection:**

Instance. The application creates a default instance attachment if one is not present.

**Command Syntax:**

```
►►──db2mtrk──┬───┬──┬───┬──┬───┬──┬───┬──┬──────────────────────────┬──┬───┬──────►
             └─i─┘  └─d─┘  └─p─┘  ├─m─┤  └─r──interval──┬───────┬──┘  └─v─┘
                                  └─w─┘                 └─count─┘

►──┬───┬──────────────────────────────────────────────────────────────────────►◄
   └─h─┘
```

**Command Parameters:**

**-i** On UNIX platforms, show instance level memory. On Windows platforms, show instance and database level memory.

**-d** Show database level memory.

Not available on Windows.

**-p** Show private memory.

**-m** Show maximum values for each pool.

**-w** Show high watermark values for each pool.

**-r** Repeat mode

*interval*
Number of seconds to wait between subsequent calls to the memory tracker (in repeat mode).

*count* Number of times to repeat.

**-v** Verbose output.

**-h** Show help screen. If you specify *-h*, only the help screen appears. No other information is displayed.

**Examples:**

The following call returns database and instance normal values and repeats every 10 seconds:

```
db2mtrk -i -d -v -r 10
```

Consider the following output samples:

The command `db2mtrk -i -d -p` displays the following output:

```
Tracking Memory on: 2002/02/25 at 02:14:10

Memory for instance
   monh      other
   168       3.1M

Memory for database: EKWAN
   utilh     pckcacheh catcacheh lockh     dbh       other
   56        588.8K    470.2K    432.8K    1.8M      5.1M

Memory for database: AJSTORM
   utilh     pckcacheh catcacheh lockh     dbh       other
   56        55.6K     38.3K     432.8K    1.7M      5.1M

Memory for agent 154374
   apph      appctlh   stmth
   357.1K    37.2K     209.5K

Memory for agent 213930
   apph      appctlh
   26.3K     4.0K
```

The command `db2mtrk -i -d -p -v` displays the following output:

```
Tracking Memory on: 2002/02/25 at 17:19:12

Memory for instance
   Database Monitor Heap is of size 168 bytes
   Other Memory is of size 3275619 bytes
   Total: 3275787 bytes

Memory for database: EKWAN
   Backup/Restore/Util Heap is of size 56 bytes
   Package Cache is of size 56888 bytes
   Catalog Cache Heap is of size 39184 bytes
   Lock Manager Heap is of size 443200 bytes
   Database Heap is of size 1749734 bytes
   Other Memory is of size 5349197 bytes
   Total: 7638259 bytes

Memory for database: AJSTORM
   Backup/Restore/Util Heap is of size 56 bytes
   Package Cache is of size 56888 bytes
   Catalog Cache Heap is of size 39184 bytes
   Lock Manager Heap is of size 443200 bytes
   Database Heap is of size 1749734 bytes
   Other Memory is of size 5349197 bytes
   Total: 7638259 bytes
```

```
Memory for agent 154374
   Application Heap is of size 26888 bytes
   Application Control Heap is of size 4107 bytes
   Total: 30995 bytes

Memory for agent 213930
   Application Heap is of size 26888 bytes
   Application Control Heap is of size 4107 bytes
   Total: 30995 bytes
```

**Usage Notes:**

**Notes:**

1. When no flags are specified, usage is returned.
2. On Windows platforms, the -h, -i, or -p flag must be specified. On UNIX-based platforms, the -d, -h, -i, or -p flag must be specified.
3. When the -p flag is specified, detailed private memory usage information is returned ordered by agent ID.
4. The ″Other Memory″ reported is the memory associated with the overhead of operating the database management system.
5. In some cases (such as the package cache) the maximum size displayed will be larger than the value assigned to the configuration parameter. In such cases, the value assigned to the configuration parameter is used as a ’soft limit’, and the pool’s actual memory usage might grow beyond the configured size.

## db2nchg - Change Database Partition Server Configuration

Modifies database partition server configuration. This includes moving the database partition server (node) from one machine to another; changing the TCP/IP host name of the machine; and selecting a different logical port number or a different network name for the database partition server (node). This command can only be used if the database partition server is stopped.

This command is available on Windows NT-based operating systems only.

**Authorization:**

Local Administrator

**Command syntax:**

```
►►──db2nchg──/n:──dbpartitionnum──┬──────────────────────┬──────────────────────────►
                                  └─/i:──instance_name───┘

►──┬──────────────────────┬──┬─────────────────────┬──┬──────────────────┬───────────►
   └─/u:──user,password───┘  └─/p:──logical_port───┘  └─/h:──hostname────┘

►──┬──────────────────────┬──┬─────────────────────┬─────────────────────────────────►◄
   └─/m:──machine_name────┘  └─/g:──network_name───┘
```

**Command parameters:**

**/n:dbpartitionnum**
: Specifies the database partition number of the database partition server's configuration that is to be changed.

**/i:instance_name**
: Specifies the instance in which this database partition server participates. If a parameter is not specified, the default is the current instance.

**/u:username,password**
: Specifies the user name and password. If a parameter is not specified, the existing user name and password will apply.

**/p:logical_port**
: Specifies the logical port for the database partition server. This parameter must be specified to move the database partition server to a different machine. If a parameter is not specified, the logical port number will remain unchanged.

**/h:host_name**
: Specifies TCP/IP host name used by FCM for internal communications. If this parameter is not specified, the host name will remain the same.

**/m:machine_name**
: Specifies the machine where the database partition server will reside. The database partition server can only be moved if there are no existing databases in the instance.

**/g:network_name**
: Changes the network name for the database partition server. This parameter can be used to apply a specific IP address to the database partition server when there are multiple IP addresses on a machine. The network name or the IP address can be entered.

**Examples:**

To change the logical port assigned to database partition 2, which participates in the instance TESTMPP, to logical port 3, enter the following command:

```
db2nchg /n:2 /i:TESTMPP /p:3
```

**Related reference:**

- "db2ncrt - Add Database Partition Server to an Instance" on page 152
- "db2ndrop - Drop Database Partition Server from an Instance" on page 154

# db2ncrt - Add Database Partition Server to an Instance

Adds a database partition server (node) to an instance.

This command is available on Windows operating systems only.

**Scope:**

If a database partition server is added to a computer where an instance already exists, a database partition server is added as a logical database partition server to the computer. If a database partition server is added to a computer where an instance does not exist, the instance is added and the computer becomes a new physical database partition server. This command should not be used if there are databases in an instance. Instead, the START DATABASE MANAGER command should be issued with the ADD DBPARTITIONNUM option. This ensures that the database is correctly added to the new database partition server. It is also possible to add a database partition server to an instance in which a database has been created.

**Note:** The db2nodes.cfg file should not be edited since changing the file might cause inconsistencies in the partitioned database system.

**Authorization:**

Local Administrator authority on the computer where the new database partition server is added.

**Command syntax:**

```
►►──db2ncrt──/n:──dbpartitionnum──/u:──username,password──────────────────────►

►─┬─────────────────────┬─┬────────────────────┬─┬────────────────────┬────────►
  └─/i:──instance_name──┘ └─/m:──machine_name──┘ └─/p:──logical_port──┘

►─┬──────────────────┬─┬─────────────────────┬─┬──────────────────────────────┬──►◄
  └─/h:──host_name──┘ └─/g:──network_name──┘ └─/o:──instance_owning_machine──┘
```

**Command parameters:**

**/n:dbpartitionnum**
: A unique database partition number which identifies the database partition server. The number entered can range from 1 to 999.

**/u:domain_name\username,password**
: Specifies the domain, logon account name and password for DB2.

**/i:instance_name**
: Specifies the instance name. If a parameter is not specified, the default is the current instance.

**/m:machine_name**
: Specifies the computer name of the Windows workstation on which the database partition server resides. This parameter is required if a database partition server is added on a remote computer.

**/p:logical_port**
: Specifies the logical port number used for the database partition server. If this parameter is not specified, the logical port number assigned will be 0.

**Note:** When creating a logical database partition server, this parameter must be specified and a logical port number that is not in use must be selected. Note the following restrictions:

- Every computer must have a database partition server that has a logical port 0.
- The port number cannot exceed the port range reserved for FCM communications in the x:\winnt\system32\drivers\etc\ directory. For example, if a range of 4 ports is reserved for the current instance, then the maximum port number is 3. Port 0 is used for the default logical database partition server.

**/h:host_name**

Specifies the TCP/IP host name that is used by FCM for internal communications. This parameter is required when the database partition server is being added on a remote computer.

**/g:network_name**

Specifies the network name for the database partition server. If a parameter is not specified, the first IP address detected on the system will be used. This parameter can be used to apply a specific IP address to the database partition server when there are multiple IP addresses on a computer. The network name or the IP address can be entered.

**/o:instance_owning_machine**

Specifies the computer name of the instance-owning computer. The default is the local computer. This parameter is required when the **db2ncrt** command is invoked on any computer that is not the instance-owning computer.

**Examples:**

To add a new database partition server to the instance TESTMPP on the instance-owning computer SHAYER, where the new database partition server is known as database partition 2 and uses logical port 1, enter the following command:

```
db2ncrt /n:2 /u:QBPAULZ\paulz,g1reeky /i:TESTMPP /m:TEST /p:1 /o:SHAYER
```

**Related reference:**

- "db2nchg - Change Database Partition Server Configuration" on page 150
- "db2ndrop - Drop Database Partition Server from an Instance" on page 154

## db2ndrop - Drop Database Partition Server from an Instance

Drops a database partition server (node) from an instance that has no databases. If a database partition server is dropped, its database partition number can be reused for a new database partition server. This command can only be used if the database partition server is stopped.

This command is available on Windows NT-based operating systems only.

**Authorization:**

Local Administrator authority on the machine where the database partition server is being dropped.

**Command syntax:**

```
►►──db2ndrop──/n:──dbpartitionnum──┬─────────────────────┬──────────────────►◄
                                   └─/i:──instance_name──┘
```

**Command parameters:**

**/n:dbpartitionnum**
A unique database partition number which identifies the database partition server.

**/i:instance_name**
Specifies the instance name. If a parameter is not specified, the default is the current instance.

**Examples:**

```
db2ndrop /n:2 /i=KMASCI
```

**Usage notes:**

If the instance-owning database partition server (dbpartitionnum 0) is dropped from the instance, the instance becomes unusable. To drop the instance, use the **db2idrop** command.

This command should not be used if there are databases in this instance. Instead, the `db2stop drop nodenum` command should be used. This ensures that the database partition server is correctly removed from the partition database system. It is also possible to drop a database partition server in an instance where a database exists.

**Note:** The db2nodes.cfg file should not be edited since changing the file might cause inconsistencies in the partitioned database system.

To drop a database partition server that is assigned to the logical port 0 from a machine that is running multiple logical database partition servers, all other database partition servers assigned to the other logical ports must be dropped first. Each database partition server must have a database partition server assigned to logical port 0.

**Related reference:**

- "db2nchg - Change Database Partition Server Configuration" on page 150
- "db2ncrt - Add Database Partition Server to an Instance" on page 152

## db2osconf - Utility for Kernel Parameter Values

Makes recommendations for kernel parameter values based on the size of a system. The recommended values are high enough for a given system that they can accommodate most reasonable workloads. This command is currently available only for DB2 on HP-UX on 64-bit instances and the Solaris Operating Environment.

**Authorization:**

- On DB2 for HP-UX, no authorization is required. To make the changes recommended by the **db2osconf** utility, you must have root access.
- On DB2 for the Solaris Operating Environment, you must have root access or be a member of the sys group.

**Command syntax:**

To get the list of currently supported options, enter `db2osconf -h`:

```
   db2osconf -h
Usage:
 -c                 # Client only
 -f                 # Compare to current
 -h                 # Help screen
 -l                 # List current
 -m <mem in GB>     # Specify memory in GB
 -n <num CPUs>      # Specify number of CPUs
 -p <perf level>    # Msg Q performance level (0-3)
 -s <scale factor>  # Scale factor (1-3)
 -t <threads>       # Number of threads
```

**Command parameters:**

**-c**
The ’-c’ switch is for client only installations. This option is available only on DB2 for the Solaris Operating Environment.

**-f**
The ’-f’ switch can be used to compare the current kernel parameters with the values that would be recommended by the **db2osconf** utility. The -f option is the default if no other options are entered with the **db2osconf** command. On the Solaris Operating Environment, only the kernel parameters that differ will be displayed. Since the current kernel parameters are taken directly from the live kernel, they might not match those in /etc/system, the Solaris system specification file. If the kernel parameters from the live kernel are different than those listed in the /etc/system, the /etc/system file might have been changed without a reboot or there might be a syntax error in the file. On HP-UX, the -f option returns a list of recommended parameters and a list of recommended changes to parameter values:

```
****** Please Change the Following in the Given Order ******

WARNING [<parameter name>] should be set to <value>
```

**-l**
The ’-l’ switch lists the current kernel parameters.

**-m**
The ’-m’ switch overrides the amount of physical memory in GB. Normally, the db2osconf utility determines the amount of physical memory automatically. This option is available only on DB2 for the Solaris Operating Environment.

**-n**
The ’-n’ switch overrides the number of CPUs on the system. Normally, the db2osconf utility determines the number of CPUs automatically. This option is available only on DB2 for the Solaris Operating Environment.

**-p** The ′-p′ switch sets the performance level for SYSV message queues. 0 (zero) is the default and 3 is the highest setting. Setting this value higher can increase the performance of the message queue facility at the expense of using more memory.

**-s** The ′-s′ switch sets the scale factor. The default scale factor is 1 and should be sufficient for almost any workload. If a scale factor of 1 is not enough, the system might be too small to handle the workload. The scale factor sets the kernel parameters recommendations to that of a system proportionally larger then the size of the current system. For example, a scale factor of 2.5 would recommend kernel parameters for a system that is 2.5 times the size of the current system.

**-t** The ′-t′ switch provides recommendations for `semsys:seminfo_semume` and `shmsys:shminfo_shmseg` kernel parameter values. This option is available only on DB2 for the Solaris Operating Environment. For multi-threaded programs with a fair number of connections, these kernel parameters might have to be set beyond their default values. They only need to be reset if the multi-threaded program requiring them is a local application:

**semsys:seminfo_semume**
Limit of semaphore undo structures that can be used by any one process

**shmsys:shminfo_shmseg**
Limit on the number of shared memory segments that any one process can create.

These parameters are set in the `/etc/system` file. The following is a guide to set the values, and is what the **db2osconf** utility uses to recommend them. For each local connection DB2 will use one semaphore and one shared memory segment to communicate. If the multi-threaded application is a local application and has X number of connections to DB2, then that application (process) will need X number of shared memory segments and X number of the semaphore undo structures to communicate with DB2. So the value of the two kernel Parameters should be set to X + 10 (the plus 10 provides a safety margin).

Without the ′-l′ or ′-f′ switches, the **db2osconf** utility displays the kernel parameters using the syntax of the /etc/system file. To prevent human errors, the output can be cut and pasted directly into the /etc/system file.

The kernel parameters are recommended based on both the number of CPUs and the amount of physical memory on the system. If one is unproportionately low, the recommendations will be based on the lower of the two.

**Examples:**

Here is a sample output produced by running the **db2osconf** utility with the `-t` switch set for 500 threads.

**Note:** The results received are machine-specific, so the results you receive will vary depending on your environment.

```
db2osconf -t 500

set msgsys:msginfo_msgmax = 65535
set msgsys:msginfo_msgmnb = 65535
set msgsys:msginfo_msgssz = 32
set msgsys:msginfo_msgseg = 32767
set msgsys:msginfo_msgmap = 2562
set msgsys:msginfo_msgmni = 2560
set msgsys:msginfo_msgtql = 2560
set semsys:seminfo_semmap = 3074
set semsys:seminfo_semmni = 3072
set semsys:seminfo_semmns = 6452
set semsys:seminfo_semmnu = 3072
set semsys:seminfo_semume = 600
set shmsys:shminfo_shmmax = 2134020096
set shmsys:shminfo_shmmni = 3072
set shmsys:shminfo_shmseg = 600

Total kernel space for IPC:
0.35MB (shm) + 1.77MB (sem) + 1.34MB (msg) == 3.46MB (total)
```

The recommended values for `set semsys:seminfo_semume` and `set shmsys:shminfo_shmseg` were the additional values provided by running `db2osconf -t 500`.

**Usage notes:**

Even though it is possible to recommend kernel parameters based on a particular DB2 workload, this level of accuracy is not beneficial. If the kernel parameter values are too close to what are actually needed and the workload changes in the future, DB2 might encounter a problem due to a lack of interprocess communication (IPC) resources. A lack of IPC resources can lead to an unplanned outage for DB2 and a reboot would be necessary in order to increase kernel parameters. By setting the kernel parameters reasonably high, it should reduce or eliminate the need to change them in the future. The amount of memory consumed by the kernel parameter recommendations is almost trivial compared to the size of the system. For example, for a system with 4GB of RAM and 4 CPUs, the amount of memory for the recommended kernel parameters is 4.67MB or 0.11%. This small fraction of memory used for the kernel parameters should be acceptable given the benefits.

On the Solaris Operating Environment, there are two versions of the **db2osconf** utility: one for 64-bit kernels and one for 32-bit kernels. The utility needs to be run as root or with the group sys since it accesses the following special devices (accesses are read-only):

```
crw-r-----   1 root     sys       13,  1 Jul 19 18:06 /dev/kmem
crw-rw-rw-   1 root     sys       72,  0 Feb 19  1999 /dev/ksyms
crw-r-----   1 root     sys       13,  0 Feb 19  1999 /dev/mem
```

# db2pd - Monitor and Troubleshoot DB2

The db2pd utility retrieves information from the DB2 memory sets.

**Authorization:**

One of the following:

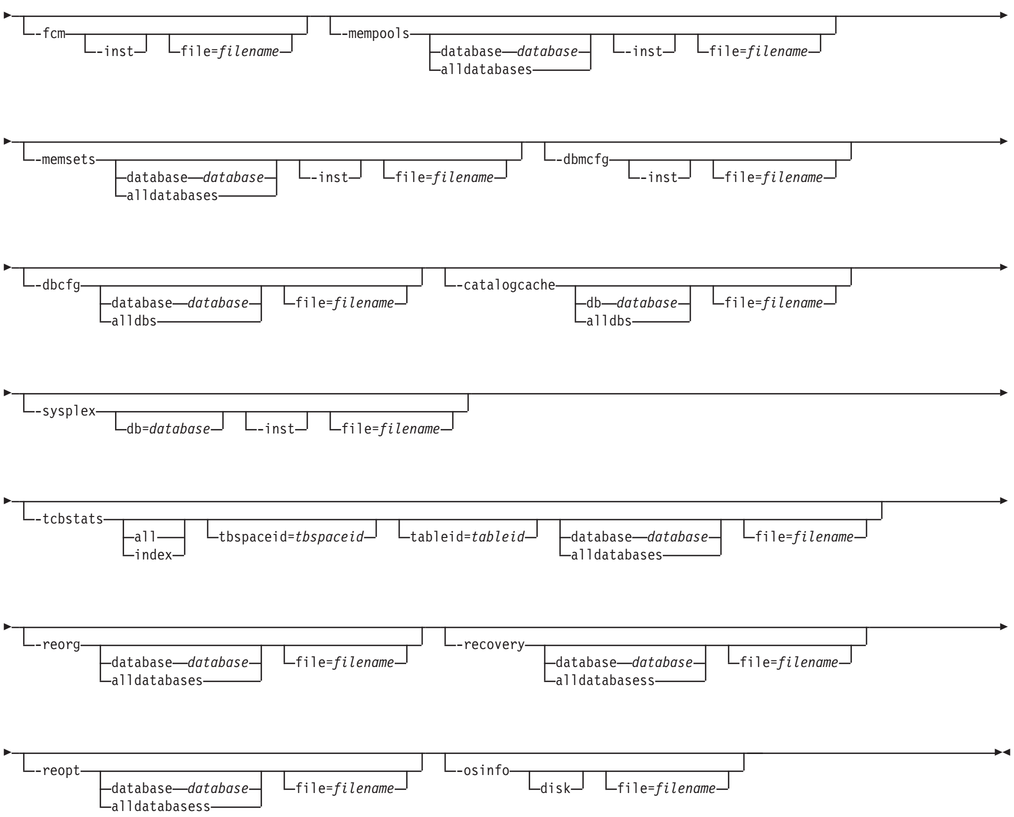
- On Windows-based platforms, the *sysadm* authority level.
- On UNIX-based platforms, the *sysadm* authority level. You must also be the instance owner.

**Required connection:**

None. However, if a database scope option is specified, that database must be active before the command can return the requested information.

**Command syntax:**

```
                                             ┌─,──────────────────────┐
►►──db2pd──┬───────┬──┬───────┬──┬──────────┬──▼─┬──────────────────────┬─┴──┬────────────────────┬──────►
           └─-inst─┘  └─-help─┘  └─-version─┘    └─-dbpartitionnum──num─┘    └─-alldbpartitionnums─┘

   ┌─,───────────────────────┐
►──▼─┬───────────────────────┬┴──┬───────────────┬──┬─────────────────┬──┬─────────────┬──┬───────────────────┬──►
     └─-database──database───┘   └─-alldatabases─┘  └─-file──filename─┘  └─-everything─┘  └─-command──filename─┘

►──┬──────────────┬──┬───────┬──┬────────────────────────────┬────────────────────────────────────────►
   └─-interactive─┘  └─-full─┘  └─-repeat──num sec──count────┘

►──┬────────────────────────────────────────────────────────────────────────────────┬─────────────────►
   └─-applications──┬──────────────────────┬──┬─────────────────────┬──┬───────────────┬┘
                    ├─application=appid────┤  ├─database──database──┤  └─file=filename─┘
                    └─agent=agentid────────┘  └─alldatabases────────┘

►──┬──────────────────────────────────────────────────────────────────┬───────────────────────────────►
   └─-agents──┬───────────────────────┬──┬───────┬──┬───────────────┬─┘
              ├─agent=agentid─────────┤  └─-inst─┘  └─file=filename─┘
              └─application=appid─────┘

►──┬────────────────────────────────────────────────────────────────────────────────────┬─────────────►
   └─-transactions──┬─────────────────────────────┬──┬─────────────────────┬──┬───────────────┬┘
                    ├─transaction=tranhandl───────┤  ├─database──database──┤  └─file=filename─┘
                    └─application=apphdl──────────┘  └─alldatabases────────┘

►──┬───────────────────────────────────────────────────────────────┬──┬──────────────────────────────────────────────────────┬──►
   └─-bufferpools──┬─────────────────────┬──┬───────────────┬──────┘  └─-logs──┬─────────────────────┬──┬───────────────┬──┘
                   ├─database──database──┤  └─file=filename─┘                 ├─database──database──┤  └─file=filename─┘
                   └─alldatabases────────┘                                    └─alldatabases────────┘

►──┬────────────────────────────────────────────────────────────────────────────────────────────────┬──►
   └─-locks──┬──────────────────────┬──┬─────────────────────┬──┬───────────────┬──┬───────────┬────┘
             └─transaction=tranhdl──┘  ├─database──database──┤  └─file=filename─┘  └─showlocks─┘
                                       └─alldatabases────────┘

►──┬──────────────────────────────────────────────────────────────────────────────────────────────────────────┬──►
   └─-tablespaces──┬─────────────────────┬──┬───────────────┬──┬───────┬──┬─────────────────────────────┬─────┘
                   ├─database──database──┤  └─file=filename─┘  └─group─┘  └─tablespace=tablespace_id───┘
                   └─alldatabases────────┘

►──┬──────────────────────────────────────────────────────────┬──┬─────────────────────────────────────────────────────────┬──►
   └─-dynamic──┬─────────────────────┬──┬───────────────┬─────┘  └─-static──┬─────────────────────┬──┬───────────────┬────┘
               ├─database──database──┤  └─file=filename─┘                    ├─database──database──┤  └─file=filename─┘
               └─alldatabases────────┘                                       └─alldatabases────────┘
```



**Command parameters:**

**-inst** Returns all instance-scope information.

**-help** Displays the online help information.

**-version**
Displays the current version and service level of the installed DB2 product.

**-dbpartitionnum** *num*
Specifies that the command is to run on the specified database partition server.

**alldbpartitionnums**
Specifies that this command should be run on all database partition servers in the instance.

**-database** *database*
Specifies that the command attaches to the database memory sets of the specified database.

**-alldatabases**
Specifies that the command attaches to all memory sets of all the databases.

**-everything**
Runs all options for all databases on all database partition servers that are local to the machine.

**-file** *filename*
Specifies to write the output to the specified file.

**-command** *filename*
Specifies to read and execute the **db2pd** options that are specified in the file.

**-interactive**
Specifies to override the values specified for the DB2PDOPT environment variable when running the **db2pd** command.

**-full** Specifies that all output is expanded to its maximum length. If not specified, output is truncated to save space on the display.

**-repeat** *num sec count*
Specifies that the command is to be repeated after the specified number of seconds. If a value is not specified for the number of seconds, the command repeats every five seconds. You can also specify the number of times the output will be repeated. If you do not specify a value for *count*, the command is repeated until it is interrupted.

**-applications**
Returns information about applications.

If an application ID is specified, information is returned about that application.

If an agent ID is specified, information is returned about the agent that is working on behalf of the application.

**-agents**
Returns information about agents.

If an agent ID is specified, information is returned about the agent.

If an application ID is specified, information is returned about all the agents that are performing work for the application.

Specify this option with the `-inst` option, if you have chosen a database that you want scope output for.

**-transactions**
Returns information about active transactions.

If a transaction handle is specified, information is returned about that transaction handle.

If an application handle is specified, information is returned about the application handle of the transaction.

**-bufferpools**
Returns information about the buffer pools.

**-logs** Returns information about the logs.

**-locks** Returns information about the locks.

Specify a transaction handle to obtain information about the locks that are held by a specific transaction.

Specify this option with the *showlocks* option to return detailed information about lock names.

**-tablespaces**
Returns information about the table spaces.

Specify this option with the `group` option to display the information about the containers of a table space grouped with the table space.

Specify this option with the `tablespace` option to display the information about a specific table space and its containers.

**-dynamic**
Returns information about the execution of dynamic SQL.

**-static** Returns information about the execution of static SQL and packages.

**-fcm** Returns information about the fast communication manager.

Specify this option with the `-inst` option, if you have chosen a database for which you want scope output.

**-mempools**
Returns information about the memory pools.

Specify this option with the -inst option to include all the instance-scope information in the returned information.

**-memsets**
Returns information about the memory sets.

Specify this option with the -inst option to include all the instance-scope information in the returned information.

**-dbmcfg**
Returns the settings of the database manager configuration parameters.

Specify this option with the `-inst` option, if you have chosen a database for which you want scope output.

**-dbcfg** Returns the settings of the database configuration parameters.

**-catalogcache**
Returns information about the catalog cache.

**-sysplex**
Returns information about the list of servers associated with the database alias indicated by the db parameter. If the -database parameter is not specified, information is returned for all databases.

Specify this option with the `-inst` option, if you have chosen a database for which you want scope output.

**-tcbstats**
Returns information about tables and indexes.

**-reorg** Returns information about table reorganization.

**-recovery**
Returns information about recovery activity.

**-reopt** Returns information about cached SQL statements that were reoptimized using the REOPT ONCE option.

**-osinfo**
Returns operating system information. If a disk path is specified, information about the disk will be printed.

**Examples:**

The following example shows how to invoke the **db2pd** command from the command line to obtain information about agents that are servicing client requests:

```
db2pd -agents
```

The following example shows how to invoke the **db2pd** command from the command line to obtain information about agents that are servicing client requests. In this example, the DB2PDOPT environment variable is set with the -agents parameter before invoking the **db2pd** command. The command uses the information set in the environment variable when it executes.

```
export DB2PDOPT="-agents"
db2pd
```

The following example shows how to invoke the **db2pd** command from the command line to obtain information about agents that are servicing client requests. In this example, the -agents parameter is set in the file `file.out` before invoking the **db2pd** command. The **-command** parameter causes the command to use the information in the `file.out` file when it executes.

```
echo "-agents" > file.out
db2pd -command file.out
```

The following example shows how to invoke the **db2pd** command from the command line to obtain all database and instance-scope information:

```
db2pd -inst -alldbs
```

**Usage notes:**

The sections that follow describe the output produced by the different **db2pd** parameters.
- “-applications”
- “-agents” on page 163
- “-transactions” on page 163
- “-bufferpools” on page 164
- “-logs” on page 165
- “-locks” on page 165
- “-tablespaces” on page 167
- “-dynamic” on page 169
- “-static” on page 170
- “-fcm” on page 171
- “-mempools” on page 172
- “-memsets” on page 173
- “-dbmcfg” on page 173
- “-dbcfg” on page 173
- “-catalogcache” on page 173
- “-sysplex” on page 176
- “-tcbstats” on page 177
- “-reorg” on page 178
- “-recovery” on page 180
- “-reopt” on page 181
- “-osinfo” on page 181

**-applications parameter:**

For the -applications parameter, the following information is returned:

**ApplHandl**
The application handle, including the node and the index.

**NumAgents**
The number of agents that are working on behalf of the application.

**CoorPid**
The process ID of the coordinator agent for the application.

**Status** The status of the application.

**Appid** The application ID.

**-agents parameter:**

For the -agents parameter, the following information is returned:

**AppHandl**
The application handle, including the node and the index.

**AgentPid**
The process ID of the agent process.

**Priority**
The priority of the agent.

**Type** The type of agent.

**State** The state of the agent.

**ClientPid**
The process ID of the client process.

**Userid**
The user ID running the agent.

**ClientNm**
The name of the client process.

**Rowsread**
The number of rows that were read by the agent.

**Rowswrtn**
The number of rows that were written by the agent.

**LkTmOt**
The lock timeout setting for the agent.

**-transactions parameter:**

For the -transactions parameter, the following information is returned:

**ApplHandl**
The application handle of the transaction.

**TranHdl**
The transaction handle of the transaction.

**Locks** The number of locks held by the transaction.

**State** The transaction state.

**Tflag** The transaction flag. The possible values are:

- `0x00000002`. This value is only written to the coordinator node of a two-phase commit application, and indicates that all subordinate nodes have sent a ″prepare to commit″ request.
- `0x00000020`. The transaction must change a capture source table (used for data replication only).
- `0x00000040`. Crash recovery considers the transaction to be in the prepare state.

- 0x00010000. This value is only written to the coordinator partition in a partitioned database environment, and indicates that the coordinator partition has not received a commit request from all subordinate partitions in a two-phase commit transaction.
- 0x00040000. Rollback of the transaction is pending.
- 0x01000000. The transaction resulted in an update on a database partition server that is not the coordinator partition.
- 0x04000000. Loosely coupled XA transactions are supported.
- 0x08000000. Multiple branches are associated with this transaction, and are using the loosely coupled XA protocol.
- 0x10000000. A data definition language (DDL) statement has been issued, indicating that the loosely coupled XA protocol cannot be used by the branches participating in the transaction.

**Tflag2** Transaction flag 2. The possible values are:

- 0x00000004. The transaction has exceeded the limit specified by the *num_log_span* database configuration parameter.
- 0x00000008. The transaction resulted because of the execution of a DB2 utility.
- 0x00000020. The transaction will cede its locks an application with a higher priority (this value ordinarily occurs for jobs that DB2 automatically starts for self tuning and self management).
- 0x00000040. The transaction will not cede its row-level locks to an application with a higher priority (this value ordinarily occurs for jobs that DB2 automatically starts for self tuning and self management)

**Firstlsn**
First LSN of the transaction.

**Lastlsn**
Last LSN of the transaction.

**LogSpace**
The amount of log space that is reserved for the transaction.

**SpaceReserved**
The total log space that is reserved for the transaction, including the used space and all compensation records.

**TID** Transaction ID.

**AxRegCnt**
The number of applications that are registered for a global transaction. For local transactions, the value is 1.

**GXID** Global transaction ID. For local transactions, the value is 0.

**-bufferpools parameter:**

For the -bufferpools parameter, the following information is returned:

**First Active Pool ID**
The ID of the first active buffer pool.

**Max Bufferpool ID**
The maximum ID of all active buffer pools.

**Max Bufferpool ID on Disk**
The maximum ID of all buffer pools defined on disk.

**Num Bufferpools**
The number of available buffer pools.

**ID** The ID of the buffer pool.

**Name** The name of the buffer pool.

**PageSz**
The size of the buffer pool pages.

**PA-NumPgs**
The number of pages in the page area of the buffer pool.

**BA-NumPgs**
The number of pages in the block area of the buffer pool. This value is `0` if the buffer pool is not enabled for block-based I/O.

**BlkSize**
The block size of a block in the block area of the buffer pool. This value is `0` if the buffer pool is not enabled for block-based I/O.

**ES** `Y` or `N` to indicate whether extended storage is enabled for the buffer pool.

**NumTbsp**
The number of table spaces that are using the buffer pool.

**PgsLeft**
The number of pages left to remove in the buffer pool if its size is being decreased.

**CurrentSz**
The current size of the buffer pool in pages.

**PostAlter**
The size of the buffer pool in pages when the buffer pool is restarted.

**SuspndTSCt**
The number of table spaces mapped to the buffer pool that are currently I/O suspended. If `0` is returned for all buffer pools, the database I/O is not suspended.

**-logs parameter:**

For the -logs parameter, the following information is returned:

**Current Log Number**
The number of the current active log.

**Pages Written**
The current page being written in the current log.

**StartLSN**
The starting log sequence number.

**State** `0x00000020` indicates that the log has been archived.

**Size** The size of the log's extent, in pages.

**Pages** The number of pages in the log.

**Filename**
The filename of the log.

**-locks parameter:**

For the -locks parameter, the following information is returned:

**TranHdl**
The transaction handle that is requesting the lock.

**Lockname**
The name of the lock.

**Type** The type of lock. The possible values are:
- `Row`
- `Pool`
- `Table`
- `AlterTab`
- `ObjectTab`
- `OnlBackup`
- `DMS Seq`
- `Internal P`
- `Internal V`
- `Key Value`
- `No Lock`
- `Block Lock`
- `LOG Release`
- `LF Release`
- `LFM File`
- `LOB/LF 4K`
- `APM Seq`
- `Tbsp Load`
- `Table Part`
- `DJ UserMap`
- `DF NickNm`
- `CatCache`
- `OnlReorg`
- `Buf Pool`

**Mode** The lock mode. The possible values are:
- `no lock`
- `IS`
- `IX`
- `S`
- `SIX`
- `X`
- `IN`
- `Z`
- `U`
- `NS`
- `NX`
- `W`
- `NW`

**Sts** The lock status. The possible values are:
- `G` (granted)
- `C` (converting)
- `W` (waiting)

**Owner**
The transaction handle that owns the lock.

**Dur** The duration of the lock.

**HldCnt**
The number of locks currently held.

**Att** The attributes of the lock.

**Rlse** The lock release flags.

**-tablespaces parameter:**

For the -tablespaces parameter, the following information is returned:

**Id** The table space ID.

**Type** The type of table space. The possible values are:
- SMS
- DMS

**Content**
The type of content. The possible values are:
- Any
- Long
- SysTmp
- UsrTmp

**PageSize**
The page size used for the table space.

**ExtentSize**
The size of an extent in pages.

**Prefetch**
The number of pages read from the table space for each range prefetch request.

**BufID**
The ID of the buffer pool that this table space is mapped to.

**BufIDDisk**
The ID of the buffer pool that this table space will be mapped to at next startup.

**State**
- 0x0000000 - NORMAL
- 0x0000001 - QUIESCED: SHARE
- 0x0000002 - QUIESCED: UPDATE
- 0x0000004 - QUIESCED: EXCLUSIVE
- 0x0000008 - LOAD PENDING
- 0x0000010 - DELETE PENDING
- 0x0000020 - BACKUP PENDING
- 0x0000040 - ROLLFORWARD IN PROGRESS
- 0x0000080 - ROLLFORWARD PENDING
- 0x0000100 - RESTORE PENDING
- 0x0000200 - DISABLE PENDING
- 0x0000400 - REORG IN PROGRESS
- 0x0000800 - BACKUP IN PROGRESS
- 0x0001000 - STORAGE MUST BE DEFINED
- 0x0002000 - RESTORE IN PROGRESS
- 0x0004000 - OFFLINE
- 0x0008000 - DROP PENDING
- 0x0010000 - WRITE SUSPENDED
- 0x0020000 - LOAD IN PROGRESS

- 0x0200000 - STORAGE MAY BE DEFINED
- 0x0400000 - STORAGE DEFINITION IS IN FINAL STATE
- 0x0800000 - STORAGE DEFINITION CHANGED PRIOR TO ROLLFORWARD
- 0x1000000 - DMS REBALANCER IS ACTIVE
- 0x2000000 - DELETION IN PROGRESS
- 0x4000000 - CREATION IN PROGRESS

**TotPages**
For DMS table spaces, the sum of the gross size of each of the table space's containers (reported in the total pages field of the container).

For SMS table spaces, this value is always `0`.

**UsablePgs**
For DMS table spaces, the sum of the net size of each of the table space's containers (reported in the usable pages field of the container).

For SMS table spaces, this value is always `0`.

**UsedPgs**
For DMS table spaces, the total number of pages currently in use in the table space.

For SMS table spaces, this value is always `0`.

**PndFreePgs**
The number of pages that are not available for use, but will be if all the currently outstanding transactions commit.

**FreePgs**
For DMS table spaces, the number of pages available for use in the table space.

For SMS table spaces, this value is always `0`.

**HWM** The highest allocated page in the table space.

**MinRecTime**
The minimum recovery time for the table space.

**NQuiescers**
The number of quiescers.

**NumCntrs**
The number of containers owned by a table space.

**MaxStripe**
The maximum stripe set currently defined in the table space (applicable to DMS table spaces only).

**Name** The name of the table space.

The following output describes containers:

**TspId** The ID of the table space that owns the container.

**ContainNum**
The number assigned to the container in the table space.

**Type** The type of container. The possible values are:

- `Path`
- `Disk`
- `File`
- `Striped Disk`

- `Striped File`

**TotalPages**
The number of pages in the container.

**UsablePgs**
The number of usable pages in the container.

**StripeSet**
The stripe set where the container resides (applicable to DMS table spaces only).

**Container**
The name of the container.

**-dynamic parameter:**

For the -dynamic parameter, the following information is returned:

**Dynamic Cache:**

**Current Memory Used**
The number of bytes used by the package cache.

**Total Heap Size**
The number of bytes configured internally for the package cache.

**Cache Overflow flag state**
A flag to indicate whether the package cache is in an overflow state.

**Number of references**
The number of times the dynamic portion of the package cache has been referenced.

**Number of Statement Inserts**
The number of statement inserts into the package cache.

**Number of Statement Deletes**
The number of statement deletions from the package cache.

**Number of Variation Inserts**
The number of variation inserts into the package cache.

**Number of statements**
The number of statements in the package cache.

**Dynamic SQL Statements:**

**AnchID**
The hash anchor identifier.

**StmtID**
The statement identifier.

**NumEnv**
The number of environments that belong to the statement.

**NumVar**
The number of variations that belong to the statement.

**NumRef**
The number of times that the statement has been referenced.

**NumExe**
The number of times that the statement has been executed.

**Text** The text of the SQL statement.

**Dynamic SQL Environments:**

**AnchID**
The hash anchor identifier.

**StmtID**
The statement identifier.

**EnvID** The environment identifier.

**Iso** The isolation level of the environment.

**QOpt** The query optimization level of the environment.

**Blk** The blocking factor of the environment.

**Dynamic SQL Variations:**

**AnchID**
The hash anchor identifier.

**StmtID**
The statement identifier for this variation.

**EnvID** The environment identifier for this variation.

**VarID** The variation identifier.

**NumRef**
The number of times this variation has been referenced.

**Typ** The internal statement type value for the variation section.

**Lockname**
The variation lockname.

**-static parameter:**

For the -static parameter, the following information is returned:

**Static Cache:**

**Current Memory Used**
The number of bytes used by the package cache.

**Total Heap Size**
The number of bytes internally configured for the package cache.

**Cache Overflow flag state**
A flag to indicate whether the package cache is in an overflow state.

**Number of References**
The number of references to packages in the package cache.

**Number of Package Inserts**
The number of package inserts into the package cache.

**Number of Section Inserts**
The number of static section inserts into the package cache.

**Packages:**

**Schema**
The qualifier of the package.

**PkgName**
The name of the package.

**Version**
The version identifier of the package.

**UniqueID**
The consistency token associated with the package.

**NumSec**
The number of sections that have been loaded.

**UseCount**
The usage count of the cached package.

**NumRef**
The number of times the cached package has been referenced.

**Iso** The isolation level of the package.

**QOpt** The query optimization of the package.

**Blk** The blocking factor of the package.

**Lockname**
The lockname of the package.

**Sections:**

**Schema**
The qualifier of the package that the section belongs to.

**PkgName**
The package name that the section belongs to.

**UniqueID**
The consistency token associated with the package that the section belongs to.

**SecNo** The section number.

**NumRef**
The number of times the cached section has been referenced.

**UseCount**
The usage count of the cached section.

**StmtType**
The internal statement type value for the cached section.

**Cursor**
The cursor name (if applicable).

**W-Hld**
Indicates whether the cursor is a WITH HOLD cursor.

**-fcm parameter:**

For the -fcm parameter, the following information is returned:

**Bufs** The current number of free fast communication manager (FCM) buffers.

**BufLWM**
The lowest number of free FCM buffers reached during processing.

**Anchors**
The current number of free message anchors.

**AnchLWM**
The lowest number of free message anchors reached during processing.

**Entries**
The current number of free connection entries.

**EntryLWM**
The lowest number of free connection entries reached during processing.

**RQBs** The current number of free request blocks.

**RQBLWM**
The lowest number of free request blocks reached during processing.

**DBP** The database partition server number.

**TotBufSnt**
The total number of FCM buffers that are sent from the database partition server where **db2pd** is running to the database partition server that is identified in the output.

**TotBufRcv**
The total number of FCM buffers that are received by the database partition server where **db2pd** is running from the database partition server that is identified in the output.

**Status** The connection communication status between the database partition server where **db2pd** and the other database partition servers that are listed in the output. The possible values are:

- `Not Active`
- `Pending`
- `Pending Ack`
- `Active`
- `Congested`
- `Failed`
- `Reconnect`

**-mempools parameter:**

For the -mempools parameter, the following information is returned (All sizes are specified in bytes.):

**MemSet**
The memory set that owns the memory pool.

**PoolName**
The name of the memory pool.

**Id** The memory pool identifier.

**Overhead**
The internal overhead required for the pool structures.

**LogSz** The current total of pool memory requests.

**LogUpBnd**
The current logical size upper bound.

**LogHWM**
The logical size high water mark.

**PhySz** The physical memory required for logical size.

**PhyUpBnd**
The current physical size upper bound.

**PhyHWM**
The largest physical size reached during processing.

**Bnd** The internal bounding strategy.

**BlkCnt**
The current number of allocated blocks in the memory pool.

**CfgParm**
The configuation parameter that declares the size of the pool being reported.

**-memsets parameter:**

For the -memsets parameter, the following information is returned:

**Name** The name of the memory set.

**Address**
The address of the memory set.

**Id** The memory set identifier.

**Size** The size of the memory set in bytes.

**Key** The memory set key (for UNIX-based platforms only).

**DBP** The database partition server that owns the memory set.

**Type** The type of memory set.

**Ov** `Y` or `N` to indicate whether pool overflows are allowed.

**OvSize**
The size of the overflow area for memory pools.

**-dbmcfg parameter:**

For the -dbmcfg parameter, current values of the database manager configuration parameters are returned.

**-dbcfg parameter:**

For the -dbcfg parameter, the current values of the database configuration parameters are returned.

**-catalogcache parameter:**

For the -catalogcache parameter, the following information is returned:

**Catalog Cache:**

**Configured Size**
The number of bytes as specified by the *catalogcache_sz* database configuration parameter.

**Current Size**
The current number of bytes used in the catalog cache.

**Maximum Size**
The maximum amount of memory that is available to the cache (up to the maximum database global memory).

**High Water Mark**
The largest physical size reached during processing.

**SYSTABLES:**

**Schema**
The schema qualifier for the table.

**Name** The name of the table.

**Type** The type of the table.

**TableID**
The table identifier.

**TbspaceID**
The identifier of the table space where the table resides.

**LastRefID**
The last process identifier that referenced the table.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**CatalogCache UsageLock**
The name of the usage lock for the cache entry.

**Sts** The status of the entry. The possible values are:
- V (valid).
- I (invalid).

**SYSRTNS:**

**RoutineID**
The routine identifier.

**Schema**
The schema qualifier of the routine.

**Name** The name of the routine.

**LastRefID**
The last process identifier that referenced the routine.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**CatalogCache UsageLock**
The name of the usage lock for the cache entry.

**Sts** The status of the entry. The possible values are:
- V (valid).
- I (invalid).

**SYSRTNS_PROCSCHEMAS:**

**RtnName**
The name of the routine.

**ParmCount**
The number of parameters in the routine.

**LastRefID**
The last process identifier that referenced the PROCSCHEMAS entry.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**CatalogCache UsageLock**
The name of the usage lock for the cache entry.

**Sts** The status of the entry. The possible values are:
- V (valid).
- I (invalid).

**SYSDATATYPES:**

**TypID** The type identifier.

**LastRefID**
The last process identifier that referenced the type.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**CatalogCache UsageLock**
The name of the usage lock for the cache entry.

**Sts** The status of the entry. The possible values are:
- V (valid).
- I (invalid).

**SYSCODEPROPERTIES:**

**LastRefID**
The last process identifier to reference the SYSCODEPROPERTIES entry.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**CatalogCache UsageLock**
The name of the usage lock for the cache entry.

**Sts** The status of the entry. The possible values are:
- V (valid).
- I (invalid).

**SYSNODEGROUPS:**

**PMapID**
The partitioning map identifier.

**RBalID**
The identifier if the partitioning map that was used for the data redistribution.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**CatalogCache UsageLock**
The name of the usage lock for the cache entry.

**Sts** The status of the entry. The possible values are:
- V (valid).
- I (invalid).

**SYSDBAUTH:**

**AuthID**
The authorization identifier (*authid*).

**AuthType**
The authorization type.

**LastRefID**
The last process identifer to reference the cache entry.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**SYSRTNAUTH:**

**AuthID**
The authorization identifier (*authid*).

**AuthType**
The authorization type.

**Schema**
The schema qualifier of the routine.

**RoutineName**
The name of the routine.

**RtnType**
The type of the routine.

**CatalogCache LoadingLock**
The name of the catalog cache loading lock for the cache entry.

**-sysplex parameter:**

For the -sysplex parameter, the following information is returned:

**Alias** The database alias.

**Location Name**
The unique name of the database server.

**Count** The number of entries found in the list of servers.

**IP Address**
The IP address of the server

**Port** The IP port being used by the server.

**Priority**
The normalized Workload Manager (WLM) weight.

**Connections**
The number of active connections to this server.

**Status** The status of the connection. The possible values are:

- `0`. Healthy.
- `1`. Unhealthy. The server is in the list but a connection cannot be established. This entry currently will not be considered when establishing connections.
- `2`. Unhealthy. The server was previously unavailable, but currently will be considered when establishing connections.

**PRDID**
The product identifier of the server as of the last connection.

**-tcbstats parameter:**

For the -tcbstats parameter, the following information is returned:

**TCB Table Stats:**

**TbspaceID**
The table space identifier.

**TableID**
The table identifier.

**TableName**
The name of the table.

**SchemaNm**
The schema that qualifies the table name.

**Scans** The number of scans that have been performed against the table.

**ObjClass**
The object class. The possible values are:

- `Perm` (permanent).
- `Temp` (temporary).

**UDI** The number of updates, deletes, and inserts that have been performed against the table since the last time that the table statistics were updated via RUNSTATS.

**DataSize**
The number of pages in the data object.

**IndexSize**
The number of pages in use by the table's indexes.

**PgReorgs**
The number of page reorganizations performed.

**NoChgUpdts**
The number of updates that did not change any columns in the table.

**Reads** The number of rows read from the table when the table switch was on for monitoring.

**FscrUpdates**
The number of updates to a free space control record.

**Inserts**
The number of inserts performed on the table.

**Updates**
The number of updates performed on the table.

**Deletes**
The number of deletes performed on the table.

**OvFlReads**
The number of overflows read on the table when the table switch was on for monitoring.

**OvFiCrtes**
The number of new overflows that were created.

**LfSize** The number of pages in the long field object.

**LobSize**
The number of pages in the large object.

**TCB Index Stats:**

**Note:** The following data is only displayed when the -all or -index option is specified with the -tcbstats parameter.

**TbspaceID**
The table space identifier.

**TableID**
The table identifier.

**SchemaNm**
The schema that qualifies the table name.

**ID** The index identifier.

**EmpPgDel**
The number of empty leaf nodes that were deleted.

**RootSplits**
The number of key inserts or updates that caused the index tree depth to increase.

**BndrySplits**
The number of boundary leaf splits (which result in an insert into either the lowest or the highest key).

**PseuEmptPg**
The number of leaf nodes that are marked as being pseudo empty.

**Scans** The number of scans against the index.

**KeyUpdates**
The number of updates to the key.

**InclUpdats**
The number of included column updates.

**NonBndSpts**
The number of non-boundary leaf splits.

**PgAllocs**
The number of allocated pages.

**Merges**
The number merges performed on index pages.

**PseuDels**
The number of keys that are marked as pseudo deleted.

**DelClean**
The number of pseudo deleted keys that have been deleted.

**IntNodSpl**
The number of intermediate level splits.

**-reorg parameter:**

For the -reorg parameter, the following information is returned:

**TabSpaceID**
The table space identifier.

**TableID**
The table identifier.

**TableName**
The name of the table.

**Start** The time that the table reorganization started.

**End** The time that the table reorganization ended.

**PhaseStart**
The start time for a phase of table reorganization.

**MaxPhase**
The maximum number of reorganization phases that will occur during the reorganization. This value only applies to offline table reorganization.

**Phase** The phase of the table reorganization. This value only applies to offline table reorganization. The possible values are:
- `Sort`
- `Build`
- `Replace`
- `InxRecreat`

**CurCount**
A unit of progress that indicates the amount of table reorganization that has been completed. The amount of progress represented by this value is relative to the value of **MaxCount**, which indicates the total amount of work required to reorganize the table.

**MaxCount**
A value that indicates the total amount of work required to reorganize the table. This value can be used in conjunction with **CurCount** to determine the progress of the table reorganization.

**Type** The type of reorganization. The possible values are:
- `Online`
- `Offline`

**Status** The status of an online table reorganization. This value does not apply to offline table reorganizations. The possible values are:
- `Started`
- `Paused`
- `Stopped`
- `Done`
- `Truncat`

**Completion**
The success indicator for the table reorganization. The possible values are:
- `0`. The table reorganization completed successfully.
- `-1`. The table reorganization failed.

**IndexID**
The identifier of the index that is being used to reorganize the table.

**TempSpaceID**
The table space in which the table is being reorganized.

**-recovery parameter:**

For the -recovery parameter, the following information is returned:

**Recovery Status**
The internal recovery status.

**Current Log**
The current log being used by the recovery operation.

**Current LSN**
The current log sequence number.

**Job Type**
The type of recovery being performed. The possible values are:
- `5`. Crash recovery.
- `6`. Rollforward recovery on either the database or a table space.

**Job ID**
The job identifier.

**Job Start Time**
The time the recovery operation started.

**Job Description**
A description of the recovery activity. The possible values are:
- `Tablespace Rollforward Recovery`
- `Database Rollforward Recovery`
- `Crash Recovery`

**Invoker Type**
How the recovery operation was invoked. The possible values are:
- `User`
- `DB2`

**Total Phases**
The number of phases required to complete the recovery operation.

**Current phase**
The current phase of the recovery operation.

**Phase** The number of the current phase.

**Forward phase**
The first phase of rollforward recovery. This phase is also known as the REDO phase.

**Backward phase**
The second phase of rollforward recovery. This phase is also known as the UNDO phase.

**Metric** The units of work. The possible values are:
- `1`. Bytes.
- `2`. Extents.
- `3`. Rows.
- `4`. Pages.
- `5`. Indexes

**TotWkUnits**
The total number of units of work (UOW) to be done for this phase of the recovery operation.

**TotCompUnits**
The total number of UOWs that have been completed.

**-reopt parameter:**

For the -reopt parameter, the following information is returned:

**Dynamic SQL Statements**
See “-dynamic” on page 169.

**Dynamic SQL Environments**
See the “-dynamic” on page 169.

**Dynamic SQL Variations**
See the “-dynamic” on page 169.

**Reopt Values**
Displays information about the variables that were used to reoptimize a given SQL statement. Information is not returned for variables that were not used. Valid values are:

**AnchID**
The hash anchor identifier.

**StmtID**
The statement identifier for this variation.

**EnvID** The environment identifier for this variation.

**VarID** The variation identifier.

**OrderNum**
Ordinal number of the variable that was used to reoptimize of the SQL statement

**SQLZType**
The variable type.

**CodPg**
The variable code page.

**NulID** The flag indicating whether or not the value is null-terminated.

**Len** The length in bytes of the variable value.

**Data** The value used for the variable.

**-osinfo parameter:**

For the -osinfo parameter, the following information is returned:

**CPU information: (On Windows, AIX, HP-UX, Sun and Linux only)**

**TotalCPU**
Total number of CPUs.

**OnlineCPU**
Number of CPUs online.

**ConfigCPU**
Number of CPUs configured.

**Speed(MHz)**
Speed, in MHz, of CPUs.

**HMTDegree**
On systems that support hardware multithreading, this is the number of processors that a physical processor will appear to the operating system as. On nonHMT systems, this value is always 1. On HMT systems, TOTAL reflects the number of logical CPUs. To get the number of physical CPUs, divide the total by THREADING DEGREE.

**Timebase**
Frequency, in Hz, of the timebase register increment. This is supported on Linux PPC only.

**Physical memory and swap in megabytes: (On Windows, AIX, HP-UX, Sun, Linux only)**

**TotalMemTotal**
Size of memory in megabytes.

**FreeMem**
Amount of free in megabytes.

**AvailMem**
Amount of memory available to the product in megabytes.

**TotalSwap**
Total amount of swapspace in megabytes.

**FreeSwap**
Amount of swapspace free in megabytes.

**Virtual memory in megabytes (On Windows, AIX, HP-UX, and Sun only)**

**Total** Total amount of virtual memory on the system in megabytes.

**Reserved**
Amount of reserved virtual memory in megabytes.

**Available**
Amount of virtual memory available in megabytes.

**Free** Amount of virtual memory free in megabytes.

**Operating system information (On Windows, AIX, HP-UX, Sun, and Linux only)**

**OSName**
Name of the operating system software.

**NodeName**
Name of the system.

**Version**
Version of the operating system.

**Machine**
Machine hardware identification.

**Message queue information (On AIX, HP-UX, and Linux only)**

**MsgSeg**
System-wide total of SysV msg segments.

**MsgMax**
System-wide maximum size of a message.

**MsgMap**
System-wide number of entries in message map.

**MsgMni**
System-wide number of msg queue identifiers for system.

**MsgTql**
System-wide number of message headers.

**MsgMnb**
Maximum number of bytes on a message queue.

**MsgSsz**
Message segment size.

**Shared memory information (On AIX, HP-UX, and Linux only)**

**ShmMax**
System-wide maximum size of a shared memory segment in bytes.

**ShmMin**
System-wide minimum size of a shared memory segment in bytes.

**ShmIds**
System-wide number of shared memory identifiers.

**ShmSeg**
Process-wide maximum number of shared memory segments per process.

**Semaphore information: (On AIX, HP-UX, and Linux only)**

**SemMap**
System-wide number of entries in semaphore map.

**SemMni**
System-wide maximum number of a semaphore identifiers.

**SemMns**
System-wide maximum number of semaphores on system.

**SemMnu**
System-wide maximum number of undo structures on system.

**SemMsl**
System-wide maximum number of semaphores per ID.

**SemOpm**
System-wide maximum number of operations per semop call.

**SemUme**
Process-wide maximum number of undo structures per process.

**SemUsz**
System-wide size of undo structure. Derived from semume.

**SemVmx**
System-wide maximum value of a semaphore.

**SemAem**
System-wide maximum adjust on exit value.

**CPU load information (On Windows, AIX, HP-UX, Sun, and Linux only)**

**Short** Shortest duration period.

**Medium**
Medium duration period.

**Long** Long period duration

**Disk information**

**BkSz(bytes)**
File system block size in bytes.

**Total(bytes)**
Total number of bytes on the device in bytes.

**Free(bytes)**
Number of free bytes on the device in bytes.

**Inodes**
Total number of inodes.

**FSID** File system ID.

**DeviceType**
Device type.

**FSName**
File system name.

**MountPoint**
Mount point of the file system.

**Related tasks:**

- "Identifying the owner of a lock that is being waited on" in the *Troubleshooting Guide*

**Related reference:**

- "SYSCAT.ROUTINES catalog view" in the *SQL Reference, Volume 1*
- "SYSCAT.TABLES catalog view" in the *SQL Reference, Volume 1*
- "GET DATABASE CONFIGURATION" on page 389
- "GET DATABASE MANAGER CONFIGURATION" on page 395

## db2perfc - Reset Database Performance Values

Resets the performance values for one or more databases. It is used with the Performance Monitor on Windows operating systems.

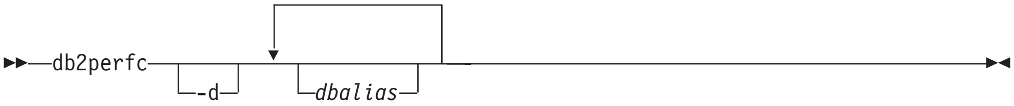
**Authorization:**

Local Windows administrator.

**Required connection:**

None

**Command syntax:**



**Command parameters:**

**-d** Specifies that performance values for DCS databases should be reset.

*dbalias* Specifies the databases for which the performance values should be reset. If no databases are specified, the performance values for all active databases will be reset..

**Usage notes:**

When an application calls the DB2 monitor APIs, the information returned is normally the cumulative values since the DB2 server was started. However, it is often useful to reset performance values, run a test, reset the values again, and then rerun the test.

The program resets the values for all programs currently accessing database performance information for the relevant DB2 server instance (that is, the one held in db2instance in the session in which you run **db2perfc**). Invoking **db2perfc** also resets the values seen by anyone remotely accessing DB2 performance information when the command is executed.

The db2ResetMonitor API allows an application to reset the values it sees locally, not globally, for particular databases.

**Examples:**

The following example resets performance values for all active DB2 databases:

```
db2perfc
```

The following example resets performance values for specific DB2 databases:

```
db2perfc dbalias1 dbalias2
```

The following example resets performance values for all active DB2 DCS databases:

```
db2perfc -d
```

The following example resets performance values for specific DB2 DCS databases:

```
db2perfc -d dbalias1 dbalias2
```

**Related reference:**

- "db2ResetMonitor - Reset Monitor" in the *Administrative API Reference*

# db2perfi - Performance Counters Registration Utility

Adds the DB2 Performance Counters to the Windows operating system. This must be done to make DB2 and DB2 Connect performance information accessible to the Windows Performance Monitor.

**Authorization:**

Local Windows administrator.

**Required connection:**

None

**Command syntax:**

```
►►──db2perfi──┬─-i─┬──────────────────────────────────────────────────────►◄
              └─-u─┘
```

**Command parameters:**

**-i** Registers the DB2 performance counters.

**-u** Deregisters the DB2 performance counters.

**Usage notes:**

The **db2perfi -i** command will do the following:

1. Add the names and descriptions of the DB2 counter objects to the Windows registry.
2. Create a registry key in the Services key in the Windows registry as follows:

```
HKEY_LOCAL_MACHINE
  \System
    \CurrentControlSet
      \Services
        \DB2_NT_Performance
          \Performance
            Library=Name of the DB2 performance support DLL
            Open=Open function name, called when the DLL is
             first loaded
            Collect=Collect function name, called to request
             performance information
            Close=Close function name, called when the DLL is
             unloaded
```

# db2perfr - Performance Monitor Registration Tool

Used with the Performance Monitor on Windows operating systems. The db2perfr command is used to register an administrator user name and password with DB2 with DB2 when accessing the performance counters. This allows a remote Performance Monitor request to correctly identify itself to the DB2 database manager, and be allowed access to the relevant DB2 performance information. You also need to register an administrator user name and password if you want to log counter information into a file using the Performance Logs function.

**Authorization:**

Local Windows administrator.

**Required connection:**

None

**Command syntax:**

```
►►──db2perfr──┬──-r──username──password──┬──────────────────────────────────────►◄
              └──-u──────────────────────┘
```

**Command parameters:**

**-r** Registers the user name and password.

**-u** Deregisters the user name and password.

**Usage notes:**

- Once a user name and password combination has been registered with DB2, even local instances of the Performance Monitor will explicitly log on using that user name and password. This means that if the user name information registered with DB2 does not match, local sessions of the Performance Monitor will not show DB2 performance information.
- The user name and password combination must be maintained to match the user name and password values stored in the Windows NT Security database. If the user name or password is changed in the Windows NT Security database, the user name and password combination used for remote performance monitoring must be reset.
- The default Windows Performance Monitor user name, SYSTEM, is a DB2 reserved word and cannot be used.

## db2rbind - Rebind all Packages

Rebinds packages in a database.

**Authorization:**

One of the following:
- *sysadm*

**Required connection:**

None

**Command syntax:**

```
►►──db2rbind──database──/l logfile──┬─────┬──┬──────────────────────────┬────────────►
                                    └─all─┘  └─/u userid──/p password─┘

       ┌─conservative─┐
►─/r───┴─any──────────┴───────────────────────────────────────────────────────────►◄
```

**Command parameters:**

**database**
Specifies an alias name for the database whose packages are to be revalidated.

**/l**
Specifies the (optional) path and the (mandatory) file name to be used for recording errors that result from the package revalidation procedure.

**all**
Specifies that rebinding of all valid and invalid packages is to be done. If this option is not specified, all packages in the database are examined, but only those packages that are marked as invalid are rebound, so that they are not rebound implicitly during application execution.

**/u**
User ID. This parameter must be specified if a password is specified.

**/p**
Password. This parameter must be specified if a user ID is specified.

**/r**
Resolve. Specifies whether rebinding of the package is to be performed with or without conservative binding semantics. This affects whether new functions and data types are considered during function resolution and type resolution on static DML statements in the package. This option is not supported by DRDA. Valid values are:

**conservative**
Only functions and types in the SQL path that were defined before the last explicit bind time stamp are considered for function and type resolution. Conservative binding semantics are used. This is the default. This option is not supported for an inoperative package.

**any**
Any of the functions and types in the SQL path are considered for function and type resolution. Conservative binding semantics are not used.

**Usage notes:**

- This command uses the rebind API (sqlarbnd) to attempt the revalidation of all packages in a database.
- Use of **db2rbind** is not mandatory.
- For packages that are invalid, you can choose to allow package revalidation to occur implicitly when the package is first used. You can choose to selectively revalidate packages with either the REBIND or the BIND command.
- If the rebind of any of the packages encounters a deadlock or a lock timeout the rebind of all the packages will be rolled back.

**Related reference:**

- "BIND" on page 286
- "PRECOMPILE" on page 560
- "REBIND" on page 595

## db2_recon_aid - RECONCILE Multiple Tables

The db2_recon_aid utility provides an interface to the DB2 RECONCILE utility. The RECONCILE utility operates on one table at a time to validate all DATALINK column references in that table (and ″repair″ them accordingly). There are times when the RECONCILE utility might need to be run against multiple tables. db2_recon_aid is provided for this purpose.

Like the RECONCILE utility, the db2_recon_aid utility must be run on a DB2 server containing tables with DATALINK columns to be reconciled.

**Authorization:**

One of the following:
- sysadm
- sysctrl
- sysmaint
- dbadm

**Required connection:**

None. This command automatically establishes a connection to the specified database.

**Command syntax:**

```
►►──db2_recon_aid──────────────────────────────────────────────────────►


   ┌────────────────────────────────────────────────────┐
   ▼                                                    │
►──┬─-db──database name─────────────────────────────────┴──────────────►◄
   ├──┬─-check─────────────────────┬─────────────────┤
   │  └─-reportdir──report directory─┘                 │
   └─-selective──-server──dlfm server──prefixes──prefix list─┘
```

Where *prefix list* is one or more DLFS prefixes delimited by a colon character, for instance *prefix1:prefix2:prefix3*.

**Command parameters:**

**-db** *database name*
: The name of the database containing the tables with DATALINK columns that need to be reconciled. This parameter is required.

**-check**
: List the tables that might need reconciliation. If you use this parameter, no reconcile operations will be performed. This parameter is required when the -reportdir parameter is not specified.

**-reportdir**
: Specifies the directory where the utility is to place a report for each of the reconcile operations. For each table on which the reconcile is performed, files of the format <tbschema>.<tbname>.<ext> will be created where

- <tbschema> is the schema of the table;

- <tbname> is the table name;
- <ext> is `.ulk` or `.exp`. The `.ulk` file contains a list of files that were unlinked on the Data Links server, and the `.exp` file contains a list of files that were in exception on the Data Links server.

If -check and -reportdir are both specified, -reportdir is ignored.

**-selective**
Process only those tables with DATALINK columns containing file references that match the specified -server and -prefixes criteria.

- If you use this parameter, you must also use both the -server and -prefixes parameters.
- If you do not use this parameter, then all Data Links servers and their prefixes that are registered with the specified DB2 database will either be reconciled, or will be flagged as needing reconciliation.

**-prefixes** *prefix list*
Required when the -selective parameter is used. Specifies the name of one or more Data Links File System (DLFS) prefixes. Prefix values must start with a slash, and must be registered with the specified Data Links file server. Separate multiple prefix names with a colon(:), but do not include any embedded spaces. For example:

```
/dlfsdir1/smith/:/dlfsdir2/smith/
```

The path in a DATALINK column value is considered to match the prefix list if any of the prefixes in the list are a left-most substring of the path.

If this parameter is not used, all prefixdes for all Data Links servers that are registered with the specified DB2 database will be reconciled.

**-server**
The name of the Data Links server for which the reconcile operation is to be performed. The parameter *dlfm server* represents an IP hostname. This hostname must exactly match the DLFM server hostname registered with the given DB2 database.

**Examples:**

```
db2_recon_aid  -db STAFF  -check

db2_recon_aid  -db STAFF  -reportdir /home/smith

db2_recon_aid  -db STAFF  -check  -selective  -server dlmserver.services.com
   -prefixes /dlfsdir1/smith/

db2_recon_aid  -db STAFF  -reportdir /home/smith  -selective -server
   dlmserver.services.com -prefixes /dlfsdir1/smith/:/dlfsdir2/smith/
```

**Usage notes:**

1. On AIX systems or Solaris Operating Environments, the db2_recon_aid utility is located in the INSTHOME/sqllib/adm directory, where INSTHOME is the home directory of the instance owner.
2. On Windows systems, the utility is located in x:\sqllib\bin directory where x: is the drive where you installed DB2 Data Links Manager.
3. db2_recon_aid can identify all tables in a given database which contain DATALINK columns with the FILE LINK CONTROL column attribute. It is these types of columns which might require file reference validation via the RECONCILE utility. By specifying the -check option, the tables of interest can

be simply listed. By specifying the -reportdir option, the RECONCILE utility can actually be automatically run against this set of tables. By specifying the -selective option, you can narrow down the set of tables which db2_recon_aid identifies as candidates for reconciliation (based upon the table’s DATALINK column(s) containing references to a specific Data Links server and one or more of its Data Links File Systems).

4. Depending upon what problem you are trying to solve, you will need to choose between running the RECONCILE or db2_recon_aid utility. The overriding consideration is how many tables might need to be reconciled. For example:
   - If you have an individual table in a state like DRP or DRNP, you might only need to run RECONCILE for that specific table to restore the table to a normal state.
   - If you have had a corruption or loss of a Data Links File System (DLFS) on a given Data Links server, you should use db2_recon_aid (with the -selective option) to locate all tables referencing that Data Links server and that specific ″prefix″ (DLFS path), and perform the reconciliation on each of these tables.
   - If you are simply wanting to validate ALL of your DATALINK file references in your database, you would run db2_recon_aid (without the -selective option).
5. Each prefix must be an absolute path (the path must start with a slash), and must be registered with the given DLFM server.
6. The path in a DATALINK column value is considered to match the prefix list if any of the prefixes in the list are a leftmost substring of the path.

## db2relocatedb - Relocate Database

Renames a database, or relocates a database or part of a database (for example, the container and the log directory) as specified in the configuration file provided by the user. This tool makes the necessary changes to the DB2 instance and database support files.

**Authorization:**

None

**Command syntax:**

```
▶▶──db2relocatedb──-f──configFilename──────────────────────────────────▶◀
```

**Command parameters:**

**-f configFilename**

Specifies the name of the file containing configuration information necessary for relocating the database. This can be a relative or absolute filename. The format of the configuration file is:

```
DB_NAME=oldName,newName
DB_PATH=oldPath,newPath
INSTANCE=oldInst,newInst
NODENUM=nodeNumber
LOG_DIR=oldDirPath,newDirPath
CONT_PATH=oldContPath1,newContPath1
CONT_PATH=oldContPath2,newContPath2
...
```

Where:

**DB_NAME**

Specifies the name of the database being relocated. If the database name is being changed, both the old name and the new name must be specified. This is a required field.

**DB_PATH**

Specifies the path of the database being relocated. This is the path where the database was originally created. If the database path is changing, both the old path and new path must be specified. This is a required field.

**INSTANCE**

Specifies the instance where the database exists. If the database is being moved to a new instance, both the old instance and new instance must be specified. This is a required field.

**NODENUM**

Specifies the node number for the database node being changed. The default is 0.

**LOG_DIR**

Specifies a change in the location of the log path. If the log path is being changed, then both the old path and new path must be specified. This specification is optional if the log path resides under the database path, in which case the path is updated automatically.

**CONT_PATH**

Specifies a change in the location of table space containers. Both the old and new container path must be specified. Multiple CONT_PATH lines can be provided if there are multiple container path changes to be made. This specification is optional if the container paths reside under the database path, in which case the paths are updated automatically. If you are making changes to more than one container where the same old path is being replaced by a common new path, a single `CONT_PATH` entry can be used. In such a case, an asterisk (*) could be used both in the old and new paths as a wildcard.

**Note:** Blank lines or lines beginning with a comment character (#) will be ignored.

**Usage notes:**

If the instance that a database belongs to is changing, the following must be done before running this command to ensure that changes to the instance and database support files will be made:

- If a database is being moved to another instance, create the new instance.
- Copy the files/devices belonging to the databases being copied onto the system where the new instance resides. The path names must be changed as necessary. However, if there are already databases in the directory where the database files are moved to, you can mistakenly overwrite the existing `sqldbdir` file, thereby removing the references to the existing databases. In this scenario, the **db2relocatedb** utility cannot be used. Instead of **db2relocatedb**, an alternative is a redirected restore.
- Change the permission of the files/devices that were copied so that they are owned by the instance owner.

If the instance is changing, the tool must be run by the new instance owner.

In a partitioned database environment, this tool must be run against every partition that requires changes. A separate configuration file must be supplied for each partition, that includes the NODENUM value of the partition being changed. For example, if the name of a database is being changed, every partition will be affected and the **db2relocatedb** command must be run with a separate configuration file on each partition. If containers belonging to a single database partition are being moved, the **db2relocatedb** command only needs to be run once on that partition.

**Examples:**

**Example 1**

To change the name of the database TESTDB to PRODDB in the instance db2inst1 that resides on the path `/home/db2inst1`, create the following configuration file:

```
DB_NAME=TESTDB,PRODDB
DB_PATH=/home/db2inst1
INSTANCE=db2inst1
NODENUM=0
```

Save the configuration file as `relocate.cfg` and use the following command to make the changes to the database files:

```
db2relocatedb -f relocate.cfg
```

**Example 2**

To move the database DATAB1 from the instance jsmith on the path /dbpath to the instance prodinst do the following:

1. Move the files in the directory /dbpath/jsmith to /dbpath/prodinst.
2. Use the following configuration file with the **db2relocatedb** command to make the changes to the database files:

```
DB_NAME=DATAB1
DB_PATH=/dbpath
INSTANCE=jsmith,prodinst
NODENUM=0
```

**Example 3**

The database PRODDB exists in the instance inst1 on the path /databases/PRODDB. The location of two table space containers needs to be changed as follows:

- SMS container /data/SMS1 needs to be moved to /DATA/NewSMS1.
- DMS container /data/DMS1 needs to be moved to /DATA/DMS1.

After the physical directories and files have been moved to the new locations, the following configuration file can be used with the **db2relocatedb** command to make changes to the database files so that they recognize the new locations:

```
DB_NAME=PRODDB
DB_PATH=/databases/PRODDB
INSTANCE=inst1
NODENUM=0
CONT_PATH=/data/SMS1,/DATA/NewSMS1
CONT_PATH=/data/DMS1,/DATA/DMS1
```

**Example 4**

The database TESTDB exists in the instance db2inst1 and was created on the path /databases/TESTDB. Table spaces were then created with the following containers:

```
TS1
TS2_Cont0
TS2_Cont1
/databases/TESTDB/TS3_Cont0
/databases/TESTDB/TS4/Cont0
/Data/TS5_Cont0
/dev/rTS5_Cont1
```

TESTDB is to be moved to a new system. The instance on the new system will be newinst and the location of the database will be /DB2.

When moving the database, all of the files that exist in the /databases/TESTDB/db2inst1 directory must be moved to the /DB2/newinst directory. This means that the first 5 containers will be relocated as part of this move. (The first 3 are relative to the database directory and the next 2 are relative to the database path.) Since these containers are located within the database directory or database path, they do not need to be listed in the configuration file. If the 2 remaining containers are to be moved to different locations on the new system, they must be listed in the configuration file.

After the physical directories and files have been moved to their new locations, the following configuration file can be used with **db2relocatedb** to make changes to the database files so that they recognize the new locations:

```
DB_NAME=TESTDB
DB_PATH=/databases/TESTDB,/DB2
INSTANCE=db2inst1,newinst
NODENUM=0
CONT_PATH=/Data/TS5_Cont0,/DB2/TESTDB/TS5_Cont0
CONT_PATH=/dev/rTS5_Cont1,/dev/rTESTDB_TS5_Cont1
```

**Example 5**

The database TESTDB has two partitions on database partition servers 10 and 20. The instance is servinst and the database path is `/home/servinst` on both database partition servers. The name of the database is being changed to SERVDB and the database path is being changed to `/databases` on both database partition servers. In addition, the log directory is being changed on database partition server 20 from `/testdb_logdir` to `/servdb_logdir`.

Since changes are being made to both database partitions, a configuration file must be created for each database partition and **db2relocatedb** must be run on each database partition server with the corresponding configuration file.

On database partition server 10, the following configuration file will be used:

```
DB_NAME=TESTDB,SERVDB
DB_PATH=/home/servinst,/databases
INSTANCE=servinst
NODE_NUM=10
```

On database partition server 20, the following configuration file will be used:

```
DB_NAME=TESTDB,SERVDB
DB_PATH=/home/servinst,/databases
INSTANCE=servinst
NODE_NUM=20
LOG_DIR=/testdb_logdir,/servdb_logdir
```

**Example 6**

The database MAINDB exists in the instance `maininst` on the path `/home/maininst`. The location of four table space containers needs to be changed as follows:

```
/maininst_files/allcons/C0 needs to be moved to /MAINDB/C0
/maininst_files/allcons/C1 needs to be moved to /MAINDB/C1
/maininst_files/allcons/C2 needs to be moved to /MAINDB/C2
/maininst_files/allcons/C3 needs to be moved to /MAINDB/C3
```

After the physical directories and files are moved to the new locations, the following configuration file can be used with the **db2relocatedb** command to make changes to the database files so that they recognize the new locations.

**Note:** A similar change is being made to all of the containters; that is, `/maininst_files/allcons/` is being replaced by `/MAINDB/` so that a single entry with the wildcard character can be used:

```
DB_NAME=MAINDB
DB_PATH=/home/maininst
INSTANCE=maininst
NODE_NUM=0
CONT_PATH=/maininst_files/allcons/*, /MAINDB/*
```

**Related reference:**

- "db2inidb - Initialize a Mirrored Database" on page 111

## db2rfpen - Reset rollforward pending state

Puts a database in rollforward pending state. If you are using high availability disaster recovery (HADR), the database is reset to a standard database.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──db2rfpen──ON──database_alias──-log──logfile_path──────────────────────────►◄
```

**Command parameters:**

**database_alias**
Specifies the name of the database to be placed in rollforward pending state. If you are using high availability disaster recovery (HADR), the database is reset to a standard database.

**-log logfile_path**
Specifies the log file path.

**Related concepts:**

- "High availability disaster recovery overview" in the *Data Recovery and High Availability Guide and Reference*

# db2rmicons - Remove DB2 icons

Removes DB2 icons and folders from a Linux desktop. This command is only available on Gnome and KDE desktops for supported Intel-based Linux distributions. It is located in the `DB2DIR/bin` directory, where DB2DIR represents `/opt/IBM/db2/V8.1`. It is also located in `/sqllib/bin` in the `home` directory of the instance owner.

**Authorization:**

One of the following:

- To invoke the command for other users: root authority or authority to write to the `home` directories of the specified users
- To invoke the command for your own desktop: none

**Required connection:**

None

**Command syntax:**

```
                   ┌─────────────┐
                   ▼             │
►►──db2rmicons─────user_name─────┴────────────────────────────────────►◄
```

**Command parameters:**

*user_name*
: User ID for which you want to remove desktop icons.

## db2rspgn - Response File Generator (Windows)

The **db2rspgn** command is available only on Windows.

**Command syntax:**

```
                             ┌───────────────────┐
                             ▼                   │
►►──db2rspgn── -d x:\path────┬──────────────┬────┴──┬────────────┬──┬──────────┬──►◄
                             └─ -i instance─┘       └─ -noctlsrv─┘  └─ -nodlfm─┘
```

**Command parameters:**

**-d**
Destination directory for a response file and any instance files. This parameter is required.

**-i**
A list of instances for which you want to create a profile. The default is to generate an instance profile file for all instances. This parameter is optional.

**-noctlsrv**
Indicates that an instance profile file will not be generated for the Control Server instance. This parameter is optional.

**-nodlfm**
Indicates that an instance profile file will not be generated for the Data Links File Manager instance. This parameter is optional.

**Related concepts:**

- "About the response file generator (Windows)" in the *Installation and Configuration Supplement*

**Related tasks:**

- "Response file installation of DB2 overview (Windows)" in the *Installation and Configuration Supplement*

# db2sampl - Create Sample Database

Creates a sample database named SAMPLE.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Command syntax:**

```
►►──db2sampl──┬──────┬──┬────┬──────────────────────────────────────────►◄
              └─path─┘  └─-k─┘
```

**Command parameters:**

**path**
Specifies the path on which to create the SAMPLE database. The path is a single drive letter for Windows.

If a path is not specified, SAMPLE is created on the default database path (the *dftdbpath* parameter in the database manager configuration file). On UNIX based systems, the default is the HOME directory of the instance owner. On Windows operating systems, it is the root directory (where DB2 is installed).

**-k**
Creates primary keys on the following SAMPLE tables:

| Table | Primary Key |
|---|---|
| DEPARTMENT | DEPTNO |
| EMPLOYEE | EMPNO |
| ORG | DEPTNUMB |
| PROJECT | PROJNO |
| STAFF | ID |
| STAFFG | ID (DBCS only) |

**Note:** The path must be specified *before* this option.

**Usage notes:**

This command can only be executed from server nodes. SAMPLE cannot be created on nodes that are database clients only.

The SAMPLE database is created with the instance authentication type that is specified by the database manager configuration parameter *authentication*.

The qualifiers for the tables in SAMPLE are determined by the user ID issuing the command.

If SAMPLE already exists, **db2sampl** creates the tables for the user ID issuing the command, and grants the appropriate privileges.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395

## db2secv82 - Set permissions for DB2 objects

Sets the permissions for DB2 objects (for example, files, directories, network shares, registry keys and services) on updated DB2 Universal Database (UDB) installations.

**Authorization:**

- *sysadm*

**Required connection:**

none

**Command syntax:**

```
►►──db2secv82──┬────────────────┬──┬─────────────────┬──┬────┬──────────────────►◄
               └─/u──usergroup──┘  └─/a──admingroup──┘  └─/r─┘
```

**Command parameters:**

**/u** *usergroup*
: Specifies the name of the user group to be added. If this option is not specified, the default DB2 user group (DB2USERS) is used.

**/a** *admingroup*
: Specifies the name of the administration group to be added. If this option is not specified, the default DB2 administration group (DB2ADMNS) is used.

**/r**
: Specifies that the changes made by previously running db2secv82.exe should be reversed. If you specify this option, all other options are ignored.

**Note:** This option will only work if no other DB2 commands have been issued since the db2secv82.exe command was issued.

## db2set - DB2 Profile Registry

Displays, sets, or removes DB2 profile variables. An external environment registry command that supports local and remote administration, via the DB2 Administration Server, of DB2's environment variables stored in the DB2 profile registry.

**Authorization:**

*sysadm*

**Required connection:**

None

**Command syntax:**

```
►►──db2set──┬──────────────────┬──┬────────────────────────────────────────┬──────────►
            └─variable=──value─┘  ├─-g─────────────────────────────────────┤
                                  ├─-i──instance──┬──────────────────────┬─┤
                                  │               └─db-partition-number──┘ │
                                  └─-gl────────────────────────────────────┘

►──┬────────┬──┬─────────┬──┬──────────────────────────────────────────────┬──────────►
   └─-all───┘  └─-null───┘  └─-r──┬────────────┬──┬──────────────────────┬─┘
                                  └─instance───┘  └─db-partition-number──┘

►──┬──────────────────────────────────────────────┬──┬───────┬──┬──────┬──┬───────┬──►
   └─-n──DAS node──┬───────────────────────────┬──┘  ├─-l────┤  └─-v───┘  ├─-ul───┤
                   └─-u──user──┬────────────────┬┘   └─-lr───┘            └─-ur───┘
                               └─-p──password───┘

►──┬──────┬───────────────────────────────────────────────────────────────────────►◄
   ├─-h───┤
   └─-?───┘
```

**Command parameters:**

**variable= value**
: Sets a specified variable to a specified value. To delete a variable, do not specify a value for the specified variable. Changes to settings take effect after the instance has been restarted.

**-g**
: Accesses the global profile variables.

**-i**
: Specifies the instance profile to use instead of the current, or default.

**db-partition-number**
: Specifies a number listed in the `db2nodes.cfg` file.

**-gl**
: Accesses the global profile variables stored in LDAP. This option is only effective if the registry variable DB2_ENABLE_LDAP has been set to YES.

**-all**
: Displays all occurrences of the local environment variables as defined in:

- The environment, denoted by [e]
- The node level registry, denoted by [n]
- The instance level registry, denoted by [i]
- The global level registry, denoted by [g].

**-null** Sets the value of the variable at the specified registry level to NULL. This avoids having to look up the value in the next registry level, as defined by the search order.

**-r instance**
Resets the profile registry for the given instance. If no instance is specified, and an instance attachment exists, resets the profile for the current instance. If no instance is specified, and no attachment exists, resets the profile for the instance specified by the DB2INSTANCE environment variable.

**-n DAS node**
Specifies the remote DB2 administration server node name.

**-u user**
Specifies the user ID to use for the administration server attachment.

**-p password**
Specifies the password to use for the administration server attachment.

**-l** Lists all instance profiles.

**-lr** Lists all supported registry variables.

**-v** Specifies verbose mode.

**-ul** Accesses the user profile variables.

**Note:** This parameter is supported on Windows operating systems only.

**-ur** Refreshes the user profile variables.

**Note:** This parameter is supported on Windows operating systems only.

**-h/-?** Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Examples:**

- Display all defined profiles (DB2 instances):
```
db2set -l
```
- Display all supported registry variables:
```
db2set -lr
```
- Display all defined global variables:
```
db2set -g
```
- Display all defined variables for the current instance:
```
db2set
```
- Display all defined values for the current instance:
```
db2set -all
```
- Display all defined values for DB2COMM for the current instance:
```
db2set -all DB2COMM
```
- Reset all defined variables for the instance INST on node 3:
```
db2set -r -i INST 3
```
- Unset the variable DB2CHKPTR on the remote instance RMTINST through the DAS node RMTDAS using user ID MYID and password MYPASSWD:
```
db2set -i RMTINST -n RMTDAS -u MYID -p MYPASSWD DB2CHKPTR=
```
- Set the variable DB2COMM to be TCPIP,IPXSPX,NETBIOS globally:
```
db2set -g DB2COMM=TCPIP,IPXSPX,NETBIOS
```

- Set the variable DB2COMM to be only TCPIP for instance MYINST:

  ```
  db2set -i MYINST DB2COMM=TCPIP
  ```

- Set the variable DB2COMM to null at the given instance level:

  ```
  db2set -null DB2COMM
  ```

**Usage notes:**

If no variable name is specified, the values of all defined variables are displayed. If a variable name *is* specified, only the value of that variable is displayed. To display all the defined values of a variable, specify *variable* `-all`. To display all the defined variables in all registries, specify `-all`.

To modify the value of a variable, specify *variable*`=`, followed by its new value. To set the value of a variable to NULL, specify *variable* `-null`.

**Note:** Changes to settings take effect after the instance has been restarted.

To delete a variable, specify *variable*`=`, followed by no value.

## db2setup - Install DB2

Installs DB2 products. This command is only available on UNIX-based systems. The command for Windows operating systems is **setup**.

This utility is located on the DB2 installation media. It launches the DB2 Setup wizard to define the installation and install DB2 products. If invoked with the *-r* option, it performs an installation without further input, taking installation configuration information from a response file.

**Command syntax:**

```
►►──db2setup──┬─────────────────┬──┬─────────────────┬──┬─────────────────────┬──────────────►
              └─-i──language────┘  └─-l──log_file────┘  └─-t──trace_file──────┘

►──┬──────────────────────┬──┬──────┬───────────────────────────────────────────────────────►◄
   └─-r──response_file────┘  ├─-?───┤
                             └─-h───┘
```

**Command parameters:**

**-i** *language*
: Two-letter language code of the language in which to perform the installation.

**-l** *log_file*
: Full path and file name of the log file to use.

**-t** *trace_file*
: Generates a file with install trace information.

**-r** *response_file*
: Full path and file name of the response file to use.

**-?, -h**
: Generates usage information.

**Related reference:**

- "setup - Install DB2" on page 243

## db2sql92 - SQL92 Compliant SQL Statement Processor

Reads SQL statements from either a flat file or standard input, dynamically describes and prepares the statements, and returns an answer set. Supports concurrent connections to multiple databases.

**Authorization:**

*sysadm*

**Required connection:**

None. This command establishes a database connection.

**Command syntax:**

```
►►──db2sql92──┬─────────────┬──┬───────────────┬──┬──────────────────────┬──────────────►
              └─-d──dbname──┘  └─-f──file_name─┘  └─-a──userid/passwd────┘

►──┬──────────────────────────────┬──┬──────────────┬──┬──────────────────────┬─────────►
   └─-r──outfile──┬────────────┬──┘  │      ┌─on──┐ │  │     ┌─short────┐     │
                  └─,outfile2──┘     └─-c───┴─off─┴─┘  └─-i──┼─none─────┤─────┘
                                                             ├─long─────┤
                                                             └─complete─┘

►──┬──────────────┬──┬──────────────┬──┬──────────────┬──┬────┬─────────────────────────►◄
   └─-o──options──┘  │      ┌─off─┐ │  │      ┌─off─┐ │  └─-h─┘
                     └─-v───┴─on──┴─┘  └─-s───┴─on──┴─┘
```

**Command parameters:**

**-d dbname**
: An alias name for the database against which SQL statements are to be applied. The default is the value of the **DB2DBDFT** environment variable.

**-f file_name**
: Name of an input file containing SQL statements. The default is standard input.

Identify comment text with two hyphens at the start of each line, that is, `--` *<comment>*. If it is to be included in the output, mark the comment as follows: `--#COMMENT` *<comment>*.

A *block* is a number of SQL statements that are treated as one, that is, information is collected for all of those statements at once, instead of one at a time. Identify the beginning of a block of queries as follows: `--#BGBLK`. Identify the end of a block of queries as follows: `--#EOBLK`.

Specify one or more control options as follows: `--#SET` *<control option>* *<value>*. Valid control options are:

**ROWS_FETCH**
: Number of rows to be fetched from the answer set. Valid values are `-1` to *n*. The default value is `-1` (all rows are to be fetched).

**ROWS_OUT**
: Number of fetched rows to be sent to output. Valid values are `-1` to *n*. The default value is `-1` (all fetched rows are to be sent to output).

**AUTOCOMMIT**
Specifies autocommit on or off. Valid values are `ON` or `OFF`. The default value is `ON`.

**PAUSE**
Prompts the user to continue.

**TIMESTAMP**
Generates a time stamp.

**-a userid/passwd**
Name and password used to connect to the database.

**-r outfile**
An output file that will contain the query results. An optional *outfile2* will contain a results summary. The default is standard output.

**-c** Automatically commit changes resulting from each SQL statement.

**-i** An elapsed time interval (in seconds).

**none** Specifies that time information is not to be collected.

**short** The run time for a query.

**long** Elapsed time at the start of the next query.

**complete**
The time to prepare, execute, and fetch, expressed separately.

**-o options**
Control options. Valid options are:

**f rows_fetch**
Number of rows to be fetched from the answer set. Valid values are `-1` to *n*. The default value is `-1` (all rows are to be fetched).

**r rows_out**
Number of fetched rows to be sent to output. Valid values are `-1` to *n*. The default value is `-1` (all fetched rows are to be sent to output).

**-v** Verbose. Send information to standard error during query processing. The default value is `off`.

**-s** Summary Table. Provide a summary of elapsed times and CPU times, containing both the arithmetic and the geometric means of all collected values.

**-h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Usage notes:**

The following can be executed from the **db2sql92** command prompt:

- All control options
- SQL statements
- CONNECT statements
- `commit work`
- `help`
- `quit`

This tool supports switching between different databases during a single execution of the program. To do this, issue a CONNECT RESET and then one of the following on the **db2sql92** command prompt (stdin):

```
connect to database
connect to database USER userid USING passwd
```

SQL statements can be up to 65 535 characters in length. Statements must be terminated by a semicolon.

SQL statements are executed with the repeatable read (RR) isolation level.

When running queries, there is no support for the results set to include LOBs.

**Related reference:**

- "db2batch - Benchmark Tool" on page 24

## db2sqljbind - DB2 SQLJ Profile Binder

Binds a previously customized SQLJ profile to a database. By default, four packages are created, one for each isolation level. If the -singlepkgname option is used when customizing, only a single package is created and the ISOLATION option must be used. This utility should be run after the SQLJ application has been customized.

**Authorization:**

One of the following:

- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist, and one of:
  – IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
  – CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

**Required connection:**

This command establishes a database connection.

**Command syntax:**

```
►►──db2sqljbind──┬───────┬──-url jdbc:db2://server:port/dbname──-user──username──────►
                 └─-help─┘

►──-password──password──┬─────────────────────────────────┬──────────────────────────►
                        └─-bindoptions──"bind options"─┘

►──┬────────────────────────────────────┬───────────────────────────────────────────►
   │                     ┌─NO──┐        │
   └─-staticpositioned───┴─YES─┴────────┘

►──┬─────────────────────────────────────────────────────────────┬──────────────────►
   └─-tracefile──name──┬─────────────────────────────────────────┬┘
                       │                ┌─,──────────────────────┐ │
                       │                ▼                        │ │
                       └─-tracelevel────┬─TRACE_ALL──────────────┬┴─┘
                                        ├─TRACE_CONNECTION_CALLS─┤
                                        ├─TRACE_CONNECTS─────────┤
                                        ├─TRACE_DIAGNOSTICS──────┤
                                        ├─TRACE_DRDA_FLOWS───────┤
                                        ├─TRACE_DRIVER_CONFIGURATION─┤
                                        ├─TRACE_NONE─────────────┤
                                        ├─TRACE_PARAMETER_META_DATA──┤
                                        ├─TRACE_RESULT_SET_CALLS─┤
                                        ├─TRACE_RESULT_SET_META_DATA─┤
                                        ├─TRACE_STATEMENT_CALLS──┤
                                        └─TRACE_SQLJ─────────────┘
```

```
►–profilename–––––––––––––––––––––––––––––––––––––––––––––––––––––►◄
```

**Command parameters:**

**-help** Displays help information. All other options are ignored.

**-url jdbc:db2:**//*server*:*port*/*dbname*
: Specifies a JDBC URL for establishing the database connection. The DB2 JDBC type 4 driver is used to establish the connection.

**-user** *username*
: Specifies the name used when connecting to a database.

**-password** *password*
: Specifies the password for the user name.

**-bindoptions** *"bind options"*
: Specifies a list of bind options. The following options are supported. For detailed descriptions, see the BIND command.

- For DB2 for Windows and UNIX:
  - ACTION (but not ACTION RETAIN)
  - BLOCKING
  - COLLECTION
  - DEGREE
  - EXPLAIN
  - EXPLSNAP
  - FEDERATED
  - FUNCPATH
  - INSERT
  - ISOLATION (see the -singlepkgname option of the db2sqljcustomize command)
  - OWNER
  - QUALIFIER
  - QUERYOPT
  - REOPT
  - SQLERROR (but not SQLERROR CHECK)
  - SQLWARN
  - STATICREADONLY
  - VALIDATE
  - VERSION
- For DB2 on servers other than Windows and UNIX:
  - ACTION (but not ACTION RETAIN)
  - BLOCKING
  - COLLECTION
  - DBPROTOCOL
  - DEGREE
  - EXPLAIN
  - IMMEDWRITE
  - ISOLATION (see the -singlepkgname option of the db2sqljcustomize command)
  - OPTHINT

- OWNER
- PATH
- QUALIFIER
- RELEASE
- REOPT
- SQLERROR
- VALIDATE
- VERSION

**-staticpositioned**

Values are YES and NO. The default value is NO. The value provided to db2sqljbind must match the value provided previously to db2sqljcustomize.

**-tracefile** *name*

Enables tracing and identifies the output file for trace information. Should only be used when instructed by an IBM service technician.

**-tracelevel**

Identifies the level of tracing. If -tracelevel is omitted, TRACE_ALL is used.

**profilename**

Specifies the relative or absolute name of an SQLJ profile file. When an SQLJ file is translated into a Java file, information about the SQL operations it contains is stored in SQLJ-generated resource files called profiles. Profiles are identified by the suffix _SJProfileN (where N is an integer) following the name of the original input file. They have a `.ser` extension. Profile names can be specified with or without the `.ser` extension.

Multiple files can be bound together into a single package on DB2. A list of profiles can be provided on the command line call to db2sqljbind. Alternatively, a list of profiles can be listed one per line in a file with a .grp extension, and the .grp file name provided to db2sqljbind. The binder will bind all the statements together. When binding a group of profiles together, the profiles must have been previously customized together using the same list of files in the same order.

**Examples:**

```
db2sqljbind -user richler -password mordecai
  -url jdbc:db2://server:50000/sample -bindoptions "EXPLAIN YES"
  pgmname_SJProfile0.ser
```

**Related reference:**

- “BIND” on page 286
- “db2sqljcustomize - DB2 SQLJ Profile Customizer” on page 213
- “db2sqljprint - DB2 SQLJ Profile Printer” on page 219
- “sqlj - DB2 SQLJ Translator” on page 244

## db2sqljcustomize - DB2 SQLJ Profile Customizer

Processes an SQLJ profile containing embedded SQL statements. By default, four DB2 packages are created in the database: one for each isolation level. This utility augments the profile with DB2-specific information for use at run time, and should be run after the SQLJ application has been translated, but before the application is run.

**Authorization:**

One of the following:

- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist, and one of:
  - IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
  - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

**Required connection:**

This command establishes a database connection if -url is specified.

**Command syntax:**

```
►►──┬─db2sqljcustomize─┬──┬───────┬──────────────────────────────────────────────►
    └─db2profc─────────┘  └─-help─┘


►──┬─-url jdbc:db2://server:port/dbname──-user──username──-password──password─┬──────►
   └─-datasource──jndiName────────────────────────────────────────────────────┘


   ┌─-automaticbind──┬─YES─┬─┐
   │                 └─NO──┘ │
►──┴─────────────────────────┴──┬──────────────────────────────┬─────────────────►
                                └─-bindoptions──"bind options"─┘

                                ┌─-onlinecheck──┬─YES─┬─┐
                                │               └─NO──┘ │
►──┬───────────────────┬────────┴───────────────────────┴──┬──────────────────────────┬──►
   └─-collection──name─┘                                   └─-pkgversion──versionname─┘

                                                    ┌─-staticpositioned──┬─NO──┬─┐
                                                    │                    └─YES─┘ │
►──┬──────────────────┬──┬──────────────────────┬───┴────────────────────────────┴──────►
   └─-qualifier──name─┘  ├─-rootpkgname──name───┤
                         └─-singlepkgname──name─┘
```

```
►─┬───────────────────────────────────────────────────────────┬──────────────►
  └─-tracefile──name──┬─────────────────────────────────────┬─┘
                      │                ┌─,──────────────────┐ │
                      │                ▼                    │ │
                      └─-tracelevel────┬─TRACE_ALL────────────────┬┴─┘
                                       ├─TRACE_CONNECTION_CALLS───┤
                                       ├─TRACE_CONNECTS───────────┤
                                       ├─TRACE_DIAGNOSTICS────────┤
                                       ├─TRACE_DRDA_FLOWS─────────┤
                                       ├─TRACE_DRIVER_CONFIGURATION┤
                                       ├─TRACE_NONE───────────────┤
                                       ├─TRACE_PARAMETER_META_DATA┤
                                       ├─TRACE_RESULT_SET_CALLS───┤
                                       ├─TRACE_RESULT_SET_META_DATA┤
                                       ├─TRACE_SQLJ───────────────┤
                                       └─TRACE_STATEMENT_CALLS────┘

►─profilename────────────────────────────────────────────────────────────────►◄
```

**Command parameters:**

**-help** Displays help information. All other options are ignored.

**-url jdbc:db2:**//*server:port/dbname*
Specifies a JDBC URL for establishing the database connection. The DB2 JDBC type 4 driver is used to establish the connection. Required when online checking or automatic bind are enabled.

**-datasource** *jndiName*
Specifies a JNDI registered DataSource name for establishing the database connection for online checking or automatic binding. The registered name must map to a Universal Driver data source configured for Type 4 connectivity.

**-user** *username*
Specifies the name used when connecting to a database. This parameter is not required if the -datasource option is specified.

**-password** *password*
Specifies the password for the user name. This parameter is not required if the -datasource option is specified.

**-automaticbind**
Determines whether the **db2sqljbind** command is automatically invoked to create packages on the target database. Valid values are YES and NO. The default is YES. If enabled, -url must also be specified.

**-bindoptions** *"bind options"*
Specifies a list of bind options. The following options are supported. For detailed descriptions, see the BIND command.

- For DB2 for Linux, UNIX, and Windows:
  - ACTION (but not ACTION RETAIN)
  - BLOCKING
  - COLLECTION
  - DEGREE
  - EXPLAIN
  - EXPLSNAP
  - FEDERATED
  - FUNCPATH
  - INSERT

- ISOLATION (see the -singlepkgname option below)
- OWNER
- QUALIFIER
- QUERYOPT
- REOPT
- SQLERROR (but not SQLERROR CHECK)
- SQLWARN
- STATICREADONLY
- VALIDATE
- VERSION

- For DB2 on servers other than Linux, UNIX, and Windows:
  - ACTION (but not ACTION RETAIN)
  - BLOCKING
  - COLLECTION
  - DBPROTOCOL
  - DEGREE
  - EXPLAIN
  - IMMEDWRITE
  - ISOLATION (see the -singlepkgname option of the db2sqljcustomize command)
  - OPTHINT
  - OWNER
  - PATH
  - QUALIFIER
  - RELEASE
  - REOPT
  - SQLERROR
  - VALIDATE
  - VERSION

**-collection** *name*
: Specifies the default collection identifier. If not specified, NULLID is used. The default collection is used at runtime if a collection has not been explicitly set with the SET CURRENT PACKAGESET statement.

**-onlinecheck**
: Determines if online checking is to be performed using the database specified by the -url option. Valid values are: YES and NO. Default is YES if -url has been specified; otherwise, default is NO.

**-pkgversion** *versionname*
: Specifies the package version name to be used when binding packages at the server for a serialized profile. The version name is stored in the serialized profile to assist in manual version matching with the package at the server. Runtime version verification is based on the consistency token, not the version name. To automatically generate a version name based on the consistency token, enter the value AUTO (all capitals) for this parameter.

**-qualifier** *name*
: Provides a default dynamic qualifier for online checking. The value provided will be used to make a call to SET CURRENT SQLID before online checking begins. The default is the default qualifier for dynamic SQL. Because the dynamic default qualifer (used by DB2 for online checking) might be different than the static default qualifier (used by DB2 at run time), use of the -qualifier option will ensure that the correct object is online checked if there are any unqualified objects in the SQL statements. The value provided for this option will not automatically be

used for the bind. The QUALIFIER bind option must be explictly provided on the -bindoptions string. Conversely, a value provided on the -bindoptions string will not be used for online checking unless it is provided to the -qualifier option as well.

**-rootpkgname** *name*

Specifies the root name of the packages that are to be generated by the SQLJ binder. If this option is not specified, a root name is derived from the name of the profile. Maximum length is seven characters. The digits 1, 2, 3, and 4 are appended to the root name to create the four final package names (one for each isolation level).

**Note:** By specifying the root name of the packages, you can ensure that the packages have unique names. If packages are created that have the same names as existing packages, the existing packages will be overwritten.

**-singlepkgname** *name*

Specifies the package name to be generated by the SQLJ binder. Maximum length is eight characters. This option requires the ISOLATION bind option to be specifed on the -bindoptions flag. This should be used only for applications that use a single transaction isolation level.

**Note:** By specifying the package name, you can ensure that the name is unique. If a package is created that has the same name as an existing package, the existing package will be overwritten.

**-staticpositioned**

If the iterator is declared in the same program as the update statement, this option enables positioned updates to occur with the use of a statically bound statement, rather than with a dynamically prepared statement. Values are YES and NO. The default value is NO. Multiple files can be customized together in order to combine them into a single package on DB2. In this case any positioned update or delete statement referencing a cursor declared earlier in the resulting package will execute statically if -staticpositioned is set to YES, even if the statements did not originate from the same source file. However, the order in which the list of profiles is provided is critical to achieve this effect when combining multiple source files. The user must ensure that the cursor sections are processed prior to the update or delete statements. To do so, the profiles containing the query statements must be listed before the profiles containing the positioned update and delete statements that reference the resulting iterators. If the profiles are not provided in this order, the performance enhancement of setting -staticpositioned to YES will not be achieved.

**-tracefile** *name*

Enables tracing and identifies the output file for trace information. Should only be used when instructed by an IBM service technician.

**-tracelevel**

Identifies the level of tracing. If -tracelevel is omitted, TRACE_ALL is used.

*profilename*

Specifies the relative or absolute name of an SQLJ profile file. When an SQLJ file is translated into a Java file, information about the SQL operations it contains is stored in SQLJ-generated resource files called profiles. Profiles are identified by the suffix _SJProfileN (where N is an integer) following the name of the original input file. They have a .ser extension. Profile names can be specified with or without the .ser extension. Multiple files can be customized and bound together into a single package on DB2. A list of profiles can be provided on the command line call to **db2sqljcustomize**. Alternatively, a list of profiles can be listed one per line in a file with a .grp extension, and the .grp file name provided to **db2sqljcustomize**. The customizer will prepare the profiles in the list to be executed from a single DB2 package, and the implicit call to the binder will bind all the statements together. When combining profiles, the package name must be specified by either the -rootpkgname or -singlepkgname options. If the binder is called separately, the same list of files must be provided (in the same order) in order to combine the profiles into a single package.

**Examples:**

```
db2sqljcustomize -user richler -password mordecai
  -url jdbc:db2:/server:50000/sample -collection duddy
  -bindoptions "EXPLAIN YES" pgmname_SJProfile0.ser
```

**Usage notes:**

Implications of using the -staticpositioned YES option:

SQLJ allows iterators to be passed between methods as variables. An iterator that is passed as a variable and is used for a positioned update (UPDATE or DELETE) can be identified only at runtime. Also, the same SQLJ positioned update statement can be used with different iterators at runtime. When the SQLJ customizer prepares positioned update statements to execute statically, it must determine which queries belong to which positioned update statements. The SQLJ customizer does this by using the iterator's class to map between query statements and positioned UPDATE statements. If the iterator class does not provide a unique mapping between query statement and positioned update, the SQLJ customizer cannot determine exactly which query and positioned update statements belong together. A positioned update must be prepared once for each matching query statement (that is, query statements that use the same iterator class as the positioned update). If there is not a unique mapping from query statement to positioned update statement, this can result in a bind error. The following code fragment shows this point:

```
#sql iterator GeneralIter implements ForUpdate ( String );

public static void main( String args[] )
{
  ...
  GeneralIter iter = null;
  #sql [conn] iter = { SELECT CHAR_COL1 FROM TABLE1 };
  doUpdate( iter );
  ...
  #sql [conn] iter = { SELECT CHAR_COL2 FROM TABLE2 };
  ...
}

public static void doUpdate( GeneralIter iter )
{
  #sql [conn] { UPDATE TABLE1 ... WHERE CURRENT OF :iter };
}
```

In this example, only one iterator class is defined. Two instances of the iterator are created, and each is associated with a different SELECT statement that retrieves data from a different table. Because the iterator is passed to method doUpdate as a variable, it is impossible to know until runtime which of the iterator instances is used for the positioned UPDATE. The DB2 bind process will attempt to bind both queries to the positioned update, causing a bind error on the second SELECT.

You can avoid a bind time error for a program like the one above by specifying the DB2 BIND option SQLERROR(CONTINUE). However, a better technique is to write the program so that there is a unique mapping between iterator classes, queries, and positioned UPDATEs or DELETEs. The example below demonstrates how to do this. With this method of coding, each iterator class is associated with only one iterator instance. Therefore, the DB2 bind process can always associate the positioned UPDATE statement with the correct query.

```
#sql iterator Table1Iter implements ForUpdate ( String );
#sql iterator Table2Iter ( String );

public static void main ( String args[] )
{
  ...
  Table1Iter iter1 = null;
  #sql [conn] iter1 = { SELECT CHAR_COL1 FROM TABLE1 };
  Table2Iter iter2 = null;
  #sql [conn] iter2 = { SELECT CHAR_COL2 FROM TABLE2 };
  ...
  updateTable1( iter1 );
}

public static void updateTable1 ( Table1Iter iter )
{
  #sql [conn] { UPDATE TABLE1 ... WHERE CURRENT OF :iter };
}
```

**db2profc** is deprecated in DB2 Version 8, but can be specified in place of **db2sqljcustomize**. **db2profc** will not be supported in DB2 Version 9.

**Related reference:**

- “BIND” on page 286
- “db2sqljprint - DB2 SQLJ Profile Printer” on page 219
- “db2sqljbind - DB2 SQLJ Profile Binder” on page 210

## db2sqljprint - DB2 SQLJ Profile Printer

Prints the contents of a DB2 customized version of a profile in plain text.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──┬─db2sqljprint─┬──profilename──────────────────────────────────────►◄
    └─db2profp─────┘
```

**Command parameters:**

*profilename*
: Specifies the relative or absolute name of an SQLJ profile file. When an SQLJ file is translated into a Java file, information about the SQL operations it contains is stored in SQLJ-generated resource files called profiles. Profiles are identified by the suffix _SJProfileN (where N is an integer) following the name of the original input file. They have a `.ser` extension. Profile names can be specified with or without the `.ser` extension.

**Examples:**

```
db2sqljprint pgmname_SJProfile0.ser
```

**Usage notes:**

Currently, **db2profp** can be specified in place of **db2sqljprint**. However, **db2profp** will be deprecated in DB2 Version 9.

**Related reference:**

- "db2sqljcustomize - DB2 SQLJ Profile Customizer" on page 213
- "db2sqljbind - DB2 SQLJ Profile Binder" on page 210

# db2start - Start DB2

Starts the current database manager instance background processes on a single database partition or on all the database partitions defined in a partitioned database environment. Start DB2 at the server before connecting to a database, precompiling an application, or binding a package to a database.

**db2start** can be executed as a system command or a CLP command.

**Related reference:**

- "START DATABASE MANAGER" on page 690

## db2stop - Stop DB2

Stops the current database manager instance.

**db2stop** can be executed as a system command or a CLP command.

**Related reference:**

- "STOP DATABASE MANAGER" on page 698

## db2support - Problem Analysis and Environment Collection Tool

Collects environment data about either a client or server machine and places the files containing system data into a compressed file archive.

This tool can also collect basic data about the nature of a problem through an interactive question and answer process with the user.

**Authorization:**

For the most complete output, this utility should be invoked by the instance owner. Users with more limited privileges on the system can run this tool, however some of the data collection actions will result in reduced reporting and reduced output.

**Required connection:**

None

**Command syntax:**

```
►►──db2support──output path──┬────┬──┬────┬──────────────────────────────────►
                             └─-f─┘  ├─-a─┤
                                     └─-r─┘

►──┬────────────────────────────────────────────────────────┬──┬────┬────────►
   └─-d──database name──┬────────────────────────────────┬──┘  └─-g─┘
                        └─-c──┬────────────────────────┬─┘
                              └─-u──userid──┬────────────────┬─┘
                                            └─-p──password───┘

►──┬────┬──┬────┬──┬────┬──┬────┬──┬────┬──┬────┬──┬────┬──┬────┬────────────►◄
   └─-h─┘  └─-l─┘  └─-m─┘  └─-n─┘  └─-q─┘  └─-s─┘  └─-v─┘  └─-x─┘
```

**Command parameters:**

*output path*
: Specifies the path where the archived library is to be created. This is the directory where user created files must be placed for inclusion in the archive.

**-f or -flow**
: Ignores pauses when requests are made for the user to `Press <Enter> key to continue`. This option is useful when running or calling the **db2support** tool via a script or some other automated procedure where unattended execution is desired.

**-a or -all_core**
: Specifies that all the core files are to be captured.

**-r or -recent_core**
: Specifies that the most recent core files are to be captured. This option is ignored if the -a option is specified.

**-d** *database_name* **or -database** *database_name*
: Specifies the name of the database for which data is being collected.

**-c or -connect**
: Specifies that an attempt be made to connect to the specified database.

**-u** *userid* **or -user** *userid*
Specifies the user ID to connect to the database.

**-p** *password* **or -password** *password*
Specifies the password for the user ID.

**-g or -get_dump**
Specifies that all files in a dump directory, excluding core files, are to be captured.

**-h or -help**
Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-l or -logs**
Specifies that active logs are to be captured.

**-m or -html**
Specifies that all system output is dumped into HTML formatted files. By default, all system related information is dumped into flat text files if this parameter is not used.

**-n or -number**
Specifies the problem management report (PMR) number or identifier for the current problem.

**-q or -question_response**
Specifies that interactive problem analysis mode is to be used.

**-s or -system_detail**
Specifies that detailed hardware and operating system information is to be gathered.

**-v or -verbose**
Specifies that verbose output is to be used while this tool is running.

**-x or -xml_generate**
Specifies that an XML document containing the entire decision tree logic used during the interactive problem analysis mode (-q mode) is to be generated.

**Usage notes:**

In order to protect the security of business data, this tool does not collect table data, schema (DDL), or logs. Some of the options do allow for the inclusion of some aspects of schema and data (such as archived logs). Options that expose database schema or data should be used carefully. When this tool is invoked, a message is displayed that indicates how sensitive data is dealt with.

Data collected from the db2support tool will be from the machine where the tool runs. In a client-server environment, database-related information will be from the machine where the database resides via an instance attachment or connection to the database. For example, operating system or hardware information (-s option) and files from the diagnostic directory (DIAGPATH) will be from the local machine where the db2support tool is running. Data such as buffer pool information, database configuration, and table space information will be from the machine where the database physically resides.

## db2sync - Start DB2 Synchronizer

Facilitates the initial configuration of a satellite as well as changes to the configuration. This command can also be used to start, stop and monitor the progress of a synchronization session and to upload a satellite's configuration information (for example, communications parameters) to its control server.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──db2sync──┬─────────────────────────┬──────────────────────────────────────►◄
             ├─-t──────────────────────┤
             ├─-s──application_version─┤
             └─-g──────────────────────┘
```

**Command parameters:**

**-t** Displays a graphical user interface that allows an administrator to change either the application version or synchronization credentials for a satellite.

**-s application_version**
Sets the application version on the satellite.

**-g** Displays the application version currently set on the satellite.

# db2systray - Start DB2 System Tray

Starts the DB2 system tray tool. **db2systray** is a Windows system tray tool for monitoring the status of a local DB2 instance. The icon in the system tray will change based on the state of the instance being monitored. **db2systray** displays the status of ESE instances as stopped when one or more database partitions are stopped, and started when all database partitions are started. The instance can be started and stopped using this tool. Starting or stopping an ESE instance using **db2systray** will start or stop all database partitions in the instance. **db2systray** will only stop the instance when there are no connected applications.

**db2systray** is only available on Windows platforms.

**Authorization:**

No special authority is required for starting **db2systray**. Appropriate authority is required for taking actions.

**Required connection:**

None

**Command syntax:**

```
►►──db2systray──┬───────────────┬──────────────────────────────────────────►◄
                └─instance-name─┘
```

**Command parameters:**

*instance-name*
: Name of the DB2 instance to be monitored. If no instance name is specified, **db2systray** will monitor the default local DB2 instance. If no instance exists, or the specified instance is not found, **db2systray** will exit quietly.

# db2tapemgr - Manage Log Files on Tape

Allows the storage and retrieval of DB2 log files to and from tape. The location on tape is stored in the history file.

**Scope:**

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

**Command syntax:**

```
►►──db2tapemgr──┬─────────────────────────────────────────┬──┬──────────────────────┬──────────►
                └─┬─DATABASE─┬──source-database-alias─┘  └─ON DBPARTITIONNUM──n─┘
                  └─DB───────┘

►──┬─┬─STORE────────┬─┤ store option clause ├─┬──┬───────────────────┬──┬───────┬──────────►◄
   │ └─DOUBLE STORE─┘                         │  └─USING──blocksize─┘  └─EJECT─┘
   ├─RETRIEVE──┤ retrieve option clause ├─────┤
   ├─SHOW TAPE HEADER──tape device────────────┤
   ├─EJECT TAPE──tape device──────────────────┤
   ├─DELETE TAPE LABEL──tape label────────────┤
   └─QUERY──┤ for rollforward clause ├────────┘
```

### store option clause:

```
                                                  ┌─ALL LOGS─┐
├──ON──tape device──┬─────────────────────────┬──┼──────────┼──┬───────┬──────────┤
                    └─TAPE LABEL──tape label─┘  └─n──LOGS───┘  └─FORCE─┘
```

### retrieve option clause:

```
├──┬─┬─┤ for rollforward clause ├─┬──FROM──tape device──┬────────────────┬─┬──────┤
   │ ├─ALL LOGS───────────────────┤                     └─TO──directory─┘ │
   │ └─LOGS──n──TO──m─────────────┘                                        │
   └─HISTORY FILE──FROM──tape device──TO──directory────────────────────────┘
```

### for rollforward clause:

```
   ┌─FOR ROLLFORWARD TO END OF LOGS──────────────────────────┐
├──┼─────────────────────────────────────────────────────────┼────────────────►
   │                        ┌─local time──USING LOCAL TIME─┐ │
   └─FOR ROLLFORWARD TO─────┼──────────────────────────────┼─┘
                            └─isotime──USING GMT TIME──────┘

►──┬──────────────────────────────────┬──────────────────────────────────────┤
   └─USING HISTORY FILE──history file─┘
```

**Command parameters:**

**DATABASE** *source-database-alias*

Specifies the name of the database. If no value is specified, DB2DBDFT will be used. If no value is specified, and DB2DBDFT is not set, the operation fails.

**ON DBPARTITIONNUM**

Specifies the database partition number to work on. If no value is specified, DB2NODE is used.

**STORE ON** *tape device*

Stores log file to tape and deletes it.

**DOUBLE STORE ON** *tape device*

Stores all log files that have been stored only once and those log files never stored. Deletes only the log files that have been stored twice to tape; others are kept on disk.

**TAPE LABEL**

Specifies a label to be applied to the tape. If *tape label* is not specified, one will be generated automatically in the following format: *database-alias|timestamp* (up to 22 characters, up to 8 characters for the database alias and 14 characters for the time stamp in seconds).

**ALL LOGS or** *n* **LOGS**

Specifies that the command applies to all logs or a specified number of logs.

**FORCE**

Specifies that if the tape has not expired, then over write it.

**USING** *blocksize*

Specifies the block size for tape access. The default size is 5120, and it must be a multiple of 512. The minimum is 512.

**EJECT** Specifies that the tape is to be ejected after the operation completes.

**RETRIEVE FOR ROLLFORWARD TO**

Specifies that the utility will interactively prompt for all logs that are required for the specified rollforward and retrieve them from tape. If a directory is not specified, the path specified by the *overflowlogpath* configuration parameter is used. If a directory is not specified and *overflowlogpath* is not set, the operation fails.

**END OF LOGS**

Specifies that log files up to the end of the log will be retrieved.

*isotime USING GMT TIME*

Specifies that log files up to the time specified will be retrieved.

*local time USING LOCAL TIME*

Specifies that log files up to the time specified will be retrieved.

**USING HISTORY FILE** *history file*

Specifies an alternate history file to be used.

**FROM** *tape device*

**TO** *directory*

**RETRIEVE ALL LOGS or LOGS** *n* **TO** *m*

**FROM** *tape device*

**TO** *directory*

**RETRIEVE HISTORY FILE**
Retrieves the history file

**FROM** *tape device*

**TO** *directory*

**SHOW TAPE HEADER** *tape device*
Shows the content of the tape header file `DB2TAPEMGR.HEADER`

**EJECT TAPE** *tape device*
Ejects the tape.

**DELETE TAPE LABEL** *tape label*
Deletes all locations from the history file that refer to the specified tape label.

**QUERY FOR ROLLFORWARD TO**
Displays the location of the log files that are required for rollforward.

**END OF LOGS**

*isotime* **USING GMT TIME**
Specifies that the operation should query the logs up to the time specified.

*local time* **USING LOCAL TIME**
Specifies that the operation should query the logs up to the time specified.

**USING HISTORY FILE** *history file*
Specifies an alternate history file to be used.

**Examples:**

**Usage notes:**

## db2tbst - Get Tablespace State

Accepts a hexadecimal table space state value, and returns the state. The state value is part of the output from LIST TABLESPACES.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──db2tbst──tablespace-state─────────────────────────────────────────────►◄
```

**Command parameters:**

**tablespace-state**
A hexadecimal table space state value.

**Examples:**

The request `db2tbst 0x0000` produces the following output:

```
State = Normal
```

**Related reference:**

- "LIST TABLESPACES" on page 513

# db2trc - Trace

Controls the trace facility of a DB2 instance or the DB2 Administration Server. The trace facility records information about operations and formats this information into readable form. Enabling the trace facility might impact your system's performance. As a result, only use the trace facility when directed by a DB2 Support technical support representative.

**Authorization:**

To trace a DB2 instance on a UNIX-based system, one of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

To trace the DB2 Administration Server on a UNIX-based system:

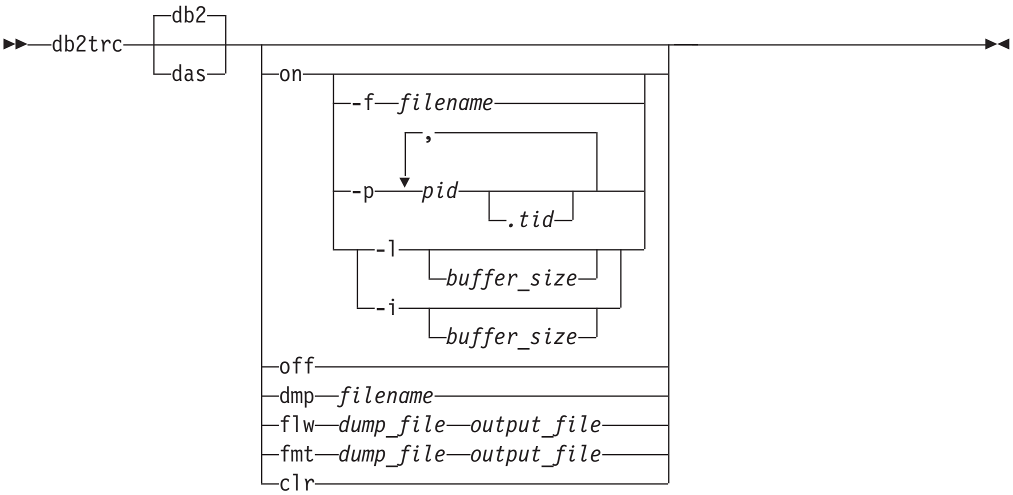
- *dasadm*

On a Windows operating system, no authorization is required.

**Required connection:**

None

**Command syntax:**



**Command parameters:**

**db2**
: Specifies that all trace operations will be performed on the DB2 instance. This is the default.

**das**
: Specifies that all trace operations will be performed on the DB2 Administration Server.

**on**
: Use this parameter to start the trace facility.

**-f** *filename*
: Specifies that trace information should be continuously written to the specified file, until **db2trc** is turned off.

**Note:** Using this option can generate an extremely large dump file. Use this option only when instructed by DB2 Support.

**-p** *pid.tid*

Only enables the trace facility for the specified process IDs (pid) and thread IDs (tid). The period (.) must be included if a tid is specified. A maximum of five pid.tid combinations is supported.

For example, to enable tracing for processes 10, 20, and 30 the syntax is:

```
db2trc on -p 10,20,30
```

To enable tracing only for thread 33 of process 100 and thread 66 of process 200 the syntax is:

```
db2trc on -p 100.33,200.66
```

**-l [** *buffer_size*] **|** **-i** [*buffer_size*]

This option specifies the size and behavior of the trace buffer.'-l' specifies that the last trace records are retained (that is, the first records are overwritten when the buffer is full). '-i' specifies that the initial trace records are retained (that is, no more records are written to the buffer once it is full). The buffer size can be specified in either bytes or megabytes. To specify the buffer size in megabytes, add the character "m" to the buffer size. For example, to start **db2trc** with a 4–megabyte buffer:

```
db2trc on -l 4m
```

The default and maximum trace buffer sizes vary by platform. The minimum buffer size is 1 MB.

**Note:** The buffer size must be a power of 2.

**dmp** Dumps the trace information to a file. The following command will put the information in the current directory in a file called `db2trc.dmp`:

```
db2trc dmp db2trc.dmp
```

Specify a file name with this parameter. The file is saved in the current directory unless the path is explicitly specified.

**off** After the trace is dumped to a file, stop the trace facility by typing:

```
db2trc off
```

**flw | fmt**

After the trace is dumped to a binary file, confirm that it is taken by formatting it into a text file. Use either the flw option (to format records sorted by process or thread), or the fmt option (to format records chronologically). For either option, specify the name of the dump file and the name of the output file that will be generated. For example:

```
db2trc flw db2trc.dmp db2trc.flw
```

**clr** Clears the contents of the trace buffer. This option can be used to reduce the amount of collected information. This option has no effect when tracing to a file.

**Usage notes:**

The **db2trc** command must be issued several times to turn tracing on, produce a dump file, format the dump file, and turn tracing off again. The parameter list shows the order in which the parameters should be used.

The default and maximum trace buffer sizes vary by platform. The minimum buffer size is 1 MB.

When tracing the database server, it is recommended that the trace facility be turned on prior to starting the database manager.

# db2undgp - Revoke Execute Privilege

Revoke the execute privilege on external stored procedures. This command can be used against external stored procedures.

During the database migration, EXECUTE for all existing functions, methods, and External stored procedure is granted to PUBLIC. This will cause a security exposure for External Stored procedures that contain SQL data access. To prevent users from accessing SQL objects which the user might not have privilege for, use the db2undgp command.

**Command syntax:**

```
►►──db2undgp──┬─────────────┬──┬────┬──┬───────────────┬──┬────┬──────────────►◄
              └─-d──dbname──┘  └─-h─┘  └─-o──outfile───┘  └─-r─┘
```

**Command parameters:**

**-d** *dbname*
database name (maximum of 8 characters)

**-h** Displays help for the command.

**-o** *outfile*
output the revoke statements in the specified file File name length <= 80

**-r** perform the revoke

**Usage notes:**

**Notes:**

1. At least one of the -r or -o options must be specified.

## db2uiddl - Prepare Unique Index Conversion to V5 Semantics

Facilitates the management of a staged migration of unique indexes on a user's own schedule. Generates CREATE UNIQUE INDEX statements for unique indexes on user tables.

**Authorization:**

*sysadm*

**Required connection:**

Database. This command automatically establishes a connection to the specified database.

**Command syntax:**

```
►►──db2uiddl──-d──database-name──┬──────────────────┬──┬───────────────┬──────────►
                                 └─-u──table-schema─┘  └─-t──table-name─┘

►──┬──────────────┬──┬────┬──────────────────────────────────────────────────────►◄
   └─-o──filename─┘  └─-h─┘
```

**Command parameters:**

**-d database-name**
The name of the database to be queried.

**-u table-schema**
Specifies the schema (creator user ID) of the tables that are to be processed. The default action is to process tables created by all user IDs.

**-t table-name**
The name of a table that is to be processed. The default action is to process all tables.

**-o filename**
The name of a file to which output is to be written. The default action is to write output to standard output.

**-h** Display help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Usage notes:**

It is not necessary to use this tool unless there are indexes in the database that were created on a database running on a version of DB2 earlier than Version 5.

**Note:** This tool was not designed to handle certain types of names. If a specific table name or table schema is a delimited identifier containing lowercase characters, special characters, or blanks, it is preferable to request processing of *all* tables or schemas. The resulting output can be edited.

## db2untag - Release Container Tag

Removes the DB2 tag on a table space container. The tag is used to prevent DB2 from reusing a container in more than one table space. Displays information about the container tag, identifying the database with which the container is associated. Useful when it is necessary to release a container last used by a database that has since been deleted. If the tag is left behind, DB2 is prevented from using the resource in future.

**Attention:** This tool should only be used by informed system administrators.

**Authorization:**

The user needs read/write access to the container for a table space that is owned by the ID that created the database.

**Required connection:**

None

**Command syntax:**

```
►►──db2untag──-f──filename──────────────────────────────────────────►◄
```

**Command parameters:**

**-f filename**
Specifies the fully qualified name of the table space container from which the DB2 tag is to be removed.

**Usage notes:**

An SQLCODE -294 (Container in Use error) is sometimes returned from create database or from create or alter table space operations, usually indicating a specification error on the operating system resource name when the container is already in use by another table space. A container can be used by only one table space at a time.

A system or database administrator who finds that the database which last used the container has been deleted, can use the **db2untag** tool if the container's tag was not removed. If the container is to be released, do one of the following:

- For SMS containers, remove the directory and its contents using the appropriate delete commands.
- For DMS raw containers, either delete the file or device, or let **db2untag** remove the container tag. The tool will leave such a DMS container otherwise unmodified.

**Related reference:**

- "CREATE DATABASE" on page 331

# db2updv8 - Update Database to Version 8 Current Level

This command updates the system catalogs in a database to support the current level by enabling the following built-in routines:

- HEALTH_DBM_INFO
- HEALTH_DBM_HI
- HEALTH_DBM_HI_HIS
- HEALTH_DB_INFO
- HEALTH_DB_HI
- HEALTH_DB_HI_HIS
- HEALTH_TBS_INFO
- HEALTH_TBS_HI
- HEALTH_TBS_HI_HIS
- HEALTH_CONT_INFO
- HEALTH_CONT_HI
- HEALTH_CONT_HI_HIS
- GET_WRAP_CFG_C
- LIST_FILES_C
- ODBC_DISCOVERY
- HEALTH_HI_REC
- EXEC_DB2_SCRIPT
- HEALTH_HI_REC
- EXEC_DB2_SCRIPT

**Authorization:**

sysadm

**Required connection:**

Database. This command automatically establishes a connection to the specified database.

**Command syntax:**

```
►►──db2updv8──-d──database-name──┬───────────────────────────┬──┬────┬──────────►◄
                                 └─-u──userid──-p──password─┘  └─-h─┘
```

**Command parameters:**

**-d database-name**
Specifies the name of the database to be updated.

**-u userid**
Specifies the user ID.

**-p password**
Specifies the password for the user.

**-h**
Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**Example:**

After installing the current level (a FixPak or a new version), update the system catalog in the sample database by issuing the following command:

```
db2updv8 -d sample
```

**Usage notes:**

1. This tool can only be used on a database running DB2 Version 8.1.2 or later. If the command is issued more than once, no errors are reported and each of the catalog updates is applied only once.
2. To enable the new built-in functions, all applications must disconnect from the database and the database must be deactivated if it has been activated.

## disable_MQFunctions

### Purpose

Disables the use of DB2 WebSphere MQ functions for the specified database.

### Authorization

One of the following:

- *sysadm*
- *dbadm*
- IMPLICIT_SCHEMA on the database, if the implicit or explicit schema name of the function does not exist
- CREATEIN privilege on the schema, if the schema name, DB2MQ or DB2MQ1C exists

### Format

```
►►──disable_MQFunctions── -n──database── -u──userid── -p──password──────────────►

►──┬──────────────┬──────────────────────────────────────────────────────────────►◄
   └─ -v──┬─────┬─┘
          ├─all─┤
          ├─0pc─┤
          └─1pc─┘
```

### Parameters

**–n database**
: Specifies the name of the database.

**–u userid**
: Specifies the user ID used to connect to the database.

**–p password**
: Specifies the password for the user ID.

**–v**
: Optional. This is used for transactional and non-transactional user-defined function support. The values can be either all, 0pc, or 1pc. When you specify 0pc, the disablement deletes from schema db2mq. If you specify 1pc, then the disablement deletes from schema db2mq1c. If you specify all, then the disablement deletes from both schemas (db2mq and db2mq1c). If you do not specify this option, the disablement defaults to the all option.

### Example

In the following example, DB2MQ and DB2MQ1C functions are disabled for the database SAMPLE.

```
   disable_MQFunctions -n sample -u user1 -p password1
```

**Related concepts:**

- "MQSeries Enablement" in the *Application Development Guide: Programming Client Applications*
- "WebSphere MQ Functional Overview" in the *Application Development Guide: Programming Client Applications*

- “WebSphere MQ Messaging” in the *Application Development Guide: Programming Client Applications*
- “Sending Messages with WebSphere MQ Functions” in the *Application Development Guide: Programming Client Applications*
- “Retrieving Messages with WebSphere MQ Functions” in the *Application Development Guide: Programming Client Applications*
- “WebSphere MQ Application-to-application Connectivity” in the *Application Development Guide: Programming Client Applications*
- “Request/Reply Communications with WebSphere MQ Functions” in the *Application Development Guide: Programming Client Applications*
- “Publish/Subscribe with WebSphere MQ Functions” in the *Application Development Guide: Programming Client Applications*
- “How to use WebSphere MQ functions within DB2” in the *IBM DB2 Information Integrator Application Developer’s Guide*

**Related tasks:**

- “Setting up DB2 WebSphere MQ functions” in the *Application Development Guide: Building and Running Applications*

**Related reference:**

- “MQSEND scalar function” in the *SQL Administrative Routines*
- “MQRECEIVE scalar function” in the *SQL Administrative Routines*
- “MQREAD scalar function” in the *SQL Administrative Routines*
- “MQPUBLISH scalar function” in the *SQL Administrative Routines*
- “MQSUBSCRIBE scalar function” in the *SQL Administrative Routines*
- “MQUNSUBSCRIBE scalar function” in the *SQL Administrative Routines*
- “MQREADALL table function” in the *SQL Administrative Routines*
- “MQRECEIVEALL table function” in the *SQL Administrative Routines*
- “MQREADCLOB scalar function” in the *SQL Administrative Routines*
- “MQRECEIVECLOB scalar function” in the *SQL Administrative Routines*
- “MQREADALLCLOB table function” in the *SQL Administrative Routines*
- “MQRECEIVEALLCLOB table function” in the *SQL Administrative Routines*
- “db2mqlsn - MQ Listener” on page 139
- “enable_MQFunctions” on page 240

## enable_MQFunctions

Enables DB2 WebSphere MQ functions for the specified database and validates that the DB2 WebSphere MQ functions can be executed properly. The command fails if WebSphere MQ and WebSphere MQ AMI have not been installed and configured.

**Authorization:**

One of the following:
- *sysadm*
- *dbadm*
- IMPLICIT_SCHEMA on the database, if the implicit or explicit schema name of the function does not exist
- CREATEIN privilege on the schema, if the schema name, DB2MQ or DB2MQ1C, exists

**Command syntax:**

```
►►──enable_MQFunctions──-n──database──-u──userid──-p──password──────────────────►

►──┬──────────────────┬──┬────────┬──┬─────────────┬──┬──────────────┬──────────►◄
   └─-q──queuemanager─┘  └─-force─┘  └─-novalidate─┘  └─ -v──┬─────┬─┘
                                                            ├─all─┤
                                                            ├─0pc─┤
                                                            └─1pc─┘
```

**Command parameters:**

**–n** Specifies the name of the database that you want to enable.

**–u** Specifies the user ID to connect to the database.

**–p** Specifies the password for the user ID.

**–q** Optional. The queue manager name that supports the transactional MQ user-defined functions. If you do not specify a name, it is the default queue manager, DB2MQ_DEFAULT_MQM. If you use this option, the function assumes the use of a -novalidate parameter.

**–force** Optional. The use of this option allows the utility program to ignore the existing MQ UDFs. In other words, the program drops any existing functions, before recreating any MQ UDFs. Without this option, the command will not proceed after it finds that the MQ UDFs already exist.

**–novalidate**
Optional. This specifies that there will not be any validation of the DB2 MQSeries functions.

**–v** Optional. This is used for transactional and non-transactional user-defined function support. The values can be either all, 0pc, or 1pc. When you specify 0pc, the enablement creates schema db2mq. If you specify 1pc, then the enablement creates schema db2mq1c. If you specify all, then the enablement creates all schemas under user-defined functions (db2mq and db2mq1c). If you do not specify this option, the enablement defaults to the all option.

**Examples:**

The following example enables the transactional and non-transactional user-defined functions. The user connects to the database SAMPLE.

```
enable_MQFunctions -n sample -u user1 -p password1
```

In the next example, the user connects to the database SAMPLE. The example creates DB2MQ1C functions with schema DB2MQ1C.

```
enable_MQFunctions -n sample -u user1 -p password1 -v 1pc
```

**Usage notes:**

The DB2 MQ user-defined functions run under the schemas DB2MQ or DB2MQ1C which are automatically created by this command. Before executing this command:

- Ensure that WebSphere MQ and WebSphere Application Messaging Interface (AMI) are installed, and that the version of WebSphere MQ is 5.1 or higher.
- Ensure that the environment variable $AMT_DATA_PATH is defined.
- If you want to use transactional MQ UDFs, make sure that the database is configured for federated operations. Do this with the following command

  ```
  update dbm cfg using federated yes
  ```
- Change the directory to the cfg subdirectory of the DB2PATH

On UNIX:

- Use db2set to add AMT_DATA_PATH to the DB2ENVLIST.
- Ensure that the user account associated with UDF execution is a member of the mqm group.
- Ensure that the user who will be calling this command is a member of the mqm group.

**Note:** AIX 4.2 is not supported by MQSeries 5.2.

**Related concepts:**

- "MQSeries Enablement" in the *Application Development Guide: Programming Client Applications*
- "WebSphere MQ Functional Overview" in the *Application Development Guide: Programming Client Applications*
- "WebSphere MQ Messaging" in the *Application Development Guide: Programming Client Applications*
- "Sending Messages with WebSphere MQ Functions" in the *Application Development Guide: Programming Client Applications*
- "Retrieving Messages with WebSphere MQ Functions" in the *Application Development Guide: Programming Client Applications*
- "WebSphere MQ Application-to-application Connectivity" in the *Application Development Guide: Programming Client Applications*
- "Request/Reply Communications with WebSphere MQ Functions" in the *Application Development Guide: Programming Client Applications*
- "Publish/Subscribe with WebSphere MQ Functions" in the *Application Development Guide: Programming Client Applications*
- "How to use WebSphere MQ functions within DB2" in the *IBM DB2 Information Integrator Application Developer's Guide*

**Related tasks:**

- "Setting up DB2 WebSphere MQ functions" in the *Application Development Guide: Building and Running Applications*

**Related reference:**

- "MQSEND scalar function" in the *SQL Administrative Routines*
- "MQRECEIVE scalar function" in the *SQL Administrative Routines*
- "MQREAD scalar function" in the *SQL Administrative Routines*
- "MQPUBLISH scalar function" in the *SQL Administrative Routines*
- "MQSUBSCRIBE scalar function" in the *SQL Administrative Routines*
- "MQUNSUBSCRIBE scalar function" in the *SQL Administrative Routines*
- "MQREADALL table function" in the *SQL Administrative Routines*
- "MQRECEIVEALL table function" in the *SQL Administrative Routines*
- "MQREADCLOB scalar function" in the *SQL Administrative Routines*
- "MQRECEIVECLOB scalar function" in the *SQL Administrative Routines*
- "MQREADALLCLOB table function" in the *SQL Administrative Routines*
- "MQRECEIVEALLCLOB table function" in the *SQL Administrative Routines*
- "db2mqlsn - MQ Listener" on page 139
- "disable_MQFunctions" on page 238

## setup - Install DB2

Installs DB2 products. This command is only available on Windows operating systems. The command for UNIX-based systems is **db2setup**.

This utility is located on the DB2 installation media. It launches the DB2 Setup wizard to define the installation and install DB2 products. If invoked with the *-u* option, it performs an installation without further input, taking installation configuration information from a response file.

**Command syntax:**

```
►►──setup──┬────┬──┬──────────────┬──┬─────────────┬──┬───────────────┬──────►
           └─-f─┘  └─-i──language─┘  └─-l──log_file─┘  └─-t──trace_file─┘

►──┬─────────────────────┬──┬────┬──┬────┬─────────────────────────────────►◄
   └─-u──response_file───┘  └─-w─┘  ├─-?─┤
                                    └─-h─┘
```

**Command parameters:**

**-f** Forces any DB2 processes to stop before installing.

**-i** *language*
Two-letter language code of the language in which to perform the installation.

**-l** *log_file*
Full path and file name of the log file to use.

**-t** *trace_file*
Generates a file with install trace information.

**-u** *response_file*
Full path and file name of the response file to use.

**-w** Keeps the parent process alive until the installation finishes.

**-?, -h** Generates usage information.

**Related reference:**

- "db2setup - Install DB2" on page 206

## sqlj - DB2 SQLJ Translator

Translates an SQLJ source file into a Java source file and zero or more SQLJ serialized profiles. By default, the Java source file is also compiled.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──sqlj──┬─────────────────────┬──┬───────────────────┬──────────────────────────►
          └─-dir=──directory────┘  └─-d=──directory────┘

                                   ┌─-compile=true──┐  ┌─-linemap=no──┐
►──┬─────────────────────────────┬─┼────────────────┼──┼──────────────┼────────────►
   └─-props=──properties-file────┘ └─-compile=false─┘  └─-linemap=yes─┘

   ┌─-smap=no──┐
►──┼───────────┼──┬───────────────────────┬──┬────────────┬──┬─────────┬───────────►
   └─-smap=yes─┘  └─-encoding=──encoding──┘  └─-ser2class─┘  └─-status─┘

►──┬──────────┬──┬─────────┬──┬──────────────────────────┬─────────────────────────►
   └─-version─┘  └─-C-help─┘  │  ┌─.──────────────────┐  │
                              └──▼─-CJava compiler option─┴─┘

►──┬─────────────────────────┬──┬──────────────────────┬──┬───────┬────────────────►
   │  ┌─.─────────────────┐  │  │  ┌─.──────────────┐  │  └─-help─┘
   └──▼─-JJava VM option──┴──┘  └──▼─SQLJ source file─┴──┘

►──┬──────────────┬───────────────────────────────────────────────────────────────►◄
   └─-db2optimize─┘
```

**Command parameters:**

**-dir=***directory*
: Specifies the name of the directory into which SQLJ puts .java files that are generated by the translator. The default directory is the directory that contains SQLJ source files.

**-d=***directory*
: Directory root where generated .ser files are placed. The default is the current directory. This option is also passed on to the Java compiler.

**-props=***properties-file*
: Specifies the name of a file from which the SQLJ translator is to obtain a list of options.

**-compile=**

**true** This turns the compilation of generated Java files on. This is the default.

**false** This turns the compilation of generated Java files off.

**-linemap=**
Specifies whether line numbers in Java exceptions match line numbers in the SQLJ source files (the .sqlj file) or line numbers in the Java source file that is generated by the SQLJ translator files (the .java file).

**no** Line numbers in Java exceptions match line numbers in the Java source file. This is the default.

**yes** Line numbers in Java exceptions match line numbers in the SQLJ source file.

**-smap=**
Specifies whether the SQLJ translator generates a source map (SMAP) file for each SQLJ source file. An SMAP file is used by some Java language debug tools. This file maps lines in the SQLJ source file to lines in the Java source file that is generated by the SQLJ translator. The file is in the Unicode UTF-8 encoding scheme. Its format is described by Original Java Specification Request (JSR) 45, which is available from www.jcp.org.

**no** Does not generate SMAP files. This is the default.

**yes** Generates SMAP files. An SMAP file name is *SQLJ-source-file-name*.java.smap. The SQLJ translator places the SMAP file in the same directory as the generated Java source file.

**-encoding=***encoding*
Specifies the input and output character encoding of files. If this option is not specified, the file encoding is taken from the Java system property ″file.encoding″.

**-ser2class**
Converts the serialized profiles into class files. This might be necessary for running SQLJ executables in certain browsers that have problems loading .ser files.

**-status** Prints immediate status display of SQLJ processing.

**-version**
Shows the SQLJ translator build version. When this option is specified, all other options are ignored.

**-C-help**
Specifies that the SQLJ translator display help information for the Java compiler.

**-CJava compiler option**
Passes the option to the Java compiler. You can specify the class path and the source path the translator will use to find required dependency files. The class path and source path specified will be used by both the translator and the compiler.

**-JJava VM option**
Passes the option to the Java virtual machine.

**SQLJ source file**
One or more .sqlj files.

**-help** Displays help information. When this option is specified, all other options are ignored, and only the help information is displayed.

**-db2optimize**
Specifies that the SQLJ translator generates code for a connection context class that is optimized for DB2. This option optimizes the code for the user-defined context but not for the default context. When you run the SQLJ translator with this option, the DB2 Universal JDBC Driver file db2jcc.jar must be in the CLASSPATH for compiling the generated Java application.

**Examples:**

```
sqlj -encoding=UTF8 -C-O MyApp.sqlj
```

**Usage notes:**

Each serialized profile generated corresponds to a connection context class used in the SQLJ source code. The profiles are named as follows: `pgmname_SJProfile0.ser`, `pgmname_SJProfile1.ser`, `pgmname_SJProfile2.ser`, and so on, where the number represents the order in which the connection context classes are referenced in the source file. The **db2sqljprint** command can be used to view the contents of a profile. Each profile must be customized separately with the **db2sqljcustomize** command.

**Related reference:**

- "db2sqljcustomize - DB2 SQLJ Profile Customizer" on page 213
- "db2sqljprint - DB2 SQLJ Profile Printer" on page 219
- "db2sqljbind - DB2 SQLJ Profile Binder" on page 210

# Chapter 2. Command Line Processor (CLP)

## db2 - Command Line Processor Invocation

The **db2** command starts the command line processor (CLP). The CLP is used to execute database utilities, SQL statements and online help. It offers a variety of command options, and can be started in:

- Interactive input mode, characterized by the **`db2 =>`** input prompt
- Command mode, where each command must be prefixed by `db2`
- Batch mode, which uses the `-f` file input option.

**Note:** On Windows, **db2cmd** opens the CLP-enabled DB2 window, and initializes the DB2 command line environment. Issuing this command is equivalent to clicking on the *DB2 Command Window* icon.

QUIT stops the command line processor. TERMINATE also stops the command line processor, but removes the associated back-end process and frees any memory that is being used. It is recommended that a TERMINATE be issued prior to every STOP DATABASE MANAGER (db2stop) command. It might also be necessary for a TERMINATE to be issued after database configuration parameters have been changed, in order for these changes to take effect.

**Note:** Existing connections should be reset before terminating the CLP.

The shell command (**!**), allows operating system commands to be executed from the interactive or the batch mode on UNIX based systems, and on Windows operating systems (`!ls` on UNIX, and `!dir` on Windows operating systems, for example).

**Command Syntax:**

```
►►──db2──┬──────────────────────────────────────────────┬──────────────►◄
         │  ┌──────────────────┐                          │
         │  ▼                  │                          │
         ├──────┬─────────────┬┴──┬─db2-command──────┬────┤
         │      └─option-flag─┘   ├─sql-statement────┤    │
         │                        └─?──┬─phrase─────┬┘    │
         │                             ├─message────┤     │
         │                             ├─sqlstate───┤     │
         │                             └─class-code─┘     │
         └─--──comment────────────────────────────────────┘
```

**option-flag**
Specifies a CLP option flag.

**db2-command**
Specifies a DB2 command.

**sql-statement**
Specifies an SQL statement.

**?** Requests CLP general help.

**? phrase**
Requests the help text associated with a specified command or topic. If the database manager cannot find the requested information, it displays the general help screen.

`? options` requests a description and the current settings of the CLP options. `? help` requests information about reading the online help syntax diagrams.

**? message**
Requests help for a message specified by a valid SQLCODE (`? sql10007n`, for example).

**? sqlstate**
Requests help for a message specified by a valid SQLSTATE.

**? class-code**
Requests help for a message specified by a valid class-code.

**-- comment**
Input that begins with the comment characters `--` is treated as a comment by the command line processor.

**Note:** In each case, a blank space must separate the question mark (?) from the variable name.

**Related concepts:**

- "Command Line Processor (CLP)" on page 255

**Related reference:**

- "Command line processor options" on page 248
- "Command Line Processor Return Codes" on page 254

## Command line processor options

The CLP command options can be specified by setting the command line processor DB2OPTIONS environment variable (which must be in uppercase), or with command line flags.

Users can set options for an entire session using DB2OPTIONS.

View the current settings for the option flags and the value of DB2OPTIONS using LIST COMMAND OPTIONS. Change an option setting from the interactive input mode or a command file using UPDATE COMMAND OPTIONS.

The command line processor sets options in the following order:

1. Sets up default options.
2. Reads DB2OPTIONS to override the defaults.
3. Reads the command line to override DB2OPTIONS.
4. Accepts input from UPDATE COMMAND OPTIONS as a final interactive override.

Table 1 on page 249 summarizes the CLP option flags. These options can be specified in any sequence and combination. To turn an option on, prefix the corresponding option letter with a minus sign (-). To turn an option off, either prefix the option letter with a minus sign and follow the option letter with another minus sign, or prefix the option letter with a plus sign (+). For example, `-c` turns the auto-commit option on, and either `-c-` or `+c` turns it off. These option letters are not case sensitive, that is, `-a` and `-A` are equivalent.

*Table 1. CLP Command Options*

| Option Flag | Description | Default Setting |
|---|---|---|
| -a | This option tells the command line processor to display SQLCA data. | OFF |
| -c | This option tells the command line processor to automatically commit SQL statements. | ON |
| -e{c\|s} | This option tells the command line processor to display SQLCODE or SQLSTATE. These options are mutually exclusive. | OFF |
| -f*filename* | This option tells the command line processor to read command input from a file instead of from standard input. | OFF |
| -l*filename* | This option tells the command line processor to log commands in a history file. | OFF |
| -n | Removes the new line character within a single delimited token. If this option is not specified, the new line character is replaced with a space. This option must be used with the -t option. | OFF |
| -o | This option tells the command line processor to display output data and messages to standard output. | ON |
| -p | This option tells the command line processor to display a command line processor prompt when in interactive input mode. | ON |
| -r*filename* | This option tells the command line processor to write the report generated by a command to a file. | OFF |
| -s | This option tells the command line processor to stop execution if errors occur while executing commands in a batch file or in interactive mode. | OFF |
| -t | This option tells the command line processor to use a semicolon (;) as the statement termination character. | OFF |
| -td*x* | This option tells the command line processor to define and to use *x* as the statement termination character. | OFF |
| -v | This option tells the command line processor to echo command text to standard output. | OFF |
| -w | This option tells the command line processor to display SQL statement warning messages. | ON |
| -x | This option tells the command line processor to return data without any headers, including column names. | OFF |
| -z*filename* | This option tells the command line processor to redirect all output to a file. It is similar to the `-r` option, but includes any messages or error codes with the output. | OFF |

**Example**

The AIX command:

```
export DB2OPTIONS='+a -c +ec -o -p'
```

sets the following default settings for the session:

```
Display SQLCA   - off
Auto Commit     -  on
Display SQLCODE - off
Display Output  -  on
Display Prompt  -  on
```

The following is a detailed description of these options:

**Show SQLCA Data Option (-a):**
Displays SQLCA data to standard output after executing a DB2 command or an SQL statement. The SQLCA data is displayed instead of an error or success message.

The default setting for this command option is OFF (+a or -a-).

The -o and the -r options affect the -a option; see the option descriptions for details.

**Auto-commit Option (-c):**
This option specifies whether each command or statement is to be treated independently. If set ON (-c), each command or statement is automatically committed or rolled back. If the command or statement is successful, it and all successful commands and statements that were issued before it with autocommit OFF (+c or -c-) are committed. If, however, the command or statement fails, it and all successful commands and statements that were issued before it with autocommit OFF are rolled back. If set OFF (+c or -c-), COMMIT or ROLLBACK must be issued explicitly, or one of these actions will occur when the next command with autocommit ON (-c) is issued.

The default setting for this command option is ON.

The auto-commit option does not affect any other command line processor option.

**Example:** Consider the following scenario:

1. `db2 create database test`
2. `db2 connect to test`
3. `db2 +c "create table a (c1 int)"`
4. `db2 select c2 from a`

The SQL statement in step 4 fails because there is no column named C2 in table A. Since that statement was issued with auto-commit ON (default), it rolls back not only the statement in step 4, but also the one in step 3, because the latter was issued with auto-commit OFF. The command:

```
db2 list tables
```

then returns an empty list.

**Display SQLCODE/SQLSTATE Option (-e):**
The -e{c|s} option tells the command line processor to display the SQLCODE (-ec) or the SQLSTATE (-es) to standard output. Options -ec and -es are not valid in CLP interactive mode.

The default setting for this command option is OFF (+e or -e-).

The -o and the -r options affect the -e option; see the option descriptions for details.

The display SQLCODE/SQLSTATE option does not affect any other command line processor option.

**Example:** To retrieve SQLCODE from the command line processor running on AIX, enter:

```
sqlcode=`db2 –ec +o db2–command`
```

**Read from Input File Option (-f):**

The `-f`*filename* option tells the command line processor to read input from a specified file, instead of from standard input. *Filename* is an absolute or relative file name which can include the directory path to the file. If the directory path is not specified, the current directory is used.

When other options are combined with option `-f`, option `-f` must be specified last. For example:

```
db2 -tvf filename
```

**Note:** This option cannot be changed from within the interactive mode.

The default setting for this command option is OFF (`+f` or `-f-`).

Commands are processed until QUIT or TERMINATE is issued, or an end-of-file is encountered.

If both this option and a database command are specified, the command line processor does not process any commands, and an error message is returned.

Input file lines which begin with the comment characters `--` are treated as comments by the command line processor. Comment characters must be the first non-blank characters on a line.

If the `-f`*filename* option is specified, the `-p` option is ignored.

The read from input file option does not affect any other command line processor option.

**Log Commands in History File Option (-l):**

The `-l`*filename* option tells the command line processor to log commands to a specified file. This history file contains records of the commands executed and their completion status. *Filename* is an absolute or relative file name which can include the directory path to the file. If the directory path is not specified, the current directory is used. If the specified file or default file already exists, the new log entry is appended to that file.

When other options are combined with option `-l`, option `-l` must be specified last. For example:

```
db2 -tvl filename
```

The default setting for this command option is OFF (`+l` or `-l-`).

The log commands in history file option does not affect any other command line processor option.

**Remove New Line Character Option (-n):**

Removes the new line character within a single delimited token. If this option is not specified, the new line character is replaced with a space.

**Note:** This option cannot be changed from within the interactive mode.

The default setting for this command option is OFF (`+n` or `-n-`).

This option must be used with the `-t` option; see the option description for details.

**Display Output Option (-o):**

The `-o` option tells the command line processor to send output data and messages to standard output.

The default setting for this command option is ON.

The interactive mode start-up information is not affected by this option. Output data consists of report output from the execution of the user-specified command, and SQLCA data (if requested).

The following options might be affected by the `+o` option:

- `-r`*filename*: Interactive mode start-up information is not saved.
- `-e`: SQLCODE or SQLSTATE is displayed on standard output even if `+o` is specified.
- `-a`: No effect if `+o` is specified. If `-a`, `+o` and `-r`*filename* are specified, SQLCA information is written to a file.

If both `-o` and `-e` options are specified, the data and either the SQLCODE or the SQLSTATE are displayed on the screen.

If both `-o` and `-v` options are specified, the data is displayed, and the text of each command issued is echoed to the screen.

The display output option does not affect any other command line processor option.

**Display DB2 Interactive Prompt Option (-p):**

The `-p` option tells the command line processor to display the command line processor prompt when the user is in interactive mode.

The default setting for this command option is ON.

Turning the prompt off is useful when commands are being piped to the command line processor. For example, a file containing CLP commands could be executed by issuing:

```
db2 +p < myfile.clp
```

The `-p` option is ignored if the `-f`*filename* option is specified.

The display DB2 interactive prompt option does not affect any other command line processor option.

**Save to Report File Option (-r):**

The `-r`*filename* option causes any output data generated by a command to be written to a specified file, and is useful for capturing a report that would otherwise scroll off the screen. Messages or error codes are not written to the file. *Filename* is an absolute or relative file name which can include the directory path to the file. If the directory path is not specified, the current directory is used. New report entries are appended to the file.

The default setting for this command option is OFF (`+r` or `-r-`).

If the `-a` option is specified, SQLCA data is written to the file.

The `-r` option does not affect the `-e` option. If the `-e` option is specified, SQLCODE or SQLSTATE is written to standard output, not to a file.

If -r*filename* is set in DB2OPTIONS, the user can set the +r (or -r-) option from the command line to prevent output data for a particular command invocation from being written to the file.

The save to report file option does not affect any other command line processor option.

**Stop Execution on Command Error Option (-s):**

When commands are issued in interactive mode, or from an input file, and syntax or command errors occur, the -s option causes the command line processor to stop execution and to write error messages to standard output.

The default setting for this command option is OFF (+s or -s-). This setting causes the command line processor to display error messages, continue execution of the remaining commands, and to stop execution only if a system error occurs (return code 8).

The following table summarizes this behavior:

*Table 2. CLP Return Codes and Command Execution*

| Return Code | -s Option Set | +s Option Set |
|---|---|---|
| 0 (success) | execution continues | execution continues |
| 1 (0 rows selected) | execution continues | execution continues |
| 2 (warning) | execution continues | execution continues |
| 4 (DB2 or SQL error) | execution stops | execution continues |
| 8 (System error) | execution stops | execution stops |

**Statement Termination Character Option (-t):**

The -t option tells the command line processor to use a semicolon (;) as the statement termination character, and disables the backslash (\) line continuation character.

**Note:** This option cannot be changed from within the interactive mode.

The default setting for this command option is OFF (+t or -t-).

To define a termination character, use -td followed by the chosen termination character. For example, -tdx sets x as the statement termination character.

The termination character cannot be used to concatenate multiple statements from the command line, since only the last non-blank character on each input line is checked for a termination symbol.

The statement termination character option does not affect any other command line processor option.

**Verbose Output Option (-v):**

The -v option causes the command line processor to echo (to standard output) the command text entered by the user prior to displaying the output, and any messages from that command. ECHO is exempt from this option.

The default setting for this command option is OFF (+v or -v-).

The -v option has no effect if +o (or -o-) is specified.

The verbose output option does not affect any other command line processor option.

**Show Warning Messages Option (-w):**

The `-w` option tells the command line processor to show SQL statement warning messages.

The default setting for this command option is ON.

**Suppress Printing of Column Headings Option (-x):**

The `-x` option tells the command line processor to return data without any headers, including column names.

The default setting for this command option is OFF.

**Save all Output to File Option (-z):**

The `-z`*filename* option causes all output generated by a command to be written to a specified file, and is useful for capturing a report that would otherwise scroll off the screen. It is similar to the `-r` option; in this case, however, messages, error codes, and other informational output are also written to the file. *Filename* is an absolute or relative file name which can include the directory path to the file. If the directory path is not specified, the current directory is used. New report entries are appended to the file.

The default setting for this command option is OFF (`+z` or `-z-`).

If the `-a` option is specified, SQLCA data is written to the file.

The `-z` option does not affect the `-e` option. If the `-e` option is specified, SQLCODE or SQLSTATE is written to standard output, not to a file.

If `-z`*filename* is set in DB2OPTIONS, the user can set the `+z` (or `-z-`) option from the command line to prevent output data for a particular command invocation from being written to the file.

The save all output to file option does not affect any other command line processor option.

**Related reference:**

- "db2 - Command Line Processor Invocation" on page 247
- "Command Line Processor Return Codes" on page 254

## Command Line Processor Return Codes

When the command line processor finishes processing a command or an SQL statement, it returns a return (or exit) code. These codes are transparent to users executing CLP functions from the command line, but they can be retrieved when those functions are executed from a shell script.

For example, the following Bourne shell script executes the GET DATABASE MANAGER CONFIGURATION command, then inspects the CLP return code:

```
db2 get database manager configuration
if [ "$?" = "0" ]
then echo "OK!"
fi
```

The return code can be one of the following:

| **Code** | **Description** |
|---|---|
| **0** | DB2 command or SQL statement executed successfully |
| **1** | SELECT or FETCH statement returned no rows |
| **2** | DB2 command or SQL statement warning |

**4** DB2 command or SQL statement error

**8** Command line processor system error

The command line processor does not provide a return code while a user is executing statements from interactive mode, or while input is being read from a file (using the `-f` option).

A return code is available only after the user quits interactive mode, or when processing of an input file ends. In these cases, the return code is the logical OR of the distinct codes returned from the individual commands or statements executed to that point.

For example, if a user in interactive mode issues commands resulting in return codes of 0, 1, and 2, a return code of 3 will be returned after the user quits interactive mode. The individual codes 0, 1, and 2 are not returned. Return code 3 tells the user that during interactive mode processing, one or more commands returned a 1, and one or more commands returned a 2.

A return code of 4 results from a negative SQLCODE returned by a DB2 command or an SQL statement. A return code of 8 results only if the command line processor encounters a system error.

If commands are issued from an input file or in interactive mode, and the command line processor experiences a system error (return code 8), command execution is halted immediately. If one or more DB2 commands or SQL statements end in error (return code 4), command execution stops if the `-s` (Stop Execution on Command Error) option is set; otherwise, execution continues.

**Related reference:**

- "db2 - Command Line Processor Invocation" on page 247
- "Command line processor options" on page 248

## Command Line Processor (CLP)

The command line processor operates as follows:

- The CLP command (in either case) is typed at the command prompt.
- The command is sent to the command shell by pressing the ENTER key.
- Output is automatically directed to the standard output device.
- Piping and redirection are supported.
- The user is notified of successful and unsuccessful completion.
- Following execution of the command, control returns to the operating system command prompt, and the user can enter more commands.

Certain CLP commands and SQL statements require that the server instance is running and a database connection exists. Connect to a database by doing one of the following:

- Issue the SQL statement `DB2® CONNECT TO` *database*.
- Establish an implicit connection to the default database defined by the DB2 Universal Database™ (UDB) registry variable DB2DBDFT.

If a command exceeds the character limit allowed at the command prompt, a backslash (\) can be used as the line continuation character. When the command line processor encounters the line continuation character, it reads the next line and concatenates the characters contained on both lines. Alternatively, the `-t` option can be used to set a different line termination character.

The command line processor recognizes a string called `NULL` as a null string. Fields that have been set previously to some value can later be set to NULL. For example,

```
db2 update database manager configuration using tm_database NULL
```

sets the *tm_database* field to NULL. This operation is case sensitive. A lowercase `null` is not interpreted as a null string, but rather as a string containing the letters `null`.

**Customizing the Command Line Processor:**

It is possible to customize the interactive input prompt by using the DB2_CLPPROMPT registry variable. This registry variable can be set to any text string of maximum length 100 and can contain the tokens %i, %ia, %d, %da and %n. Specific values will be substituted for these tokens at run-time.

*Table 3. DB2_CLPPROMPT tokens and run-time values*

| DB2_CLPPROMPT token | Value at run-time |
|---|---|
| %ia | Authorization ID of the current instance attachment |
| %i | Local alias of the currently attached instance. If no instance attachment exists, the value of the DB2INSTANCE registry variable. On Windows® platforms only, if the DB2INSTANCE registry variable is not set, the value of the DB2INSTDEF registry variable. |
| %da | Authorization ID of the current database connection |
| %d | Local alias of the currently connected database. If no database connection exists, the value of the DB2DBDFT registry variable. |
| %n | New line |

- If any token has no associated value at run-time, the empty string is substituted for that token.
- The interactive input prompt will always present the authorization IDs, database names, and instance names in upper case, so as to be consistent with the connection and attachment information displayed at the prompt.
- If the DB2_CLPPROMPT registry variable is changed within CLP interactive mode, the new value of DB2_CLPPROMPT will not take effect until CLP interactive mode has been closed and reopened.

**Examples:**

If DB2_CLPPROMPT is defined as `(%ia@%i, %da@%d)`, the input prompt will have the following values:

- No instance attachment and no database connection. DB2INSTANCE set to ″DB2″. DB2DBDFT is not set.

  ```
  (@DB2, @)
  ```

- (Windows) No instance attachment and no database connection. DB2INSTANCE and DB2DBDFT not set. DB2INSTDEF set to ″DB2″.

  ```
  (@DB2, @)
  ```

- No instance attachment and no database connection. DB2INSTANCE set to ″DB2″. DB2DBDFT set to ″SAMPLE″.

  ```
  (@DB2, @SAMPLE)
  ```
- Instance attachment to instance ″DB2″ with authorization ID ″tyronnem″. DB2INSTANCE set to ″DB2″. DB2DBDFT set to ″SAMPLE″.

  ```
  (TYRONNEM@DB2, @SAMPLE)
  ```
- Database connection to database ″sample″ with authorization ID ″horman″. DB2INSTANCE set to ″DB2″. DB2DBDFT set to ″SAMPLE″.

  ```
  (@DB2, HORMAN@SAMPLE)
  ```
- Instance attachment to instance ″DB2″ with authorization ID ″tyronnem″. Database connection to database ″sample″ with authorization ID ″horman″. DB2INSTANCE set to ″DB2″. DB2DBDFT not set.

  ```
  (TYRONNEM@DB2, HORMAN@SAMPLE)
  ```

**Using the Command Line Processor in Command Files:**

CLP requests to the database manager can be imbedded in a shell script command file. The following example shows how to enter the CREATE TABLE statement in a shell script command file:

```
db2 "create table mytable (name VARCHAR(20), color CHAR(10))"
```

For more information about commands and command files, see the appropriate operating system manual.

**Command Line Processor Design:**

The command line processor consists of two processes: the front-end process (the DB2 command), which acts as the user interface, and the back-end process (db2bp), which maintains a database connection.

**Maintaining Database Connections**

Each time that **db2** is invoked, a new front-end process is started. The back-end process is started by the first **db2** invocation, and can be explicitly terminated with TERMINATE. All front-end processes with the same parent are serviced by a single back-end process, and therefore share a single database connection.

For example, the following **db2** calls from the same operating system command prompt result in separate front-end processes sharing a single back-end process, which holds a database connection throughout:

- db2 'connect to sample’,
- db2 'select * from org’,
- . foo (where foo is a shell script containing DB2 commands), and
- db2 -tf myfile.clp.

The following invocations from the same operating system prompt result in separate database connections because each has a distinct parent process, and therefore a distinct back-end process:

- foo
- . foo &
- foo &
- sh foo

### Communication between Front-end and Back-end Processes

The front-end process and back-end processes communicate through three message queues: a request queue, an input queue, and an output queue.

### Environment Variables

The following environment variables offer a means of configuring communication between the two processes:

*Table 4. Environment Variables*

| Variable | Minimum | Maximum | Default |
|---|---|---|---|
| DB2BQTIME | 1 second | 5294967295 | 1 second |
| DB2BQTRY | 0 tries | 5294967295 | 60 tries |
| DB2RQTIME | 1 second | 5294967295 | 5 seconds |
| DB2IQTIME | 1 second | 5294967295 | 5 seconds |

**DB2BQTIME**

When the command line processor is invoked, the front-end process checks if the back-end process is already active. If it is active, the front-end process reestablishes a connection to it. If it is not active, the front-end process activates it. The front-end process then idles for the duration specified by the DB2BQTIME variable, and checks again. The front-end process continues to check for the number of times specified by the DB2BQTRY variable, after which, if the back-end process is still not active, it times out and returns an error message.

**DB2BQTRY**

Works in conjunction with the DB2BQTIME variable, and specifies the number of times the front-end process tries to determine whether the back-end process is active.

The values of DB2BQTIME and DB2BQTRY can be increased during peak periods to optimize query time.

**DB2RQTIME**

Once the back-end process has been started, it waits on its request queue for a request from the front-end. It also waits on the request queue between requests initiated from the command prompt.

The DB2RQTIME variable specifies the length of time the back-end process waits for a request from the front-end process. At the end of this time, if no request is present on the request queue, the back-end process checks whether the parent of the front-end process still exists, and terminates itself if it does not exist. Otherwise, it continues to wait on the request queue.

**DB2IQTIME**

When the back-end process receives a request from the front-end process, it sends an acknowledgment to the front-end process indicating that it is ready to receive input via the input queue. The back-end process then waits on its input queue. It also waits on the input queue while a batch file (specified with the `-f` option) is executing, and while the user is in interactive mode.

The DB2IQTIME variable specifies the length of time the back-end process waits on the input queue for the front-end process to pass the commands. After this time has elapsed, the back-end process checks whether the front-end process is active, and returns to wait on the request queue if the front-end process no longer exists. Otherwise, the back-end process continues to wait for input from the front-end process.

To view the values of these environment variables, use LIST COMMAND OPTIONS.

The back-end environment variables inherit the values set by the front-end process at the time the back-end process is initiated. However, if the front-end environment variables are changed, the back-end process will not inherit these changes. The back-end process must first be terminated, and then restarted (by issuing the **db2** command) to inherit the changed values.

An example of when the back-end process must be terminated is provided by the following scenario:

1. User A logs on, issues some CLP commands, and then logs off without issuing TERMINATE.
2. User B logs on using the same window.
3. When user B issues certain CLP commands, they fail with message DB21016 (system error).

The back-end process started by user A is still active when user B starts using the CLP, because the parent of user B’s front-end process (the operating system window from which the commands are issued) is still active. The back-end process attempts to service the new commands issued by user B; however, user B’s front-end process does not have enough authority to use the message queues of the back-end process, because it needs the authority of user A, who created that back-end process. A CLP session must end with a TERMINATE command before a user starts a new CLP session using the same operating system window. This creates a fresh back-end process for each new user, preventing authority problems, and setting the correct values of environment variables (such as DB2INSTANCE) in the new user’s back-end process.

**CLP Usage Notes:**

Commands can be entered either in upper case or in lowercase from the command prompt. However, parameters that are case sensitive to DB2 must be entered in the exact case desired. For example, the *comment-string* in the WITH clause of the CHANGE DATABASE COMMENT command is a case sensitive parameter.

Delimited identifiers are allowed in SQL statements.

Special characters, or metacharacters (such as $ & * ( ) ; < > ? \ ' ") are allowed within CLP commands. If they are used outside the CLP interactive mode, or the CLP batch input mode, these characters are interpreted by the operating system shell. Quotation marks or an escape character are required if the shell is not to take any special action.

For example, when executed inside an AIX Korn shell environment,

```
db2 select * from org where division > 'Eastern'
```

is interpreted as ″select <the names of all files> from org where division″. The result, an SQL syntax error, is redirected to the file `Eastern`. The following syntax produces the correct output:

```
db2 "select * from org where division > 'Eastern'"
```

Special characters vary from platform to platform. In the AIX Korn shell, the above example could be rewritten using an escape character (\), such as *, \>, or \'.

Most operating system environments allow input and output to be redirected. For example, if a connection to the SAMPLE database has been made, the following request queries the STAFF table, and sends the output to a file named `staflist.txt` in the `mydata` directory:

```
db2 "select * from staff" > mydata/staflist.txt
```

For environments where output redirection is not supported, CLP options can be used. For example, the request can be rewritten as

```
db2 -r mydata\staflist.txt "select * from staff"

db2 -z mydata\staflist.txt "select * from staff"
```

The command line processor is not a programming language. For example, it does not support host variables, and the statement,

```
db2 connect to :HostVar in share mode
```

is syntactically incorrect, because `:HostVar` is not a valid database name.

The command line processor represents SQL NULL values as hyphens (-). If the column is numeric, the hyphen is placed at the right of the column. If the column is not numeric, the hyphen is at the left.

To correctly display the national characters for single byte (SBCS) languages from the DB2 command line processor window, a True Type font must be selected. For example, in a Windows environment, open the command window properties notebook and select a font such as Lucinda Console.

# Chapter 3. CLP Commands

This chapter describes the DB2 commands in alphabetical order. These commands are used to control the system interactively.

**Note:** Slashes (/) in directory paths are specific to UNIX-based systems, and are equivalent to back slashes (\) in directory paths on Windows operating systems.

## DB2 CLP Commands

The following table lists the CLP commands grouped by functional category:

*Table 5. DB2 CLP Commands*

| **CLP Session Control** |
|---|
| "LIST COMMAND OPTIONS" on page 482 |
| "UPDATE COMMAND OPTIONS" on page 726 |
| "CHANGE ISOLATION LEVEL" on page 329 |
| "SET RUNTIME DEGREE" on page 681 |
| "TERMINATE" on page 705 |
| "QUIT" on page 594 |
| **Database Manager Control** |
| "START DATABASE MANAGER" on page 690 |
| "STOP DATABASE MANAGER" on page 698 |
| "GET DATABASE MANAGER CONFIGURATION" on page 395 |
| "RESET DATABASE MANAGER CONFIGURATION" on page 641 |
| "UPDATE DATABASE MANAGER CONFIGURATION" on page 733 |
| "AUTOCONFIGURE" on page 277 |
| **Database Control** |
| "RESTART DATABASE" on page 645 |
| "CREATE DATABASE" on page 331 |
| "DROP DATABASE" on page 352 |
| "MIGRATE DATABASE" on page 556 |
| "ACTIVATE DATABASE" on page 265 |
| "DEACTIVATE DATABASE" on page 342 |
| "QUIESCE" on page 588 |
| "UNQUIESCE" on page 713 |
| "LIST INDOUBT TRANSACTIONS" on page 500 |
| "LIST DRDA INDOUBT TRANSACTIONS" on page 495 |
| "GET DATABASE CONFIGURATION" on page 389 |
| "RESET DATABASE CONFIGURATION" on page 639 |
| "UPDATE DATABASE CONFIGURATION" on page 730 |
| "AUTOCONFIGURE" on page 277 |

*Table 5. DB2 CLP Commands (continued)*

| **Database Directory Management** |
|---|
| “CATALOG DATABASE” on page 307 |
| “UNCATALOG DATABASE” on page 706 |
| “CATALOG DCS DATABASE” on page 311 |
| “UNCATALOG DCS DATABASE” on page 708 |
| “CHANGE DATABASE COMMENT” on page 327 |
| “LIST DATABASE DIRECTORY” on page 483 |
| “LIST DCS DIRECTORY” on page 493 |
| **ODBC Management** |
| “CATALOG ODBC DATA SOURCE” on page 323 |
| “LIST ODBC DATA SOURCES” on page 507 |
| “UNCATALOG ODBC DATA SOURCE” on page 712 |
| “GET CLI CONFIGURATION” on page 383 |
| “UPDATE CLI CONFIGURATION” on page 724 |
| **Client/Server Directory Management** |
| “CATALOG LOCAL NODE” on page 317 |
| “CATALOG NAMED PIPE NODE” on page 319 |
| “CATALOG APPC NODE” on page 303 |
| “CATALOG APPN NODE” on page 305 |
| “CATALOG NETBIOS NODE” on page 321 |
| “CATALOG TCPIP NODE” on page 324 |
| “UNCATALOG NODE” on page 711 |
| “LIST NODE DIRECTORY” on page 504 |
| **Network Support** |
| “REGISTER” on page 613 |
| “DEREGISTER” on page 344 |
| “UPDATE LDAP NODE” on page 738 |
| “CATALOG LDAP DATABASE” on page 313 |
| “UNCATALOG LDAP DATABASE” on page 709 |
| “CATALOG LDAP NODE” on page 316 |
| “UNCATALOG LDAP NODE” on page 710 |
| “REFRESH LDAP” on page 612 |
| **DB2 Administration Server** |
| “GET ADMIN CONFIGURATION” on page 374 |
| “RESET ADMIN CONFIGURATION” on page 635 |
| “UPDATE ADMIN CONFIGURATION” on page 715 |
| “CREATE TOOLS CATALOG” on page 339 |
| “DROP TOOLS CATALOG” on page 359 |
| **Recovery** |
| “ARCHIVE LOG” on page 273 |
| “BACKUP DATABASE” on page 280 |

*Table 5. DB2 CLP Commands (continued)*

| |
|---|
| "RECONCILE" on page 599 |
| "RESTORE DATABASE" on page 647 |
| "ROLLFORWARD DATABASE" on page 657 |
| "LIST HISTORY" on page 497 |
| "PRUNE HISTORY/LOGFILE" on page 584 |
| "UPDATE HISTORY FILE" on page 736 |
| "INITIALIZE TAPE" on page 472 |
| "REWIND TAPE" on page 656 |
| "SET TAPE POSITION" on page 685 |
| **Operational Utilities** |
| "FORCE APPLICATION" on page 372 |
| "LIST PACKAGES/TABLES" on page 508 |
| "REORGCHK" on page 624 |
| "REORG INDEXES/TABLE" on page 617 |
| "RUNSTATS" on page 667 |
| **Database Monitoring** |
| "GET MONITOR SWITCHES" on page 410 |
| "UPDATE MONITOR SWITCHES" on page 740 |
| "GET DATABASE MANAGER MONITOR SWITCHES" on page 400 |
| "GET SNAPSHOT" on page 419 |
| "RESET MONITOR" on page 643 |
| "INSPECT" on page 473 |
| "LIST ACTIVE DATABASES" on page 478 |
| "LIST APPLICATIONS" on page 480 |
| "LIST DCS APPLICATIONS" on page 490 |
| **Data Utilities** |
| "EXPORT" on page 362 |
| "IMPORT" on page 449 |
| "LOAD" on page 520 |
| "LOAD QUERY" on page 554 |
| **Health Center** |
| "ADD CONTACT" on page 267 |
| "ADD CONTACTGROUP" on page 268 |
| "DROP CONTACT" on page 350 |
| "DROP CONTACTGROUP" on page 351 |
| "GET ALERT CONFIGURATION" on page 376 |
| "GET CONTACTGROUP" on page 386 |
| "GET CONTACTGROUPS" on page 387 |
| "GET CONTACTS" on page 388 |
| "GET DESCRIPTION FOR HEALTH INDICATOR" on page 403 |
| "GET HEALTH NOTIFICATION CONTACT LIST" on page 405 |

*Table 5. DB2 CLP Commands (continued)*

| |
|---|
| "GET HEALTH SNAPSHOT" on page 406 |
| "GET RECOMMENDATIONS" on page 413 |
| "RESET ALERT CONFIGURATION" on page 637 |
| "UPDATE ALERT CONFIGURATION" on page 717 |
| "UPDATE CONTACT" on page 728 |
| "UPDATE CONTACTGROUP" on page 729 |
| "UPDATE HEALTH NOTIFICATION CONTACT LIST" on page 735 |
| **Application Preparation** |
| "PRECOMPILE" on page 560 |
| "BIND" on page 286 |
| "REBIND" on page 595 |
| **Remote Server Utilities** |
| "ATTACH" on page 275 |
| "DETACH" on page 349 |
| **Table Space Management** |
| "LIST TABLESPACE CONTAINERS" on page 511 |
| "SET TABLESPACE CONTAINERS" on page 683 |
| "LIST TABLESPACES" on page 513 |
| "QUIESCE TABLESPACES FOR TABLE" on page 591 |
| **Database Partition Management** |
| "ADD DBPARTITIONNUM" on page 271 |
| "DROP DBPARTITIONNUM VERIFY" on page 358 |
| "LIST DBPARTITIONNUMS" on page 489 |
| **Database Partition Group Management** |
| "LIST DATABASE PARTITION GROUPS" on page 486 |
| "REDISTRIBUTE DATABASE PARTITION GROUP" on page 609 |
| **Data Links** |
| "ADD DATALINKS MANAGER" on page 269 |
| "DROP DATALINKS MANAGER" on page 354 |
| "LIST DATALINKS MANAGERS" on page 488 |
| **Additional Commands** |
| "DESCRIBE" on page 345 |
| "ECHO" on page 360 |
| "GET AUTHORIZATIONS" on page 382 |
| "GET CONNECTION STATE" on page 385 |
| "GET INSTANCE" on page 409 |
| "GET ROUTINE" on page 417 |
| "HELP" on page 447 |
| "PING" on page 558 |
| "PUT ROUTINE" on page 586 |
| "QUERY CLIENT" on page 587 |

*Table 5. DB2 CLP Commands (continued)*

| |
|---|
| "SET CLIENT" on page 678 |

# ACTIVATE DATABASE

Activates the specified database and starts up all necessary database services, so that the database is available for connection and use by any application.

**Scope:**

This command activates the specified database on all nodes within the system. If one or more of these nodes encounters an error during activation of the database, a warning is returned. The database remains activated on all nodes on which the command has succeeded.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

None

**Command syntax:**

```
►►──ACTIVATE──┬─DATABASE─┬──database-alias──────────────────────────────►
              └─DB───────┘

►──┬──────────────────────────────────────┬──────────────────────────►◄
   └─USER──username──┬──────────────────┬─┘
                     └─USING──password──┘
```

**Command parameters:**

**database-alias**
Specifies the alias of the database to be started.

**USER username**
Specifies the user starting the database.

**USING password**
Specifies the password for the user name.

**Usage notes:**

If a database has not been started, and a CONNECT TO (or an implicit connect) is issued in an application, the application must wait while the database manager starts the required database, before it can do any work with that database. However, once the database is started, other applications can simply connect and use it without spending time on its start up.

Database administrators can use ACTIVATE DATABASE to start up selected databases. This eliminates any application time spent on database initialization.

Databases initialized by ACTIVATE DATABASE can be shut down using the DEACTIVATE DATABASE command, or using the db2stop command.

If a database was started by a CONNECT TO (or an implicit connect) and subsequently an ACTIVATE DATABASE is issued for that same database, then DEACTIVATE DATABASE must be used to shut down that database. If ACTIVATE DATABASE was not used to start the database, the database will shut down when the last application disconnects.

ACTIVATE DATABASE behaves in a similar manner to a CONNECT TO (or an implicit connect) when working with a database requiring a restart (for example, database in an inconsistent state). The database will be restarted before it can be initialized by ACTIVATE DATABASE. Restart will only be performed if the database is configured to have AUTORESTART ON.

**Note:** The application issuing the ACTIVATE DATABASE command cannot have an active database connection to any database.

**Related reference:**

- "STOP DATABASE MANAGER" on page 698
- "DEACTIVATE DATABASE" on page 342

# ADD CONTACT

The command adds a contact to the contact list which can be either defined locally on the system or in a global list. Contacts are users to whom processes such as the Scheduler and Health Monitor send messages. The setting of the Database Administration Server (DAS) *contact_host* configuration parameter determines whether the list is local or global.

**Authorization:**

None.

**Required connection:**

None. Local execution only: this command cannot be used with a remote connection.

**Command syntax:**

```
►►──ADD CONTACT──name──TYPE──┬─EMAIL──────────────────────────────────────────┬──────►
                             └─PAGE──┬──────────────────────────────────────┬─┘
                                     └─┬─MAXIMUM PAGE LENGTH─┬──pg-length─┘
                                       └─MAX LEN─────────────┘

►─ADDRESS──recipients address──┬──────────────────────────────────────┬──────────────►◄
                               └─DESCRIPTION──contact description─────┘
```

**Command parameters:**

**CONTACT name**
The name of the contact that will be added. By default the contact will be added in the local system, unless the DB2 administration server configuration parameter *contact_host* points to another system.

**TYPE** Method of contact, which must be one of the following two:

**EMAIL**
This contact wishes to be notified by e-mail at (ADDRESS).

**PAGE** This contact wishes to be notified by a page sent to ADDRESS.

**MAXIMUM PAGE LENGTH pg-length**
If the paging service has a message-length restriction, it is specified here in characters.

**Note:** The notification system uses the SMTP protocol to send the notification to the mail server specified by the DB2 Administration Server configuration parameter *smtp_server*. It is the responsibility of the SMTP server to send the e-mail or call the pager.

**ADDRESS recipients-address**
The SMTP mailbox address of the recipient. For example, joe@somewhere.org. The *smtp_server* DAS configuration parameter must be set to the name of the SMTP server.

**DESCRIPTION contact description**
A textual description of the contact. This has a maximum length of 128 characters.

## ADD CONTACTGROUP

Adds a new contact group to the list of groups defined on the local system. A contact group is a list of users and groups to whom monitoring processes such as the Scheduler and Health Monitor can send messages.

**Authorization:**

None

**Required Connection:**

None. Local execution only: this command cannot be used with a remote connection.

**Command Syntax:**

```
                                   ┌─,───────────────────┐
                                   ▼                     │
►►──ADD CONTACTGROUP──name─────────┬─CONTACT─┬──name─────┴──────────────────────────►
                                   └─GROUP───┘

►──┬────────────────────────────────┬──────────────────────────────────────────────►◄
   └─DESCRIPTION──group description─┘
```

**Command Parameters:**

**CONTACTGROUP** *name*
: Name of the new contact group, which must be unique among the set of groups on the system.

**CONTACT** *name*
: Name of the contact which is a member of the group. You do not need to define an individual contact before you include that contact in a group.

**GROUP** *name*
: Name of the contact group of which this group is a member.

**DESCRIPTION** *group description*
: Optional. A textual description of the contact group.

# ADD DATALINKS MANAGER

Adds a DB2 Data Links Manager to the list of registered DB2 Data Links Managers for a specified database.

**Authorization:**

One of the following

- *sysadm*
- *sysctrl*
- *sysmaint*

**Command syntax:**

```
►►──ADD DATALINKS MANAGER FOR──┬─DATABASE─┬──dbname──USING──────────────────────►
                               └─DB───────┘

►─NODE──hostname──PORT──port-number──────────────────────────────────────────►◄
```

**Command parameters:**

**DATABASE dbname**
: Specifies a database name.

**USING NODE hostname**
: Specifies a fully qualified host name, or the IP address (but not both), of the DB2 Data Links Manager server.

**PORT port-number**
: Specifies the port number that has been reserved for communications from the DB2 server to the DB2 Data Links Manager server.

**Usage notes:**

This command is effective only after all applications have been disconnected from the database. The DB2 Data Links Manager being added must be completely set up and running for this command to be successful. The database must also be registered on the DB2 Data Links Manager using the `dlfm add_db` command. The maximum number of DB2 Data Links Managers that can be added to a database is 16.

A Data Links Manager added by specifying USING NODE is said to be of type ″Native″. All Data Links Managers registered to a database must be of the same type.

When registering one or more DB2 Data Links Managers for a database using this command, ensure that the DB2 Data Links Manager is not registered twice; otherwise, error SQL20056N with reason code ″99″ might be returned during processing. The `db2diag.log` file for the DB2 Data Links Manager server that is registered twice will have the following entry when such a failure occurs:

```
dfm_xnstate_cache_insert : Duplicate txn entry.
dfmBeginTxn : Unable to insert ACTIVE transaction in cache, rc = 41.
DLFM501E : Transaction management service failed.
```

**Note:** The Command Line Processor detects errors if duplicate Data Links Managers are added using the same name or address. However, duplicates are not detected if a Data Links Manager is added more than once using a different IP name or address. For example, if a Data Links Manager was added twice, once using the name `dln1.almaden.ibm.com`and again using the short name `dln1`, the failure described above is possible.

**Related reference:**

- "LIST DATALINKS MANAGERS" on page 488
- "DROP DATALINKS MANAGER" on page 354

# ADD DBPARTITIONNUM

Adds a new database partition server to the partitioned database environment. This command also creates a database partition for all databases on the new database partition server. The user can specify the source database partition server for the definitions of any system temporary table spaces to be created with the new database partition, or specify that no system temporary table spaces are to be created. The command must be issued from the database partition server that is being added.

**Scope:**

This command only affects the machine on which it is executed.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None

**Command syntax:**

```
►►──ADD DBPARTITIONNUM──┬──────────────────────────────────────────┬──────────►◄
                        ├─LIKE DBPARTITIONNUM──db-partition-number─┤
                        └─WITHOUT TABLESPACES──────────────────────┘
```

**Command parameters:**

**LIKE DBPARTITIONNUM** *db-partition-number*
: Specifies that the containers for the new system temporary table spaces are the same as the containers of the database at the database partition server specified by *db-partition-number*. The database partition server specified must already be defined in the `db2nodes.cfg` file.

**WITHOUT TABLESPACES**
: Specifies that containers for the system temporary table spaces are not created for any of the database partitions. The ALTER TABLESPACE statement must be used to add system temporary table space containers to each database partition before the database can be used.

**Note:** If no option is specified, containers for the system temporary table spaces will be the same as the containers on the catalog partition for each database. The catalog partition can be a different database partition for each database in the partitioned environment.

**Usage notes:**

Before adding a new database partition server, ensure that there is sufficient storage for the containers that must be created for all databases in the instance.

The add database partition server operation creates an empty database partition for every database that exists in the instance. The configuration parameters for the new database partitions are set to the default values.

If an add database partition server operation fails while creating a database partition locally, it enters a clean-up phase, in which it locally drops all databases that have been created. This means that the database partitions are removed only from the database partition server being added. Existing database partitions remain unaffected on all other database partition servers. If the clean-up phase fails, no further clean up is done, and an error is returned.

The database partitions on the new database partition cannot contain user data until after the ALTER DATABASE PARTITION GROUP statement has been used to add the database partition to a database partition group.

This command will fail if a create database or a drop database operation is in progress. The command can be reissued once the competing operation has completed.

If system temporary table spaces are to be created with the database partitions, ADD DBPARTITIONNUM might have to communicate with another database partition server to retrieve the table space definitions for the database partitions that reside on that server. The *start_stop_time* database manager configuration parameter is used to specify the time, in minutes, by which the other database partition server must respond with the table space definitions. If this time is exceeded, the command fails. If this situation occurs, increase the value of *start_stop_time*, and reissue the command.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "START DATABASE MANAGER" on page 690

# ARCHIVE LOG

Closes and truncates the active log file for a recoverable database.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

None. This command establishes a database connection for the duration of the command.

**Command syntax:**

```
►►──ARCHIVE LOG FOR──┬─DATABASE─┬──database-alias──────────────────────────►
                     └─DB───────┘

►──┬──────────────────────────────────────────┬─────────────────────────────►
   └─USER──username──┬─────────────────────┬──┘
                     └─USING──password─────┘

►──┬───────────────────────────────────────┬───────────────────────────────►◄
   └─┤ On Database Partition Number Clause ├─┘
```

**On Database Partition Number Clause:**

```
├──ON──┬─┤ Database Partition Number List Clause ├───────────────────────┬───┤
       └─ALL DBPARTITIONNUMS──┬────────────────────────────────────────────┬─┘
                              └─EXCEPT──┤ Database Partition Number List Clause ├─┘
```

**Database Partition Number List Clause:**

```
├──┬─DBPARTITIONNUM──┬──────────────────────────────────────────────────────►
   └─DBPARTITIONNUMS─┘

       ┌─,──────────────────────────────────────────────────┐
       ▼                                                    │
►──(───db-partition-number──┬────────────────────────────┬──┴──)────────────┤
                            └─TO──db-partition-number────┘
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database whose active log is to be archived.

**USER username**
Identifies the user name under which a connection will be attempted.

**USING password**
Specifies the password to authenticate the user name.

**ON ALL DBPARTITIONNUMS**
Specifies that the command should be issued on all database partitions in the `db2nodes.cfg` file. This is the default if a database partition number clause is not specified.

**EXCEPT**
Specifies that the command should be issued on all database partitions in the `db2nodes.cfg` file, except those specified in the database partition number list.

**ON DBPARTITIONNUM/ON DBPARTITIONNUMS**
Specifies that the logs should be archived for the specified database on a set of database partitions.

**db-partition-number**
Specifies a database partition number in the database partition number list.

**TO db-partition-number**
Used when specifying a range of database partitions for which the logs should be archived. All database partitions from the first database partition number specified up to and including the second database partition number specified are included in the database partition number list.

**Usage notes:**

This command can be used to collect a complete set of log files up to a known point. The log files can then be used to update a standby database.

This command can only be executed when the invoking application or shell does not have a database connection to the specified database. This prevents a user from executing the command with uncommitted transactions. As such, the ARCHIVE LOG command will not forcibly commit the user's incomplete transactions. If the invoking application or shell already has a database connection to the specified database, the command will terminate and return an error. If another application has transactions in progress with the specified database when this command is executed, there will be a slight performance degradation since the command flushes the log buffer to disk. Any other transactions attempting to write log records to the buffer will have to wait until the flush is complete.

If used in a partitioned database environment, a subset of database partitions can be specified by using a database partition number clause. If the database partition number clause is not specified, the default behavior for this command is to close and archive the active log on all database partitions.

Using this command will use up a portion of the active log space due to the truncation of the active log file. The active log space will resume its previous size when the truncated log becomes inactive. Frequent use of this command can drastically reduce the amount of the active log space available for transactions.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.
- The keyword NODES can be substituted for DBPARTITIONNUMS.

# ATTACH

Enables an application to specify the instance at which instance-level commands (CREATE DATABASE and FORCE APPLICATION, for example) are to be executed. This instance can be the current instance, another instance on the same workstation, or an instance on a remote workstation.

**Authorization:**

None

**Required connection:**

None. This command establishes an instance attachment.

**Command syntax:**

```
►►──ATTACH──┬────────────────┬───────────────────────────────────────────────►
            └─TO──nodename──┘

►──┬────────────────────────────────────────────────────────────────────┬──►◄
   └─USER──username──┬───────────────────────────────────────────────┬─┘
                     ├─USING──password──┬──────────────────────────────────┬─┤
                     │                  └─NEW──password──CONFIRM──password─┘ │
                     └─CHANGE PASSWORD──────────────────────────────────────┘
```

**Command parameters:**

**TO nodename**
Alias of the instance to which the user wants to attach. This instance must have a matching entry in the local node directory. The only exception to this is the local instance (as specified by the **DB2INSTANCE** environment variable) which can be specified as the object of an attach, but which cannot be used as a node name in the node directory.

**USER username**
Specifies the authentication identifier. When attaching to a DB2 Universal Database (UDB) instance on a Windows operating system, the user name can be specified in a format compatible with Microsoft Windows NT Security Account Manager (SAM), for example, *domainname\username*.

**USING password**
Specifies the password for the user name. If a user name is specified, but a password is *not* specified, the user is prompted for the current password. The password is not displayed at entry.

**NEW password**
Specifies the new password that is to be assigned to the user name. Passwords can be up to 18 characters in length. The system on which the password will be changed depends on how user authentication has been set up.

**CONFIRM password**
A string that must be identical to the new password. This parameter is used to catch entry errors.

**CHANGE PASSWORD**
If this option is specified, the user is prompted for the current password, a new password, and for confirmation of the new password. Passwords are not displayed at entry.

**Examples:**

Catalog two remote nodes:

```
db2 catalog tcpip node node1 remote freedom server server1
db2 catalog tcpip node node2 remote flash server server1
```

Attach to the first node, force all users, and then detach:

```
db2 attach to node1
db2 force application all
db2 detach
```

Attach to the second node, and see who is on:

```
db2 attach to node2
db2 list applications
```

After the command returns agent IDs 1, 2 and 3, force 1 and 3, and then detach:

```
db2 force application (1, 3)
db2 detach
```

Attach to the current instance (not necessary, will be implicit), force all users, then detach (AIX only):

```
db2 attach to $DB2INSTANCE
db2 force application all
db2 detach
```

**Usage notes:**

If *nodename* is omitted from the command, information about the current state of attachment is returned.

If ATTACH has not been executed, instance-level commands are executed against the current instance, specified by the **DB2INSTANCE** environment variable.

**Related reference:**

- "DETACH" on page 349

# AUTOCONFIGURE

Calculates and displays initial values for the buffer pool size, database configuration and database manager configuration parameters, with the option of applying these recommended values.

**Authorization:**

*sysadm*.

**Required connection:**

Database.

**Command syntax:**

```
>>-AUTOCONFIGURE--+---------------------------------------+------>
                  |        .---------------------------.  |
                  |        V                           |  |
                  '-USING----input-keyword--param-value-+-'

>-APPLY--+-DB ONLY----+----------------------------------------><
         +-DB AND DBM-+
         '-NONE-------'
```

**Command parameters:**

**USING input-keyword param-value**

*Table 6. Valid input keywords and parameter values*

| Keyword | Valid values | Default value | Explanation |
|---|---|---|---|
| mem_percent | 1–100 | 80 | Percentage of memory to dedicate. If other applications (other than the operating system) are running on this server, set this to less than 100. |
| workload_type | simple, mixed, complex | mixed | Simple workloads tend to be I/O intensive and mostly transactions, whereas complex workloads tend to be CPU intensive and mostly queries. |
| num_stmts | 1–1 000 000 | 10 | Number of statements per unit of work |
| tpm | 1–200 000 | 60 | Transactions per minute |

*Table 6. Valid input keywords and parameter values (continued)*

| Keyword | Valid values | Default value | Explanation |
|---|---|---|---|
| admin_priority | performance, recovery, both | both | Optimize for better performance (more transactions per minute) or better recovery time |
| is_populated | yes, no | yes | Is the database populated with data? |
| num_local_apps | 0–5 000 | 0 | Number of connected local applications |
| num_remote_apps | 0–5 000 | 10 | Number of connected remote applications |
| isolation | RR, RS, CS, UR | RR | Isolation level of applications connecting to this database (Repeatable Read, Read Stability, Cursor Stability, Uncommitted Read) |
| bp_resizeable | yes, no | yes | Are buffer pools resizeable? |

**APPLY**

**DB ONLY**
Displays the recommended values for the database configuration and the buffer pool settings based on the current database manager configuration. Applies the recommended changes to the database configuration and the buffer pool settings.

**DB AND DBM**
Displays and applies the recommended changes to the database manager configuration, the database configuration, and the buffer pool settings.

**NONE**
Displays the recommended changes, but does not apply them.

**Usage notes:**

If any of the input-keywords are not specified, the default value will be used for that parameter.

In a partitioned database environment, this command only applies changes to the current partition.

On systems with multiple logical partitions, the *mem_percent* parameter refers to the percentage of memory that is to be used by all logical partitions. For example, if DB2 uses 80% of the memory on the system, specify 80% regardless of the number of logical partitions. The database configuration recommendations made, however, will be adjusted for one logical partition.

This command makes configuration recommendations for the currently connected database, assuming that the database is the only active database on the system. If more than one database is active on the system, adjust the *mem_percent* parameter to reflect the current database's share of memory. For example, if DB2 uses 80% of the system's memory and there are two active databases on the system that should share the resources equally, specify 40% (80% divided by 2 databases) for the parameter *mem_percent*.

# BACKUP DATABASE

Creates a backup copy of a database or a table space.

**Scope:**

This command only affects the database partition on which it is executed.

**Authorization:**

One of the following:

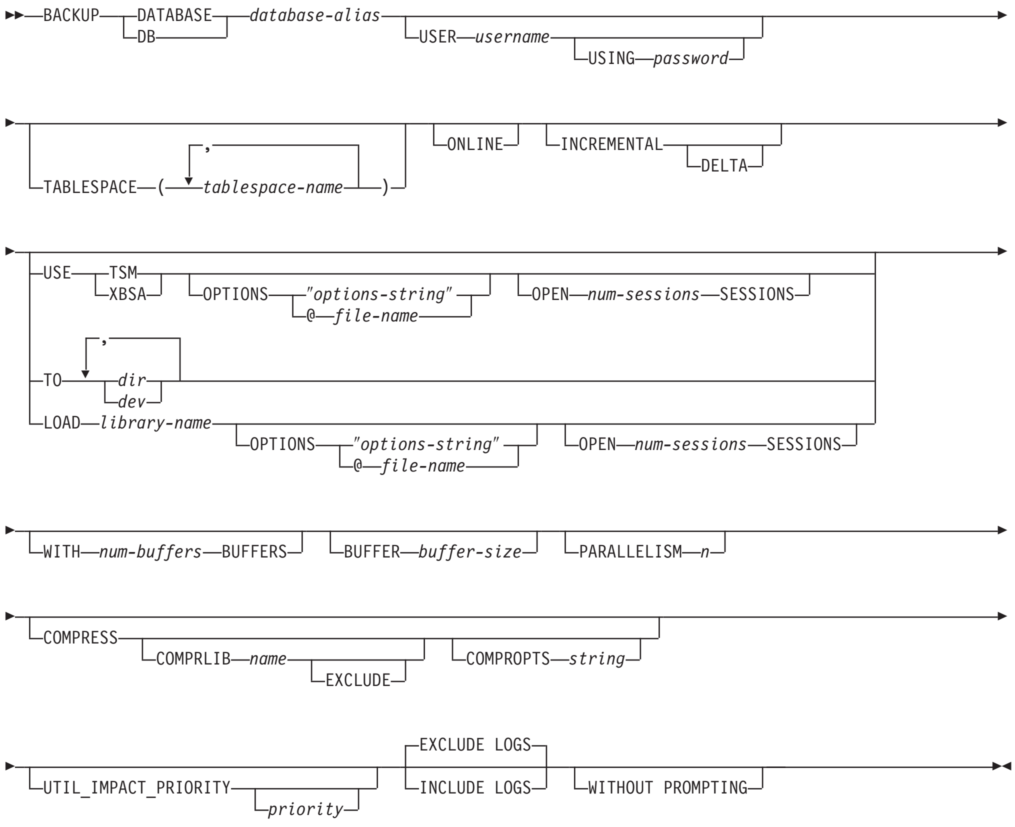
- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Database. This command automatically establishes a connection to the specified database.

**Note:** If a connection to the specified database already exists, that connection will be terminated and a new connection established specifically for the backup operation. The connection is terminated at the completion of the backup operation.

**Command syntax:**



**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database to back up.

**USER username**
Identifies the user name under which to back up the database.

**USING password**
The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

**TABLESPACE tablespace-name**
A list of names used to specify the table spaces to be backed up.

**ONLINE**
Specifies online backup. The default is offline backup. Online backups are only available for databases configured with *logretain* or *userexit* enabled. During an online backup, DB2 obtains IN (Intent None) locks on all tables existing in SMS table spaces as they are processed and S (Share) locks on LOB data in SMS table spaces.

**INCREMENTAL**
Specifies a cumulative (incremental) backup image. An incremental backup image is a copy of all database data that has changed since the most recent successful, full backup operation.

**DELTA**
Specifies a non-cumulative (delta) backup image. A delta backup image is a copy of all database data that has changed since the most recent successful backup operation of any type.

**USE TSM**
Specifies that the backup is to use Tivoli Storage Manager output.

**USE XBSA**
Specifies that the XBSA interface is to be used. Backup Services APIs (XBSA) are an open application programming interface for applications or facilities needing data storage management for backup or archiving purposes.

**OPTIONS**

*"options-string"*
Specifies options to be used for the backup operation.The string will be passed to the vendor support library, for example TSM, exactly as it was entered, without the quotes.

**Note:** Specifying this option overrides the value specified by the VENDOROPT database configuration parameter.

*@file-name*
Specifies that the options to be used for the backup operation are contained in a file located on the DB2 server. The string will be passed to the vendor support library, for example TSM. The file must be a fully qualified file name.

**OPEN num-sessions SESSIONS**
The number of I/O sessions to be created between DB2 and TSM or another backup vendor product.

**Note:** This parameter has no effect when backing up to tape, disk, or other local device.

**TO dir/dev**

A list of directory or tape device names. The full path on which the directory resides must be specified. If USE TSM, TO, and LOAD are omitted, the default target directory for the backup image is the current working directory of the client computer. This target directory or device must exist on the database server. This parameter can be repeated to specify the target directories and devices that the backup image will span. If more than one target is specified (target1, target2, and target3, for example), target1 will be opened first. The media header and special files (including the configuration file, table space table, and history file) are placed in target1. All remaining targets are opened, and are then used in parallel during the backup operation. Because there is no general tape support on Windows operating systems, each type of tape device requires a unique device driver. To back up to the FAT file system on Windows operating systems, users must conform to the 8.3 naming restriction.

Use of tape devices or floppy disks might generate messages and prompts for user input. Valid response options are:

**c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)

**d** Device terminate. Stop using *only* the device that generated the warning message (for example, when there are no more tapes)

**t** Terminate. Abort the backup operation.

If the tape system does not support the ability to uniquely reference a backup image, it is recommended that multiple backup copies of the same database not be kept on the same tape.

**LOAD library-name**

The name of the shared library (DLL on Windows operating systems) containing the vendor backup and restore I/O functions to be used. It can contain the full path. If the full path is not given, it will default to the path on which the user exit program resides.

**WITH num-buffers BUFFERS**

The number of buffers to be used. DB2 will automatically choose an optimal value for this parameter unless you explicitly enter a value. However, when creating a backup to multiple locations, a larger number of buffers can be used to improve performance.

**BUFFER buffer-size**

The size, in 4 KB pages, of the buffer used when building the backup image. DB2 will automatically choose an optimal value for this parameter unless you explicitly enter a value. The minimum value for this parameter is 8 pages.

If using tape with variable block size, reduce the buffer size to within the range that the tape device supports. Otherwise, the backup operation might succeed, but the resulting image might not be recoverable.

When using tape devices on SCO UnixWare 7, specify a buffer size of 16.

With most versions of Linux, using DB2's default buffer size for backup operations to a SCSI tape device results in error SQL2025N, reason code 75. To prevent the overflow of Linux internal SCSI buffers, use this formula:

```
bufferpages <= ST_MAX_BUFFERS * ST_BUFFER_BLOCKS / 4
```

where *bufferpages* is the value you want to use with the BUFFER parameter, and `ST_MAX_BUFFERS` and `ST_BUFFER_BLOCKS` are defined in the Linux kernel under the `drivers/scsi` directory.

**PARALLELISM n**

Determines the number of table spaces which can be read in parallel by the backup utility. DB2 will automatically choose an optimal value for this parameter unless you explicitly enter a value.

**UTIL_IMPACT_PRIORITY** *priority*

Specifies that the backup will run in throttled mode, with the priority specified. Throttling allows you to regulate the performance impact of the backup operation. Priority can be any number between 1 and 100, with 1 representing the lowest priority, and 100 representing the highest priority. If the UTIL_IMPACT_PRIORITY keyword is specified with no priority, the backup will run with the default priority of 50. If UTIL_IMPACT_PRIORITY is not specified, the backup will run in unthrottled mode. An impact policy must be defined by setting the *util_impact_lim* configuration parameter for a backup to run in throttled mode.

**COMPRESS**

Indicates that the backup is to be compressed.

**COMPRLIB** *name*

Indicates the name of the library to be used to perform the compression. The name must be a fully qualified path referring to a file on the server. If this parameter is not specified, the default DB2 compression library will be used. If the specified library cannot be loaded, the backup will fail.

**EXCLUDE**

Indicates that the compression library will not be stored in the backup image.

**COMPROPTS** *string*

Describes a block of binary data that will be passed to the initialization routine in the compression library. DB2 will pass this string directly from the client to the server, so any issues of byte reversal or code page conversion will have to be handled by the compression library. If the first character of the data block is ’@’, the remainder of the data will be interpreted by DB2 as the name of a file residing on the server. DB2 will then replace the contents of string with the contents of this file and will pass this new value to the initialization routine instead. The maximum length for *string* is 1024 bytes.

**EXCLUDE LOGS**

Specifies that the backup image should not include any log files.

**Note:** When performing an offline backup operation, logs are excluded whether or not this option is specified.

**INCLUDE LOGS**

Specifies that the backup image should include the range of log files required to restore and roll forward this image to some consistent point in time. This option is not valid for an offline backup.

**WITHOUT PROMPTING**
Specifies that the backup will run unattended, and that any actions which normally require user intervention will return an error message.

**Examples:**

1. In the following example, the database WSDB is defined on all 4 partitions, numbered 0 through 3. The path /dev3/backup is accessible from all partitions. Partition 0 is the catalog partition, and needs to be backed-up separately since this is an offline backup. To perform an offline backup of all the WSDB database partitions to /dev3/backup, issue the following commands from one of the database partitions:

   ```
   db2_all '<<+0< db2 BACKUP DATABASE wsdb TO /dev3/backup'
   db2_all '|<<-0< db2 BACKUP DATABASE wsdb TO /dev3/backup'
   ```

   In the second command, the db2_all utility will issue the same backup command to each database partition in turn (except partition 0). All four database partition backup images will be stored in the /dev3/backup directory.
2. In the following example database SAMPLE is backed up to a TSM server using two concurrent TSM client sessions. DB2 calculates the optimal buffer size for this environment.

   ```
   db2 backup database sample use tsm open 2 sessions with 4 buffers
   ```
3. In the following example, a table space-level backup of table spaces (syscatspace, userspace1) of database payroll is done to tapes.

   ```
   db2 backup database payroll tablespace (syscatspace, userspace1) to
     /dev/rmt0, /dev/rmt1 with 8 buffers without prompting
   ```
4. The USE TSM OPTIONS keywords can be used to specify the TSM information to use for the backup operation. The following example shows how to use the USE TSM OPTIONS keywords to specify a fully qualified file name:

   ```
   db2 backup db sample use TSM options @/u/dmcinnis/myoptions.txt
   ```

   The file myoptions.txt contains the following information: -fromnode=bar -fromowner=dmcinnis
5. Following is a sample weekly incremental backup strategy for a recoverable database. It includes a weekly full database backup operation, a daily non-cumulative (delta) backup operation, and a mid-week cumulative (incremental) backup operation:

   ```
   (Sun) db2 backup db sample use tsm
   (Mon) db2 backup db sample online incremental delta use tsm
   (Tue) db2 backup db sample online incremental delta use tsm
   (Wed) db2 backup db sample online incremental use tsm
   (Thu) db2 backup db sample online incremental delta use tsm
   (Fri) db2 backup db sample online incremental delta use tsm
   (Sat) db2 backup db sample online incremental use tsm
   ```
6. In the following example, three identical target directories are specified for a backup operation on database SAMPLE. You might want to do this if the target file system is made up of multiple physical disks.

   ```
   db2 backup database sample to /dev3/backup, /dev3/backup, /dev3/backup
   ```

   The data will be concurrently backed up to the three target directories, and three backup images will be generated with extensions .001, .002, and .003.

**Related reference:**

- "RESTORE DATABASE" on page 647

- “ROLLFORWARD DATABASE” on page 657

# BIND

Invokes the bind utility, which prepares SQL statements stored in the bind file generated by the precompiler, and creates a package that is stored in the database.

**Scope:**

This command can be issued from any database partition in `db2nodes.cfg`. It updates the database catalogs on the catalog database partition. Its effects are visible to all database partitions.

**Authorization:**

One of the following:
- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist and one of:
  - IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
  - CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

**For DB2 for Windows and UNIX**

```
►►──BIND──filename──────────────────────────────────────────────────────►

►──┬─────────────────────────────────────────────────────────────────┬──►
   └─ACTION───┬─ADD──────────────────────────────────────────────────┤
              │ ┌─REPLACE─┐                                        │
              └─┴─────────┴──┬──────────────────┬──┬─────────────────────┬─┘
                             └─RETAIN───┬─NO──┬─┘  └─REPLVER──version-id─┘
                                        └─YES─┘

►──┬────────────────────────────┬──┬────────────────────────┬──┬─────────────────────────┬──►
   └─BLOCKING───┬─UNAMBIG─┬─┘  └─CLIPKG──cli-packages─┘  └─COLLECTION──schema-name─┘
                ├─ALL─────┤
                └─NO──────┘
```

```
►─┬────────────────────┬─┬───────────────────────────────────────┬──────────►
  └─DATETIME──┬─DEF─┬──┘ └─DEGREE──┬─1─────────────────────┬─────┘
              ├─EUR─┤              ├─degree-of-parallelism─┤
              ├─ISO─┤              └─ANY───────────────────┘
              ├─JIS─┤
              ├─LOC─┤
              └─USA─┘

►─┬────────────────────────────┬─┬──────────────────────┬─┬─────────────────────┬──►
  └─DYNAMICRULES──┬─RUN────────┤ └─EXPLAIN──┬─NO────┬───┘ └─EXPLSNAP──┬─NO────┬─┘
                  ├─BIND───────┤            ├─ALL───┤                 ├─ALL───┤
                  ├─INVOKERUN──┤            ├─REOPT─┤                 ├─REOPT─┤
                  ├─INVOKEBIND─┤            └─YES───┘                 └─YES───┘
                  ├─DEFINERUN──┤
                  └─DEFINEBIND─┘

►─┬─────────────────────┬─┬──────────────────────────────────┬─┬────────────────────┬──►
  └─FEDERATED──┬─NO──┬──┘ │              ┌─,───────────┐     │ └─GENERIC──string──┘
               └─YES─┘    │              ▼             │     │
                          └─FUNCPATH─────schema-name───┴─────┘

►─┬─────────────────────────────┬─┬───────────────────────┬─┬──────────────────────┬──►
  ├─GRANT──┬─authid─┬───────────┤ └─INSERT──┬─DEF─┬───────┘ └─ISOLATION──┬─CS─┬────┘
  │        └─PUBLIC─┘           │           └─BUF─┘                      ├─RR─┤
  ├─GRANT_GROUP──group-name─────┤                                        ├─RS─┤
  └─GRANT_USER──user-name───────┘                                        └─UR─┘

►─┬────────────────────────┬─┬─────────────────────────────┬─┬──────────────────────────────┬──►
  └─MESSAGES──message-file─┘ └─OWNER──authorization-id─────┘ └─QUALIFIER──qualifier-name─────┘

                                       ┌─REOPT NONE───┐
►─┬──────────────────────────────────┬─┼──────────────┼─┬──────────────────────────┬──►
  └─QUERYOPT──optimization-level─────┘ ├─REOPT ONCE───┤ └─SQLERROR──┬─CHECK─────┬──┘
                                       └─REOPT ALWAYS─┘             ├─CONTINUE──┤
                                                                    └─NOPACKAGE─┘

►─┬──────────────────────┬──────────────────────────────────────────────────────►
  └─SQLWARN──┬─NO──┬─────┘
             └─YES─┘

►─┬──────────────────────────────────────────────────────────────────────────────────┬──►◄
  │                   ┌─NO──┐                                                        │
  └─STATICREADONLY────┴─YES─┴─┬─────────────────────────┬─┬───────────────────────────┬┘
                              └─VALIDATE──┬─BIND─┬──────┘ └─TRANSFORM GROUP──groupname─┘
                                          └─RUN──┘
```

### For DB2 on servers other than Windows and UNIX

```
►►──BIND──filename──────────────────────────────────────────────────────────►

►─┬───────────────────────────────────────────────────────────────────┬──►
  └─ACTION──┬─ADD─────────────────────────────────────────────────────┤
            │ ┌─REPLACE─┐                                             │
            └─┴─────────┴─┬────────────────────┬─┬─────────────────────┬┘
                          └─RETAIN──┬─NO──┬────┘ └─REPLVER──version-id─┘
                                    └─YES─┘
```

```
►──┬──────────────────────────┬──┬──────────────────────────┬──┬──────────────────────────┬──────►
   │              ┌─UNAMBIG─┐ │  └─CCSIDG──double-ccsid─┘  └─CCSIDM──mixed-ccsid─┘
   └─BLOCKING───┼─ALL─────┼─┘
                  └─NO──────┘

►──┬─────────────────────┬──┬────────────────────────────┬──┬──────────────────────────┬──────►
   └─CCSIDS──sbcs-ccsid─┘  └─CHARSUB────┬─DEFAULT─┬─┘  └─CLIPKG──cli-packages─┘
                                 ├─BIT─────┤
                                 ├─MIXED───┤
                                 └─SBCS────┘

►──┬───────────────────────┬──┬──────────────────────────┬─────────────────────────────────►
   └─CNULREQD─────┬─NO──┬─┘  └─COLLECTION──schema-name─┘
                    └─YES─┘

►──┬───────────────────────────────┬──┬──────────────────────────────┬──┬────────────────┬──►
   │  (1)                          │  └─DBPROTOCOL────┬─DRDA────┬─┘  └─DEC────┬─15─┬─┘
   └────────────DATETIME────┬─DEF─┬─┘                  └─PRIVATE─┘             └─31─┘
                              ├─EUR─┤
                              ├─ISO─┤
                              ├─JIS─┤
                              ├─LOC─┤
                              └─USA─┘

►──┬──────────────────────────┬──┬──────────────────────────────────────────────────┬────►
   └─DECDEL────┬─COMMA──┬─┘  │  (2)                                             │
                 └─PERIOD─┘    └────────────DEGREE────┬─1─────────────────────┬─┘
                                                        ├─degree-of-parallelism─┤
                                                        └─ANY───────────────────┘

►──┬─────────────────────────────────┬──┬──────────────────────────┬──────────────────────►
   │                ┌─RUN────────┐ │  └─ENCODING────┬─ASCII───┬─┘
   └─DYNAMICRULES──┼─BIND───────┼─┘                  ├─EBCDIC──┤
                    ├─INVOKERUN──┤                    ├─UNICODE─┤
                    ├─INVOKEBIND─┤                    └─CCSID───┘
                    ├─DEFINERUN──┤
                    └─DEFINEBIND─┘

►──┬──────────────────────────────┬──┬────────────────────┬──┬──────────────────────┬──────►
   │  (3)                         │  └─GENERIC──string─┘  └─GRANT────┬─authid─┬─┘
   └────────────EXPLAIN────┬─NO──┬─┘                                 └─PUBLIC─┘
                             └─YES─┘

►──┬──────────────────────────┬──┬────────────────────┬──┬─────────────────────────┬──────►
   └─IMMEDWRITE─────┬─NO──┬─┘  └─INSERT────┬─BUF─┬─┘  │              ┌─CS─┐ │
                      ├─YES─┤                └─DEF─┘      └─ISOLATION────┼─NC─┼─┘
                      └─PH1─┘                                            ├─RR─┤
                                                                          ├─RS─┤
                                                                          └─UR─┘

►──┬──────────────────────────┬──┬──────────────────────────┬──┬──────────────────────┬───►
   └─KEEPDYNAMIC─────┬─YES─┬─┘  └─MESSAGES──message-file─┘  └─OPTHINT─────hint-id───┘
                       └─NO──┘

►──┬──────────────────────────────┬──┬─────────────────────────────┬──┬─────────────────────┬──►
   └─OS400NAMING─────┬─SYSTEM─┬─┘  └─OWNER──authorization-id─┘  └─PATH──schema-name─┘
                       └─SQL────┘
```

```
                                                                  ┌─REOPT NONE───┐
►──┬──────────────────────────────┬──┬─────────────────────────┬──┼──────────────┼─────────►
   └─QUALIFIER──qualifier-name────┘  └─RELEASE──┬─COMMIT─────┬─┘  ├─REOPT ONCE───┤
                                                └─DEALLOCATE─┘    └─REOPT ALWAYS─┘

►──┬────────────────┬──┬──────────────────────┬──┬─────────────────────────────┬──────────►
   ├─REOPT VARS─────┤  └─SORTSEQ──┬─JOBRUN─┬──┘  └─SQLERROR──┬─CHECK─────┬────┘
   └─NOREOPT VARS───┘             └─HEX────┘                 ├─CONTINUE──┤
                                                             └─NOPACKAGE─┘

►──┬─────────────────────────┬──┬────────────────────────────┬──┬─────────────────┬───────►◄
   └─VALIDATE──┬─BIND─┬──────┘  └─STRDEL──┬─APOSTROPHE─┬────┘  └─TEXT──label─────┘
               └─RUN──┘                   └─QUOTE──────┘
```

**Notes:**

1 If the server does not support the DATETIME DEF option, it is mapped to DATETIME ISO.

2 The DEGREE option is only supported by DRDA Level 2 Application Servers.

3 DRDA defines the EXPLAIN option to have the value YES or NO. If the server does not support the EXPLAIN YES option, the value is mapped to EXPLAIN ALL.

**Command parameters:**

**filename**

Specifies the name of the bind file that was generated when the application program was precompiled, or a list file containing the names of several bind files. Bind files have the extension `.bnd`. The full path name can be specified.

If a list file is specified, the @ character must be the first character of the list file name. The list file can contain several lines of bind file names. Bind files listed on the same line must be separated by plus (+) characters, but a + cannot appear in front of the first file listed on each line, or after the last bind file listed. For example,

```
/u/smith/sqllib/bnd/@all.lst
```

is a list file that contains the following bind files:

```
mybind1.bnd+mybind.bnd2+mybind3.bnd+
mybind4.bnd+mybind5.bnd+
mybind6.bnd+
mybind7.bnd
```

**ACTION**

Indicates whether the package can be added or replaced.

**ADD** Indicates that the named package does not exist, and that a new package is to be created. If the package already exists, execution stops, and a diagnostic error message is returned.

**REPLACE**

Indicates that the existing package is to be replaced by a new one with the same package name and creator. This is the default value for the ACTION option.

**RETAIN**

| Indicates whether BIND and EXECUTE authorities are to
| be preserved when a package is replaced. If ownership of

the package changes, the new owner grants the BIND and EXECUTE authority to the previous package owner.

**NO** Does not preserve BIND and EXECUTE authorities when a package is replaced. This value is not supported by DB2.

**YES** Preserves BIND and EXECUTE authorities when a package is replaced. This is the default value.

**REPLVER version-id**
Replaces a specific version of a package. The version identifier specifies which version of the package is to be replaced. If the specified version does not exist, an error is returned. If the REPLVER option of REPLACE is not specified, and a package already exists that matches the package name, creator, and version of the package being bound, that package will be replaced; if not, a new package will be added.

**BLOCKING**
Specifies the type of row blocking for cursors.

**ALL** Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as read-only.

**NO** Specifies not to block any cursors. Ambiguous cursors are treated as updatable.

**UNAMBIG**
Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as updatable.

**CCSIDG double-ccsid**
An integer specifying the coded character set identifier (CCSID) to be used for double byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**CCSIDM mixed-ccsid**
An integer specifying the coded character set identifier (CCSID) to be used for mixed byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**CCSIDS sbcs-ccsid**
An integer specifying the coded character set identifier (CCSID) to be used for single byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements.

This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**CHARSUB**
Designates the default character sub-type that is to be used for column definitions in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2 for Windows and UNIX.

**BIT** Use the FOR BIT DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**DEFAULT**
Use the target system defined default in all new character columns for which an explicit sub-type is not specified.

**MIXED**
Use the FOR MIXED DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**SBCS** Use the FOR SBCS DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**CLIPKG cli-packages**
An integer between 3 and 30 specifying the number of CLI large packages to be created when binding CLI bind files against a database.

**CNULREQD**
This option is related to the LANGLEVEL precompile option, which is not supported by DRDA. It is valid only if the bind file is created from a C or a C++ application. This DRDA bind option is not supported by DB2 for Windows and UNIX.

**NO** The application was coded on the basis of the LANGLEVEL SAA1 precompile option with respect to the null terminator in C string host variables.

**YES** The application was coded on the basis of the LANGLEVEL MIA precompile option with respect to the null terminator in C string host variables.

**COLLECTION schema-name**
Specifies a 30-character collection identifier for the package. If not specified, the authorization identifier for the user processing the package is used.

**DATETIME**
Specifies the date and time format to be used.

**DEF** Use a date and time format associated with the territory code of the database.

**EUR** Use the IBM standard for Europe date and time format.

**ISO** Use the date and time format of the International Standards Organization.

**JIS** Use the date and time format of the Japanese Industrial Standard.

**LOC** Use the date and time format in local form associated with the territory code of the database.

**USA** Use the IBM standard for U.S. date and time format.

**DBPROTOCOL**
Specifies what protocol to use when connecting to a remote site that is identified by a three-part name statement. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**DEC** Specifies the maximum precision to be used in decimal arithmetic operations. This DRDA precompile/bind option is not supported by DB2 for Windows and UNIX. The DRDA server will use a system defined default value if this option is not specified.

**15** 15-digit precision is used in decimal arithmetic operations.

**31** 31-digit precision is used in decimal arithmetic operations.

**DECDEL**
Designates whether a period (.) or a comma (,) will be used as the decimal point indicator in decimal and floating point literals. This DRDA precompile/bind option is not supported by DB2 for Windows and UNIX. The DRDA server will use a system defined default value if this option is not specified.

**COMMA**
Use a comma (,) as the decimal point indicator.

**PERIOD**
Use a period (.) as the decimal point indicator.

**DEGREE**
Specifies the degree of parallelism for the execution of static SQL statements in an SMP system. This option does not affect CREATE INDEX parallelism.

**1** The execution of the statement will not use parallelism.

**degree-of-parallelism**
Specifies the degree of parallelism with which the statement can be executed, a value between 2 and 32 767 (inclusive).

**ANY** Specifies that the execution of the statement can involve parallelism using a degree determined by the database manager.

**DYNAMICRULES**
Defines which rules apply to dynamic SQL at run time for the initial setting of the values used for authorization ID and for the implicit qualification of unqualified object references.

**RUN** Specifies that the authorization ID of the user executing the package is to be used for authorization checking of dynamic SQL statements. The authorization ID will also be used as the default package qualifier for implicit qualification of unqualified object references within dynamic SQL statements. This is the default value.

**BIND** Specifies that all of the rules that apply to static SQL for authorization and qualification are to be used at run time. That is, the authorization ID of the package owner is to be used for authorization checking of dynamic SQL statements, and the default package qualifier is to be used for implicit qualification of unqualified object references within dynamic SQL statements.

**DEFINERUN**
If the package is used within a routine context, the authorization

ID of the routine definer is to be used for authorization checking and for implicit qualification of unqualified object references within dynamic SQL statements within the routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES RUN.

**DEFINEBIND**

If the package is used within a routine context, the authorization ID of the routine definer is to be used for authorization checking and for implicit qualification of unqualified object references within dynamic SQL statements within the routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES BIND.

**INVOKERUN**

If the package is used within a routine context, the current statement authorization ID in effect when the routine is invoked is to be used for authorization checking of dynamic SQL statements and for implicit qualification of unqualified object references within dynamic SQL statements within that routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES RUN.

**INVOKEBIND**

If the package is used within a routine context, the current statement authorization ID in effect when the routine is invoked is to be used for authorization checking of dynamic SQL statements and for implicit qualification of unqualified object references within dynamic SQL statements within that routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES BIND.

**Note:** Because dynamic SQL statements will be using the authorization ID of the package owner in a package exhibiting bind behavior, the binder of the package should not have any authorities granted to them that the user of the package should not receive. Similarly, when defining a routine that will exhibit define behavior, the definer of the routine should not have any authorities granted to them that the user of the package should not receive since a dynamic statement will be using the authorization ID of the routine's definer.

The following dynamically prepared SQL statements cannot be used within a package that was not bound with DYNAMICRULES RUN: GRANT, REVOKE, ALTER, CREATE, DROP, COMMENT ON, RENAME, SET INTEGRITY, and SET EVENT MONITOR STATE.

**ENCODING**

Specifies the encoding for all host variables in static statements in the plan or package. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**EXPLAIN**

Stores information in the Explain tables about the access plans chosen for each SQL statement in the package. DRDA does not support the ALL value for this option.

**NO** Explain information will not be captured.

**YES** Explain tables will be populated with information about the chosen access plan at prep/bind time for static statements and at run time for incremental bind statements.

If the package is to be used for a routine and the package contains incremental bind statements, then the routine must be defined as MODIFIES SQL DATA. If this is not done, incremental bind statements in the package will cause a run time error (SQLSTATE 42985).

**REOPT**

Explain information for each reoptimizable incremental bind SQL statement is placed in the explain tables at run time. In addition, explain information is gathered for reoptimizable dynamic SQL statements at run time, even if the CURRENT EXPLAIN MODE register is set to NO.

If the package is to be used for a routine, the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**ALL** Explain information for each eligible static SQL statement will be placed in the Explain tables at prep/bind time. Explain information for each eligible incremental bind SQL statement will be placed in the Explain tables at run time. In addition, Explain information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN MODE register is set to `NO`.

If the package is to be used for a routine, the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**Note:** This value for EXPLAIN is not supported by DRDA.

**EXPLSNAP**

Stores Explain Snapshot information in the Explain tables. This DB2 precompile/bind option is not supported by DRDA.

**NO** An Explain Snapshot will not be captured.

**YES** An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables at prep/bind time for static statements and at run time for incremental bind statements.

If the package is to be used for a routine and the package contains incremental bind statements, then the routine must be defined as MODIFIES SQL DATA or incremental bind statements in the package will cause a run time error (SQLSTATE 42985).

**REOPT**

Explain snapshot information for each reoptimizable incremental bind SQL statement is placed in the explain tables at run time. In addition, explain snapshot information is gathered for reoptimizable dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO.

If the package is to be used for a routine, the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**ALL** An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables at prep/bind time. Explain snapshot information for each eligible incremental bind SQL statement will be placed in the Explain tables at run time. In addition, explain snapshot information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT register is set to NO.

If the package is to be used for a routine, then the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**Note:** This value is not supported by DRDA

**FEDERATED**

Specifies whether a static SQL statement in a package references a nickname or a federated view. If this option is not specified and a static SQL statement in the package references a nickname or a federated view, a warning is returned and the package is created. This option is not supported for DRDA.

**NO** A nickname or federated view is not referenced in the static SQL statements of the package. If a nickname or federated view is encountered in a static SQL statement during the prepare or bind phase of this package, an error is returned and the package is *not* created.

**YES** A nickname or federated view can be referenced in the static SQL statements of the package. If no nicknames or federated views are encountered in static SQL statements during the prepare or bind of the package, no errors or warnings are returned and the package is created.

**FUNCPATH**

Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is ″SYSIBM″,″SYSFUN″,USER where USER is the value of the USER special register.

**schema-name**

An SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time. The same schema cannot appear more than once in the function path. The number of schemas that can be specified is limited by the length of the resulting function path, which cannot exceed 254 bytes. The schema SYSIBM does not need to be explicitly specified; it is implicitly assumed to be the first schema if it is not included in the function path.

**GENERIC string**
Supports the binding of new options that are defined in the target database, but are not supported by DRDA. Do not use this option to pass bind options that *are* defined in BIND or PRECOMPILE. This option can substantially improve dynamic SQL performance. The syntax is as follows:

```
generic "option1 value1 option2 value2 ..."
```

Each option and value must be separated by one or more blank spaces. For example, if the target DRDA database is DB2 Universal Database, Version 8, one could use:

```
generic "explsnap all queryopt 3 federated yes"
```

to bind each of the EXPLSNAP, QUERYOPT, and FEDERATED options.

The maximum length of the string is 1023 bytes.

**GRANT**

**authid** Grants EXECUTE and BIND privileges to a specified user name or group ID.

**PUBLIC**
Grants EXECUTE and BIND privileges to PUBLIC.

**GRANT_GROUP group-name**
Grants EXECUTE and BIND privileges to a specified group ID.

**GRANT_USER user-name**
Grants EXECUTE and BIND privileges to a specified user name.

**Note:** If more than one of the GRANT, GRANT_GROUP, and GRANT_USER options are specified, only the last option specified is executed.

**INSERT**
Allows a program being precompiled or bound against a DB2 Enterprise Server Edition server to request that data inserts be buffered to increase performance.

**BUF** Specifies that inserts from an application should be buffered.

**DEF** Specifies that inserts from an application should not be buffered.

**ISOLATION**
Determines how far a program bound to this package can be isolated from the effect of other executing programs.

**CS** Specifies Cursor Stability as the isolation level.

**NC** No Commit. Specifies that commitment control is not to be used. This isolation level is not supported by DB2 for Windows and UNIX.

**RR** Specifies Repeatable Read as the isolation level.

**RS** Specifies Read Stability as the isolation level. Read Stability ensures that the execution of SQL statements in the package is isolated from other application processes for rows read and changed by the application.

**UR** Specifies Uncommitted Read as the isolation level.

**IMMEDWRITE**
Indicates whether immediate writes will be done for updates made to group buffer pool dependent pagesets or partitions. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**KEEPDYNAMIC**
Specifies whether dynamic SQL statements are to be kept after commit points. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**MESSAGES message-file**
Specifies the destination for warning, error, and completion status messages. A message file is created whether the bind is successful or not. If a message file name is not specified, the messages are written to standard output. If the complete path to the file is not specified, the current directory is used. If the name of an existing file is specified, the contents of the file are overwritten.

**OPTHINT**
Controls whether query optimization hints are used for static SQL. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**OS400NAMING**
Specifies which naming option is to be used when accessing DB2 UDB for iSeries data. Supported by DB2 UDB for iSeries only. For a list of supported option values, refer to the documentation for DB2 for iSeries.

Please note that because of the slashes used as separators, a DB2 utility can still report a syntax error at execution time on certain SQL statements which use the iSeries system naming convention, even though the utility might have been precompiled or bound with the OS400NAMING SYSTEM option. For example, the Command Line Processor will report a syntax error on an SQL CALL statement if the iSeries system naming convention is used, whether or not it has been precompiled or bound using the OS400NAMING SYSTEM option.

**OWNER authorization-id**
Designates a 30-character authorization identifier for the package owner. The owner must have the privileges required to execute the SQL statements contained in the package. Only a user with SYSADM or DBADM authority can specify an authorization identifier other than the user ID. The default value is the primary authorization ID of the precompile/bind process. SYSIBM, SYSCAT, and SYSSTAT are not valid values for this option.

**PATH** Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is ″SYSIBM″,″SYSFUN″,USER where USER is the value of the USER special register.

**schema-name**
An SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time.

**QUALIFIER qualifier-name**

Provides a 30-character implicit qualifier for unqualified objects contained in the package. The default is the owner's authorization ID, whether or not **owner** is explicitly specified.

**QUERYOPT optimization-level**

Indicates the desired level of optimization for all static SQL statements contained in the package. The default value is 5. The SET CURRENT QUERY OPTIMIZATION statement describes the complete range of optimization levels available. This DB2 precompile/bind option is not supported by DRDA.

**RELEASE**

Indicates whether resources are released at each COMMIT point, or when the application terminates. This DRDA precompile/bind option is not supported by DB2 for Windows and UNIX.

**COMMIT**

Release resources at each COMMIT point. Used for dynamic SQL statements.

**DEALLOCATE**

Release resources only when the application terminates.

**SORTSEQ**

Specifies which sort sequence table to use on the iSeries system. Supported by DB2 UDB for iSeries only. For a list of supported option values, refer to the documentation for DB2 for iSeries.

**SQLERROR**

Indicates whether to create a package or a bind file if an error is encountered.

**CHECK**

Specifies that the target system performs all syntax and semantic checks on the SQL statements being bound. A package will not be created as part of this process. If, while binding, an existing package with the same name and version is encountered, the existing package is neither dropped nor replaced even if **action replace** was specified.

**CONTINUE**

Creates a package, even if errors occur when binding SQL statements. Those statements that failed to bind for authorization or existence reasons can be incrementally bound at execution time if VALIDATE RUN is also specified. Any attempt to execute them at run time generates an error (SQLCODE -525, SQLSTATE 51015).

**NOPACKAGE**

A package or a bind file is not created if an error is encountered.

**REOPT**

Specifies whether to have DB2 determine an access path at run time using values for host variables, parameter markers, and special registers. Valid values are:

**NONE**

The access path for a given SQL statement containing host variables, parameter markers, or special registers will not be optimized using real values. The default estimates for the these variables is used, and the plan is cached and will be used subsequently. This is the default value.

**ONCE** The access path for a given SQL statement will be optimized using the real values of the host variables, parameter markers, or special registers when the query is first executed. This plan is cached and used subsequently.

**ALWAYS**
The access path for a given SQL statement will always be compiled and reoptimized using the values of the host variables, parameter markers, or special registers that are known each time the query is executed.

**REOPT / NOREOPT VARS**
These options have been replaced by REOPT ALWAYS and REOPT NONE; however, they are still supported for back-level compatibility. Specifies whether to have DB2 determine an access path at run time using values for host variables, parameter markers, and special registers. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**SQLWARN**
Indicates whether warnings will be returned from the compilation of dynamic SQL statements (via PREPARE or EXECUTE IMMEDIATE), or from describe processing (via PREPARE...INTO or DESCRIBE). This DB2 precompile/bind option is not supported by DRDA.

**NO** Warnings will not be returned from the SQL compiler.

**YES** Warnings will be returned from the SQL compiler.

**Note:** SQLCODE +238 is an exception. It is returned regardless of the **sqlwarn** option value.

**STATICREADONLY**
Determines whether static cursors will be treated as being READ ONLY. This DB2 precompile/bind option is not supported by DRDA.

**NO** All static cursors will take on the attributes as would normally be generated given the statement text and the setting of the LANGLEVEL precompile option.

**YES** Any static cursor that does not contain the FOR UPDATE or FOR READ ONLY clause will be considered READ ONLY. This is the default value.

**STRDEL**
Designates whether an apostrophe (′) or double quotation marks (") will be used as the string delimiter within SQL statements. This DRDA precompile/bind option is not supported by DB2 for Windows and UNIX. The DRDA server will use a system defined default value if this option is not specified.

**APOSTROPHE**
Use an apostrophe (′) as the string delimiter.

**QUOTE**
Use double quotation marks (") as the string delimiter.

**TEXT label**

The description of a package. Maximum length is 255 characters. The default value is blanks. This DRDA precompile/bind option is not supported by DB2 for Windows and UNIX.

**TRANSFORM GROUP**

Specifies the transform group name to be used by static SQL statements for exchanging user-defined structured type values with host programs. This transform group is not used for dynamic SQL statements or for the exchange of parameters and results with external functions or methods. This option is not supported by DRDA.

**groupname**

An SQL identifier of up to 18 characters in length. A group name cannot include a qualifier prefix and cannot begin with the prefix SYS since this is reserved for database use. In a static SQL statement that interacts with host variables, the name of the transform group to be used for exchanging values of a structured type is as follows:

- The group name in the TRANSFORM GROUP bind option, if any
- The group name in the TRANSFORM GROUP prep option as specified at the original precompilation time, if any
- The DB2_PROGRAM group, if a transform exists for the given type whose group name is DB2_PROGRAM
- No transform group is used if none of the above conditions exist.

The following errors are possible during the bind of a static SQL statement:

- SQLCODE yyyyy, SQLSTATE xxxxx: A transform is needed, but no static transform group has been selected.
- SQLCODE yyyyy, SQLSTATE xxxxx: The selected transform group does not include a necessary transform (TO SQL for input variables, FROM SQL for output variables) for the data type that needs to be exchanged.
- SQLCODE yyyyy, SQLSTATE xxxxx: The result type of the FROM SQL transform is not compatible with the type of the output variable, or the parameter type of the TO SQL transform is not compatible with the type of the input variable.

In these error messages, *yyyyy* is replaced by the SQL error code, and *xxxxx* by the SQL state code.

**VALIDATE**

Determines when the database manager checks for authorization errors and object not found errors. The package owner authorization ID is used for validity checking.

**BIND** Validation is performed at precompile/bind time. If all objects do not exist, or all authority is not held, error messages are produced. If **sqlerror continue** is specified, a package/bind file is produced despite the error message, but the statements in error are not executable.

**RUN** Validation is attempted at bind time. If all objects exist, and all authority is held, no further checking is performed at execution time.

If all objects do not exist, or all authority is not held at precompile/bind time, warning messages are produced, and the package is successfully bound, regardless of the **sqlerror continue** option setting. However, authority checking and existence checking for SQL statements that failed these checks during the precompile/bind process can be redone at execution time.

**Examples:**

The following example binds `myapp.bnd` (the bind file generated when the `myapp.sqc` program was precompiled) to the database to which a connection has been established:

```
db2 bind myapp.bnd
```

Any messages resulting from the bind process are sent to standard output.

**Usage notes:**

Binding a package using the REOPT option with the ONCE or ALWAYS value specified might change the static and dynamic statement compilation and performance.

Binding can be done as part of the precompile process for an application program source file, or as a separate step at a later time. Use BIND when binding is performed as a separate process.

The name used to create the package is stored in the bind file, and is based on the source file name from which it was generated (existing paths or extensions are discarded). For example, a precompiled source file called `myapp.sql` generates a default bind file called `myapp.bnd` and a default package name of `MYAPP`. However, the bind file name and the package name can be overridden at precompile time by using the **bindfile** and the **package** options.

Binding a package with a schema name that does not already exist results in the implicit creation of that schema. The schema owner is SYSIBM. The CREATEIN privilege on the schema is granted to PUBLIC.

BIND executes under the transaction that was started. After performing the bind, BIND issues a COMMIT or a ROLLBACK to terminate the current transaction and start another one.

Binding stops if a fatal error or more than 100 errors occur. If a fatal error occurs, the utility stops binding, attempts to close all files, and discards the package.

When a package exhibits bind behavior, the following will be true:

1. The implicit or explicit value of the BIND option OWNER will be used for authorization checking of dynamic SQL statements.
2. The implicit or explicit value of the BIND option QUALIFIER will be used as the implicit qualifier for qualification of unqualified objects within dynamic SQL statements.

3. The value of the special register CURRENT SCHEMA has no effect on qualification.

In the event that multiple packages are referenced during a single connection, all dynamic SQL statements prepared by those packages will exhibit the behavior as specified by the DYNAMICRULES option for that specific package and the environment they are used in.

Parameters displayed in the SQL0020W message are correctly noted as errors, and will be ignored as indicated by the message.

If an SQL statement is found to be in error and the BIND option SQLERROR CONTINUE was specified, the statement will be marked as invalid. In order to change the state of the SQL statement, another BIND must be issued . Implicit and explicit rebind will not change the state of an invalid statement. In a package bound with VALIDATE RUN, a statement can change from static to incremental bind or incremental bind to static across implicit and explicit rebinds depending on whether or not object existence or authority problems exist during the rebind.

**Related concepts:**

- "Isolation levels" in the *SQL Reference, Volume 1*
- "Authorization Considerations for Dynamic SQL" in the *Application Development Guide: Programming Client Applications*
- "Effect of DYNAMICRULES bind option on dynamic SQL" in the *Application Development Guide: Programming Client Applications*
- "Effects of REOPT on static SQL" in the *Application Development Guide: Programming Client Applications*
- "Effects of REOPT on dynamic SQL" in the *Application Development Guide: Programming Client Applications*

**Related reference:**

- "SET CURRENT QUERY OPTIMIZATION statement" in the *SQL Reference, Volume 2*
- "PRECOMPILE" on page 560
- "DB2 CLI bind files and package names" in the *CLI Guide and Reference, Volume 1*
- "Special registers" in the *SQL Reference, Volume 1*
- "Datetime values" in the *SQL Reference, Volume 1*

# CATALOG APPC NODE

Adds an APPC node entry to the node directory. The Advanced Program-to-Program Communications protocol is used to access the remote node.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None

**Command syntax:**

```
►►──CATALOG──┬───────┬──APPC NODE──nodename──REMOTE──symbolic-destination-name────────►
             └─ADMIN─┘

►──┬─────────────────────────────┬──┬──────────────────────────────────┬──────────────►
   │           ┌─PROGRAM─┐       │  └─REMOTE_INSTANCE──instance-name──┘
   └─SECURITY──┼─NONE────┼───────┘
               └─SAME────┘

►──┬─────────────────────┬──┬──────────────────────────────────┬──────────────────────►
   └─SYSTEM──system-name─┘  └─OSTYPE──operating-system-type────┘

►──┬──────────────────────────┬───────────────────────────────────────────────────────►◄
   └─WITH──"comment-string"───┘
```

**Command parameters:**

**ADMIN**
: Specifies administration server nodes.

**NODE nodename**
: A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions.

**REMOTE symbolic-destination-name**
: Specifies the symbolic destination name of the remote partner node. The name corresponds to an entry in the CPI Communications side information table that contains the necessary information for the client to set up an APPC connection to the server (partner LU name, mode name, partner TP name). Maximum length is 8 characters.

**SECURITY**
: Specifies the extent to which security information is to be used in the allocation request sent to the partner LU. Valid values are:

  **PROGRAM**
  : Specifies that both a user name and a password are to be included in the allocation request sent to the partner LU. This is the default.

  **NONE**
  : Specifies that no security information is to be included in the allocation request sent to the partner LU.

**SAME** Specifies that a user name is to be included in the allocation request sent to the partner LU, together with an indicator that the user name has been ″already verified″. The partner must be configured to accept ″already verified″ security.

**Note:** For connections using APPC to a DB2 for Windows NT Version 7.1 (or later) server, a user ID longer than 8 bytes is only supported when SECURITY is specified as NONE.

**REMOTE_INSTANCE instance-name**
Specifies the real name of the instance to which an attachment is being made on the remote server machine.

**SYSTEM system-name**
Specifies a name that is used to identify the server machine.

**OSTYPE operating-system-type**
Specifies the operating system type of the server machine. Valid values are: AIX, WIN, HPUX, SUN, OS400, OS390, VM, VSE, SNI, SCO, and LINUX.

**WITH "comment-string"**
Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**Examples:**

```
db2 catalog appc node db2appc1 remote db2inst1 security program
   with "A remote APPC node"
```

**Usage notes:**

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a Windows client, it stores and maintains the node directory in the instance subdirectory where the client is installed. On an AIX client, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using the LIST NODE DIRECTORY command.

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "LIST NODE DIRECTORY" on page 504
- "TERMINATE" on page 705

# CATALOG APPN NODE

Writes information to the node directory about a remote workstation that uses APPN as its communication protocol. DB2 uses this information to establish the connection between an application and a remote database cataloged on this node.

This command is available on Windows, AIX, and Solaris only.

**Authorization:**

*sysadm*

**Required connection:**

None

**Command syntax:**

```
►►──CATALOG APPN NODE──nodename──┬──────────────────┬──REMOTE──partner-lu──────────►
                                 └─NETWORK──netid─┘

►─┬────────────────────┬──┬─────────────────────┬──┬──────────────┬─────────────────►
  └─LOCAL──local-lu─┘  └─TPNAME──tp-name─┘  └─MODE──mode─┘

►─┬─────────────────────────────┬──┬─────────────────────────────────────┬──────────►
  │            ┌─PROGRAM─┐      │  └─LANADDRESS──lan-adapter-address─┘
  └─SECURITY───┼─NONE────┼──────┘
               └─SAME────┘

►─┬────────────────────────────────┬──┬──────────────────────────┬─────────────────►◄
  └─CHGPWDLU──change_password_lu─┘  └─WITH──"comment-string"─┘
```

**Command parameters:**

**NODE nodename**
: Specifies the name of the remote workstation to catalog. This is the same name that is entered for the node name parameter when a database residing on that workstation is cataloged (using the CATALOG DATABASE command). The name must conform to DB2 naming conventions.

**NETWORK netid**
: Specifies the ID of the SNA network where the remote LU resides. This network ID is a string of one to eight characters that follows naming conventions for SNA.

**REMOTE partner-lu**
: Specifies the SNA partner logical unit used for the connection. Enter the LU name of the remote node. The name must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration). The name must follow SNA naming conventions.

**LOCAL local-lu**
: Specifies the alias of the SNA local logical unit used for the connection. It must be a string containing 1 to 8 non-blank characters. The alias must be entered exactly as it appears (using mixed case characters) in the corresponding SNA definition (from the Communication Manager configuration).

**TPNAME tp-name**
Specifies the APPC transaction program name of the database server. The default is DB2DRDA.

**MODE mode**
Specifies the SNA transmission mode used for the connection. The name must conform to SNA naming conventions.

If a value is not entered, DB2 stores a character string of eight blanks as the mode type.

**SECURITY**
Specifies the extent to which security information is to be used in the allocation request sent to the partner LU. Valid values are:

**PROGRAM**
Specifies that both a user name and a password are to be included in the allocation request sent to the partner LU. This is the default.

**NONE**
Specifies that no security information is to be included in the allocation request sent to the partner LU.

**SAME** Specifies that a user name is to be included in the allocation request sent to the partner LU, together with an indicator that the user name has been ″already verified″. The partner must be configured to accept ″already verified″ security.

**Note:** For connections using APPN to a DB2 for Windows NT Version 7.1 (or later) server, a user ID longer than 8 bytes is only supported when SECURITY is specified as NONE.

**LANADDRESS lan-adapter-address**
LAN Adapter Address of the DB2 server.

**CHGPWDLU change_password_lu**
Specifies the name of the partner LU that is to be used when changing the password for a host database server.

**WITH "comment-string"**
Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**Examples:**

The following example catalogs an APPN node:

```
db2 catalog appn node remnode remote rlu with "Catalog APPN NODE"
```

**Related reference:**

- "CATALOG DATABASE" on page 307
- "ATTACH" on page 275

# CATALOG DATABASE

Stores database location information in the system database directory. The database can be located either on the local workstation or on a remote node.

**Scope:**

In a partitioned database environment, when cataloging a local database into the system database directory, this command must be issued from a database partition on the server where the database resides.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──CATALOG──┬─DATABASE─┬──database-name──┬──────────┬──┬─────────────────────┬──────►
             └─DB───────┘                  └─AS──alias─┘  ├─ON──┬─path──┬───────┤
                                                          │     └─drive─┘       │
                                                          └─AT NODE──nodename───┘

►──┬───────────────────────────────────────────────────────────────┬──────────────►
   └─AUTHENTICATION──┬─SERVER─────────────────────────────────────┬┘
                     ├─CLIENT─────────────────────────────────────┤
                     ├─SERVER_ENCRYPT─────────────────────────────┤
                     ├─KERBEROS TARGET PRINCIPAL──principalname───┤
                     ├─DATA_ENCRYPT───────────────────────────────┤
                     ├─DATA_ENCRYPT_CMP───────────────────────────┤
                     └─GSSPLUGIN──────────────────────────────────┘

►──┬──────────────────────────┬───────────────────────────────────────────────────►◄
   └─WITH──"comment-string"───┘
```

**Command parameters:**

**DATABASE database-name**
: Specifies the name of the database to catalog.

**AS alias**
: Specifies an alias as an alternate name for the database being cataloged. If an alias is not specified, the database manager uses *database-name* as the alias.

**ON path/drive**
: On UNIX based systems, specifies the path on which the database being cataloged resides. On Windows operating systems, specifies the letter of the drive on which the database being cataloged resides.

**AT NODE nodename**
: Specifies the name of the node where the database being cataloged resides. This name should match the name of an entry in the node directory. If the

node name specified does not exist in the node directory, a warning is returned, but the database is cataloged in the system database directory. The node name should be cataloged in the node directory if a connection to the cataloged database is desired.

**AUTHENTICATION**

The authentication value is stored for remote databases (it appears in the output from the LIST DATABASE DIRECTORY command) but it is not stored for local databases.

Specifying an authentication type can result in a performance benefit.

**SERVER**
Specifies that authentication takes place on the node containing the target database.

**CLIENT**
Specifies that authentication takes place on the node where the application is invoked.

**SERVER_ENCRYPT**
Specifies that authentication takes place on the node containing the target database, and that passwords are encrypted at the source. Passwords are decrypted at the target, as specified by the authentication type cataloged at the source.

**KERBEROS**
Specifies that authentication takes place using Kerberos Security Mechanism. When authentication is Kerberos, and an APPC connection is used for access, only SECURITY=NONE is supported.

**TARGET PRINCIPAL principalname**
Fully qualified Kerberos principal name for the target server; that is, the fully qualified Kerberos principal of the DB2 instance owner in the form of `name/instance@REALM`. For Windows 2000, Windows XP, and Windows Server 2003, this is the logon account of the DB2 server service in the form of *`userid@DOMAIN`*, *`userid@xxx.xxx.xxx.`*`com` or *`domain\userid`*.

**DATA_ENCRYPT**
Specifies that authentication takes place on the node containing the target database, and that connections must use data encryption.

**DATA_ENCRYPT_CMP**
Specifies that authentication takes place on the node containing the target database, and that connections must use data encryption. This option is provided for compatibility with downlevel products that do not support data encryption, in which case they will be allowed to connect with SERVER_ENCRYPT and not encrypt user data. Any product that does support data encryption will be forced to use it.

**GSSPLUGIN**
Specifies that authentication takes place using an external GSS API-based plug-in security mechanism. When

authentication is GSSPLUGIN, and an APPC connection is used for access, only SECURITY=NONE is supported.

**WITH "comment-string"**
Describes the database or the database entry in the system database directory. The maximum length of a comment string is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

**Examples:**

```
db2 catalog database sample on /databases/sample
   with "Sample Database"
```

**Usage notes:**

Use CATALOG DATABASE to catalog databases located on local or remote nodes, recatalog databases that were uncataloged previously, or maintain multiple aliases for one database (regardless of database location).

DB2 automatically catalogs databases when they are created. It catalogs an entry for the database in the local database directory and another entry in the system database directory. If the database is created from a remote client (or a client which is executing from a different instance on the same machine), an entry is also made in the system database directory at the client instance.

If neither path nor node name is specified, the database is assumed to be local, and the location of the database is assumed to be that specified in the database manager configuration parameter *dftdbpath*.

Databases on the same node as the database manager instance are cataloged as *indirect* entries. Databases on other nodes are cataloged as *remote* entries.

CATALOG DATABASE automatically creates a system database directory if one does not exist. The system database directory is stored on the path that contains the database manager instance that is being used, and is maintained outside of the database.

List the contents of the system database directory using the LIST DATABASE DIRECTORY command. To list the contents of the local database directory use the LIST DATABASE DIRECTORY ON /PATH, where PATH is where the database was created.

**Note:** If directory caching is enabled, database and node directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395

- “LIST DATABASE DIRECTORY” on page 483
- “TERMINATE” on page 705
- “UNCATALOG DATABASE” on page 706

# CATALOG DCS DATABASE

Stores information about remote host or iSeries databases in the Database Connection Services (DCS) directory. These databases are accessed through an Application Requester (AR), such as DB2 Connect. Having a DCS directory entry with a database name matching a database name in the system database directory invokes the specified AR to forward SQL requests to the remote server where the database resides.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None

**Command syntax:**

```
►►──CATALOG DCS──┬─DATABASE─┬──database-name──┬──────────────────────────┬──────►
                 └─DB───────┘                 └─AS──target-database-name─┘

►──┬──────────────────┬──┬───────────────────────────┬──────────────────────────►
   └─AR──library-name─┘  └─PARMS──"parameter-string"─┘

►──┬───────────────────────┬───────────────────────────────────────────────────►◄
   └─WITH──"comment-string"─┘
```

**Command parameters:**

**DATABASE database-name**
: Specifies the alias of the target database to catalog. This name should match the name of an entry in the database directory that is associated with the remote node.

**AS target-database-name**
: Specifies the name of the target host or iSeries database to catalog.

**AR library-name**
: Specifies the name of the Application Requester library that is loaded and used to access a remote database listed in the DCS directory.

**Note:** If using the DB2 Connect AR, do not specify a library name. The default value will cause DB2 Connect to be invoked.

If not using DB2 Connect, specify the library name of the AR, and place that library on the same path as the database manager libraries. On Windows operating systems, the path is `drive:\sqllib\bin`. On UNIX based systems, the path is `$HOME/sqllib/lib` of the instance owner.

**PARMS "parameter-string"**
: Specifies a parameter string that is to be passed to the AR when it is invoked. The parameter string must be enclosed by double quotation marks.

**WITH ″comment-string″**

Describes the DCS directory entry. Any comment that helps to describe the database cataloged in this directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

**Examples:**

The following example catalogs information about the DB1 database, which is a DB2 for z/OS database, into the DCS directory:

```
db2 catalog dcs database db1 as dsn_db_1
   with "DB2/z/OS location name DSN_DB_1"
```

**Usage notes:**

The DB2 Connect program provides connections to DRDA Application Servers such as:

- DB2 for OS/390 or z/OS databases on System/370 and System/390 architecture host computers.
- DB2 for VM and VSE databases on System/370 and System/390 architecture host computers.
- iSeries databases on Application System/400 (AS/400) and iSeries computers.

The database manager creates a Database Connection Services directory if one does not exist. This directory is stored on the path that contains the database manager instance that is being used. The DCS directory is maintained outside of the database.

The database must also be cataloged as a remote database in the system database directory.

List the contents of the DCS directory using the LIST DCS DIRECTORY command.

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "TERMINATE" on page 705
- "UNCATALOG DCS DATABASE" on page 708
- "LIST DCS DIRECTORY" on page 493

# CATALOG LDAP DATABASE

Used to register the database in Lightweight Directory Access Protocol (LDAP).

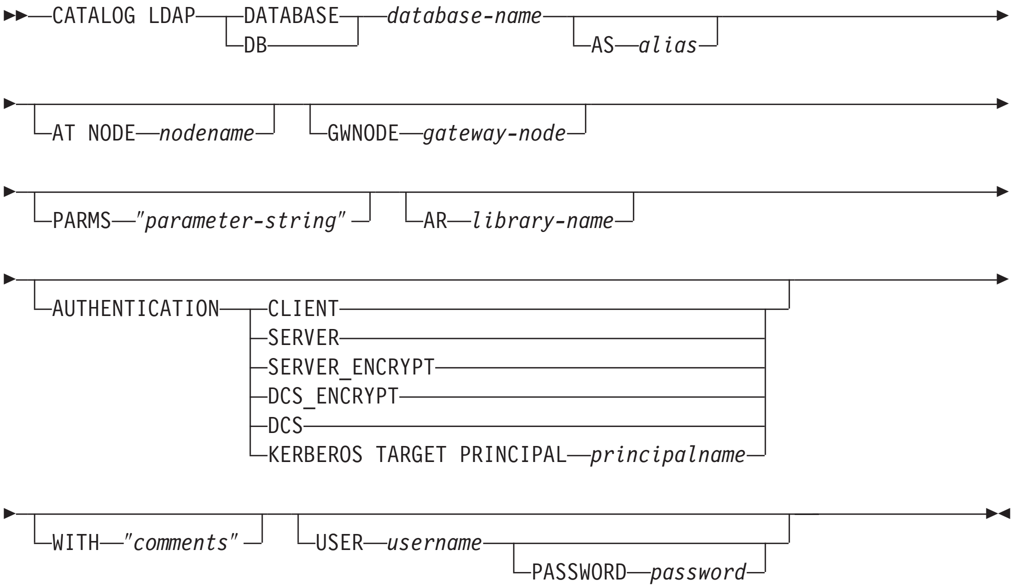
**Authorization:**

None

**Required connection:**

None

**Command syntax:**



```
>>--CATALOG LDAP--+-DATABASE-+--database-name--+-----------+------------------->
                  '-DB-------'                 '-AS--alias-'

>--+---------------------+--+-----------------------+------------------------->
   '-AT NODE--nodename---'  '-GWNODE--gateway-node--'

>--+-----------------------------+--+---------------------+-------------------->
   '-PARMS--"parameter-string"---'  '-AR--library-name----'

>--+------------------------------------------------------------------+------->
   '-AUTHENTICATION--+-CLIENT-------------------------------------+-'
                     +-SERVER-------------------------------------+
                     +-SERVER_ENCRYPT-----------------------------+
                     +-DCS_ENCRYPT--------------------------------+
                     +-DCS----------------------------------------+
                     '-KERBEROS TARGET PRINCIPAL--principalname---'

>--+---------------------+--+------------------------------------------+----><
   '-WITH--"comments"----'  '-USER--username--+---------------------+--'
                                              '-PASSWORD--password--'
```

**Command parameters:**

**DATABASE database-name**
: Specifies the name of the database to catalog.

**AS alias**
: Specifies an alias as an alternate name for the database being cataloged. If an alias is not specified, the database name is used as the alias.

**AT NODE nodename**
: Specifies the LDAP node name for the database server on which the database resides. This parameter must be specified when registering a database on a remote server.

**GWNODE gateway-node**
: Specifies the LDAP node name for the gateway server.

**PARMS ″parameter-string″**
: Specifies a parameter string that is passed to the Application Requester (AR) when accessing DCS databases.

**Note:** The change password *sym_dest_name* should not be specified in the parameter string. Use the keyword CHGPWDLU to specify the change password LU name when registering the DB2 server in LDAP.

**AR library-name**
Specifies the name of the Application Requester library that is loaded and used to access a remote database listed in the DCS directory.

**Note:** If using the DB2 Connect AR, do not specify a library name. The default value will cause DB2 Connect to be invoked.

If not using DB2 Connect, specify the library name of the AR, and place that library on the same path as the database manager libraries. On Windows operating systems, the path is `drive:\sqllib\dll`. On UNIX based systems, the path is `$HOME/sqllib/lib` of the instance owner.

**AUTHENTICATION**
Specifies the authentication level. Valid values are:

**CLIENT**
Specifies that authentication takes place on the node from which the application is invoked.

**SERVER**
Specifies that authentication takes place on the node containing the target database.

**SERVER_ENCRYPT**
Specifies that authentication takes place on the node containing the target database, and that passwords are encrypted at the source. Passwords are decrypted at the target, as specified by the authentication type cataloged at the source.

**DCS_ENCRYPT**
Specifies that authentication takes place on the node containing the target database, except when using DB2 Connect; in that case, authentication takes place at the DRDA application server (AS). Passwords are encrypted at the source, and decrypted at the target, as specified by the authentication type cataloged at the source.

**DCS** Specifies that authentication takes place on the node containing the target database, except when using DB2 Connect; in that case, authentication takes place at the DRDA application server (AS).

**KERBEROS**
Specifies that authentication takes place using Kerberos Security Mechanism. When authentication is Kerberos, and an APPC connection is used for access, only SECURITY=NONE is supported.

**TARGET PRINCIPAL principalname**
Fully qualified Kerberos principal name for the target server; that is, the logon account of the DB2 server service in the form of `*userid*@*xxx.xxx.xxx*.com` or `*domain*\*userid*`.

**Note:** This parameter is valid only on Windows 2000 clients.

**WITH "comments"**
Describes the DB2 server. Any comment that helps to describe the server registered in the network directory can be entered. Maximum length is 30

characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

**USER username**

Specifies the user's LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to create the object in the LDAP directory. If the user's LDAP DN is not specified, the credentials of the current logon user will be used.

**Note:** If the user's LDAP DN and password have been specified using **db2ldcfg**, the user name and password do not have to be specified here.

**PASSWORD password**

Account password.

**Note:** If the user's LDAP DN and password have been specified using **db2ldcfg**, the user name and password do not have to be specified here.

**Usage notes:**

If the node name is not specified, DB2 will use the first node in LDAP that represents the DB2 server on the current machine.

It might be necessary to manually register (catalog) the database in LDAP if:

- The database server does not support LDAP. The administrator must manually register each database in LDAP to allow clients that support LDAP to access the database without having to catalog the database locally on each client machine.
- The application wants to use a different name to connect to the database. In this case, the administrator can catalog the database using a different alias name.
- The database resides at the host or iSeries database server. In this case, the administrator can register the database in LDAP and specify the gateway node through the GWNODE parameter.
- During CREATE DATABASE IN LDAP the database name already exists in LDAP. The database is still created on the local machine (and can be accessed by local applications), but the existing entry in LDAP will not be modified to reflect the new database. In this case, the administrator can:
  - Remove the existing database entry in LDAP and manually register the new database in LDAP.
  - Register the new database in LDAP using a different alias name.

**Related reference:**

- "REGISTER" on page 613
- "UNCATALOG LDAP DATABASE" on page 709
- "CATALOG LDAP NODE" on page 316
- "UNCATALOG LDAP NODE" on page 710
- "db2ldcfg - Configure LDAP Environment" on page 120

# CATALOG LDAP NODE

Catalogs a new node entry in Lightweight Directory Access Protocol (LDAP).

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──CATALOG LDAP──NODE──nodename──AS──nodealias──────────────────────────────►

►──┬─────────────────────────────────────┬──────────────────────────────────►◄
   └─USER──username──┬─────────────────────┬┘
                     └─PASSWORD──password──┘
```

**Command parameters:**

**NODE nodename**
Specifies the LDAP node name of the DB2 server.

**AS nodealias**
Specifies a new alias name for the LDAP node entry.

**USER username**
Specifies the user's LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to create the object in the LDAP directory. If the user's LDAP DN is not specified, the credentials of the current logon user will be used.

**PASSWORD password**
Account password.

**Usage notes:**

The CATALOG LDAP NODE command is used to specify a different alias name for the node that represents the DB2 server.

**Related reference:**

- "CATALOG LDAP DATABASE" on page 313
- "UNCATALOG LDAP DATABASE" on page 709
- "UNCATALOG LDAP NODE" on page 710

# CATALOG LOCAL NODE

Creates a local alias for an instance that resides on the same machine. A local node should be cataloged when there is more than one instance on the same workstation to be accessed from the user's client. Interprocess Communications (IPC) is used to access the local node.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None

**Command syntax:**

```
►►──CATALOG──┬───────┬──LOCAL NODE──nodename──┬──────────────────────────┬──────────►
             └─ADMIN─┘                         └─INSTANCE──instancename─┘

►──┬──────────────────────┬──┬────────────────────────────────────┬──────────────────►
   └─SYSTEM──system-name─┘  └─OSTYPE──operating-system-type─┘

►──┬─────────────────────────┬────────────────────────────────────────────────────►◄
   └─WITH──"comment-string"─┘
```

**Command parameters:**

**ADMIN**
Specifies that a local administration server node is to be cataloged.

**NODE nodename**
A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions.

**INSTANCE instancename**
Name of the local instance to be accessed.

**SYSTEM system-name**
Specifies the DB2 system name that is used to identify the server machine.

**OSTYPE operating-system-type**
Specifies the operating system type of the server machine. Valid values are: AIX, WIN, HPUX, SUN, OS390, OS400, VM, VSE, SNI, SCO, LINUX and DYNIX.

**WITH "comment-string"**
Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**Examples:**

Workstation A has two server instances, `inst1` and `inst2`. To create databases at both instances from a single CLP session, issue the following sequence of commands (assume the **DB2INSTANCE** environment variable is set to `inst1`):

1. Create a local database at `inst1`:
   ```
   db2 create database mydb1
   ```
2. Catalog another server instance on this workstation:
   ```
   db2 catalog local node mynode2 instance inst2
   ```
3. Create a database at `mynode2`:
   ```
   db2 attach to mynode2
   db2 create database mydb2
   ```

**Usage notes:**

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use TERMINATE. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "TERMINATE" on page 705

# CATALOG NAMED PIPE NODE

Adds a named pipe node entry to the node directory. The named pipe is used to access the remote node.

This command is available on Windows only.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None

**Command syntax:**

```
►►──CATALOG──┬───────┬──NPIPE NODE──nodename──REMOTE──computername──────────────►
             └─ADMIN─┘

►─INSTANCE──instancename──┬──────────────────────┬───────────────────────────────►
                          └─SYSTEM──system-name──┘

►──┬────────────────────────────────┬──┬──────────────────────────┬──────────────►◄
   └─OSTYPE──operating-system-type──┘  └─WITH──"comment-string"───┘
```

**Command parameters:**

**ADMIN**
Specifies that an NPIPE administration server node is to be cataloged.

**NODE nodename**
A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions.

**REMOTE computername**
The computer name of the node on which the target database resides. Maximum length is 15 characters.

**INSTANCE instancename**
Name of the server instance on which the target database resides. Identical to the name of the remote named pipe, which is used to communicate with the remote node.

**SYSTEM system-name**
Specifies the DB2 system name that is used to identify the server machine.

**OSTYPE operating-system-type**
Specifies the operating system type of the server machine. Valid values are: AIX, WIN, HPUX, SUN, OS390, OS400, VM, VSE, SNI, SCO, and LINUX.

**WITH "comment-string"**
Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A

carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**Examples:**

```
db2 catalog npipe node db2np1 remote nphost instance db2inst1
   with "A remote named pipe node."
```

**Usage notes:**

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a Windows client, it stores and maintains the node directory in the instance subdirectory where the client is installed. On an AIX client, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using the LIST NODE DIRECTORY command.

**Note:** If directory caching is enabled (see the configuration parameter *dir_cache* in the GET DATABASE MANAGER CONFIGURATION command), database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "LIST NODE DIRECTORY" on page 504
- "TERMINATE" on page 705

# CATALOG NETBIOS NODE

Adds a NetBIOS node entry to the node directory. The NetBIOS communications protocol is used to access the remote node.

This command is available on Windows only.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──CATALOG──┬───────┬──NETBIOS NODE──nodename──REMOTE──server-nname──────────►
             └─ADMIN─┘

►─ADAPTER──adapter-number──┬─────────────────────────────────┬────────────────►
                           └─REMOTE_INSTANCE──instance-name──┘

►──┬─────────────────────────┬──┬──────────────────────────────────┬──────────►
   └─SYSTEM──system-name─────┘  └─OSTYPE──operating-system-type────┘

►──┬──────────────────────────┬───────────────────────────────────────────────►◄
   └─WITH──"comment-string"───┘
```

**Command parameters:**

**ADMIN**
Specifies the administration server database partitition.

**NODE nodename**
A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions.

**REMOTE server-nname**
The name of the remote workstation where the target database resides. This name must conform to the naming conventions of the database manager. This is the workstation name (*nname*) found in the database manager configuration file of the server workstation.

**ADAPTER adapter-number**
Specifies the local, logical, outgoing LAN adapter number. The default value is zero.

**REMOTE_INSTANCE instance-name**
Specifies the real name of the instance to which an attachment is being made on the remote server machine.

**SYSTEM system-name**
Specifies a name that is used to identify the server machine.

**OSTYPE operating-system-type**
Specifies the operating system type of the server machine. The only supported operating system that currently permits NetBios connections is Windows, so the only value currently permitted for OSTYPE is WIN.

**WITH "comment-string"**
Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**Examples:**

```
db2 catalog netbios node db2netb1 remote db2inst1 adapter 0
   with "A remote NetBIOS node"
```

**Usage notes:**

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a Windows client, it stores and maintains the node directory in the instance subdirectory where the client is installed. On an AIX client, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using the LIST NODE DIRECTORY command.

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "LIST NODE DIRECTORY" on page 504
- "TERMINATE" on page 705

# CATALOG ODBC DATA SOURCE

Catalogs a user or system ODBC data source.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database or file system. That name is used to access the database or file system through ODBC APIs. Either user or system data sources can be cataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows platforms only.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
                .-USER---.
>>--CATALOG--+--------+--ODBC DATA SOURCE--data-source-name-------------------><
                '-SYSTEM-'
```

**Command parameters:**

**USER** Catalog a user data source. This is the default if no keyword is specified.

**SYSTEM**
Catalog a system data source.

**ODBC DATA SOURCE data-source-name**
Specifies the name of the data source to be cataloged. Maximum length is 32 characters.

**Related reference:**

- "LIST ODBC DATA SOURCES" on page 507
- "UNCATALOG ODBC DATA SOURCE" on page 712

## CATALOG TCPIP NODE

Adds a Transmission Control Protocol/Internet Protocol (TCP/IP) node entry to the node directory. The TCP/IP communications protocol is used to access the remote node. The CATALOG TCPIP NODE command is run on a client.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──CATALOG──┬───────┬──TCPIP NODE──nodename──REMOTE──┬─hostname───┬──────────────►
             └─ADMIN─┘                                └─IP address─┘

   (1)
►───────SERVER──service-name──┬────────────────┬──────────────────────────────────►
                              └─SECURITY SOCKS─┘

►──┬─────────────────────────────┬──┬────────────────────┬────────────────────────►
   └─REMOTE_INSTANCE──instance-name─┘  └─SYSTEM──system-name─┘

►──┬─────────────────────────────┬──┬──────────────────────┬────────────────────►◄
   └─OSTYPE──operating-system-type─┘  └─WITH──"comment-string"─┘
```

**Notes:**

1 SERVER must not be specified for ADMIN nodes, but is mandatory for non-ADMIN nodes.

**Command parameters:**

**ADMIN>**
Specifies that a TCP/IP administration server node is to be cataloged.

**NODE nodename**
A local alias for the node to be cataloged. This is an arbitrary name on the user's workstation, used to identify the node. It should be a meaningful name to make it easier to remember. The name must conform to database manager naming conventions.

**REMOTE** *hostname/IP address*
The host name or the IP address of the node where the target database resides. The host name is the name of the node that is known to the TCP/IP network. The maximum length of the host name is 255 characters.

**SERVER** *service-name*
Specifies the service name or the port number of the server database manager instance. The maximum length is 14 characters. This parameter is case sensitive.

If a service name is specified, the *services* file on the client is used to map the service name to a port number. A service name is specified in the server's database manager configuration file, and the *services* file on the server is used to map this service name to a port number. The port number on the client and the server must match.

A port number, instead of a service name, can be specified in the database manager configuration file on the server, but this is not recommended. If a port number is specified, no service name needs to be specified in the local TCP/IP *services* file.

**Note:** This parameter must not be specified for ADMIN nodes. The value on ADMIN nodes is always 523.

**SECURITY SOCKS**
Specifies that the node will be SOCKS-enabled.

The following environment variables are mandatory and *must* be set to enable SOCKS:

**SOCKS_NS**
The Domain Name Server for resolving the host address of the SOCKS server. This should be an IP address.

**SOCKS_SERVER**
The fully qualified host name or the IP address of the SOCKS server. If the SOCKSified DB2 client is unable to resolve the fully qualified host name, it assumes that an IP address has been entered.

One of the following conditions should be true:

- The SOCKS server should be reachable via the domain name server
- It should be listed in the `hosts` file. The location of this file is described in the TCP/IP documentation.
- It should be in an IP address format.

If this command is issued after a **db2start**, it is necessary to issue a TERMINATE command to have the command take effect.

**REMOTE_INSTANCE** *instance-name*
Specifies the name of the server instance to which an attachment is being made.

**SYSTEM** *system-name*
Specifies the DB2 system name that is used to identify the server machine. The is the name of the physical machine, server system, or workstation.

**OSTYPE** *operating-system-type*
Specifies the operating system type of the server machine. Valid values are: AIX, WIN, HPUX, SUN, OS390, OS400, VM, VSE, SNI, SCO, and LINUX.

**WITH "comment-string"**
Describes the node entry in the node directory. Any comment that helps to describe the node can be entered. Maximum length is 30 characters. A

carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**Examples:**

```
db2 catalog tcpip node db2tcp1 remote tcphost server db2inst1
   with "A remote TCP/IP node"

db2 catalog tcpip node db2tcp2 remote 9.21.15.235 server db2inst2
   with "TCP/IP node using IP address"
```

**Usage notes:**

The database manager creates the node directory when the first node is cataloged (that is, when the first CATALOG...NODE command is issued). On a Windows client, it stores and maintains the node directory in the instance subdirectory where the client is installed. On an AIX client, it creates the node directory in the DB2 installation directory.

List the contents of the local node directory using the LIST NODE DIRECTORY command.

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database manager. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "LIST NODE DIRECTORY" on page 504
- "TERMINATE" on page 705

# CHANGE DATABASE COMMENT

Changes a database comment in the system database directory or the local database directory. New comment text can be substituted for text currently associated with a comment.

**Scope:**

This command only affects the database partition on which it is executed.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

None

**Command syntax:**

```
►►──CHANGE──┬─DATABASE─┬──database-alias──COMMENT──┬────────────────────┬──────────►
            └─DB───────┘                            └─ON──┬─path──┬──┘
                                                          └─drive─┘

►─WITH──"comment-string"──────────────────────────────────────────────────────────►◄
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database whose comment is to be changed. To change the comment in the system database directory, specify the alias for the database. To change the comment in the local database directory, specify the path where the database resides (with the *path* parameter), and enter the name (not the alias) of the database.

**ON path/drive**
On UNIX based systems, specifies the path on which the database resides, and changes the comment in the local database directory. If a path is not specified, the database comment for the entry in the system database directory is changed. On Windows operating systems, specifies the letter of the drive on which the database resides.

**WITH "comment-string"**
Describes the entry in the system database directory or the local database directory. Any comment that helps to describe the cataloged database can be entered. The maximum length of a comment string is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

**Examples:**

The following example changes the text in the system database directory comment for the SAMPLE database from "Test 2 - Holding" to "Test 2 - Add employee inf rows":

```
db2 change database sample comment
   with "Test 2 - Add employee inf rows"
```

**Usage notes:**

New comment text replaces existing text. To append information, enter the old comment text, followed by the new text.

Only the comment for an entry associated with the database alias is modified. Other entries with the same database name, but with different aliases, are not affected.

If the path is specified, the database alias must be cataloged in the local database directory. If the path is not specified, the database alias must be cataloged in the system database directory.

**Related reference:**

- "CREATE DATABASE" on page 331

# CHANGE ISOLATION LEVEL

Changes the way that DB2 isolates data from other processes while a database is being accessed.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
                                  ┌─CS─┐
►►──CHANGE──┬─SQLISL────┬──TO──┼─NC─┼──────────────────────────────────────►◄
            └─ISOLATION─┘      ├─RR─┤
                               ├─RS─┤
                               └─UR─┘
```

**Command parameters:**

**TO**

**CS** Specifies cursor stability as the isolation level.

**NC** Specifies no commit as the isolation level. Not supported by DB2.

**RR** Specifies repeatable read as the isolation level.

**RS** Specifies read stability as the isolation level.

**UR** Specifies uncommitted read as the isolation level.

**Usage notes:**

DB2 uses isolation levels to maintain data integrity in a database. The isolation level defines the degree to which an application process is isolated (shielded) from changes made by other concurrently executing application processes.

If a selected isolation level is not supported by a database, it is automatically escalated to a supported level at connect time.

Isolation level changes are not permitted while connected to a database with a type 1 connection. The back end process must be terminated before isolation level can be changed:

```
db2 terminate
db2 change isolation to ur
db2 connect to sample
```

Changes are permitted using a type 2 connection, but should be made with caution, because the changes will apply to every connection made from the same command line processor back-end process. The user assumes responsibility for remembering which isolation level applies to which connected database.

In the following example, a user is in DB2 interactive mode following creation of the SAMPLE database:

```
update command options using c off
catalog db sample as sample2

set client connect 2

connect to sample
connect to sample2

change isolation to cs
set connection sample
declare c1 cursor for select * from org
open c1
fetch c1 for 3 rows

change isolation to rr
fetch c1 for 2 rows
```

An SQL0514N error occurs because c1 is not in a prepared state for this isolation level.

```
change isolation to cs
set connection sample2
fetch c1 for 2 rows
```

An SQL0514N error occurs because c1 is not in a prepared state for this database.

```
declare c1 cursor for select division from org
```

A DB21029E error occurs because cursor c1 has already been declared and opened.

```
set connection sample
fetch c1 for 2 rows
```

This works because the original database (SAMPLE) was used with the original isolation level (CS).

**Related concepts:**

- "Isolation levels" in the *SQL Reference, Volume 1*

**Related reference:**

- "SET CLIENT" on page 678
- "QUERY CLIENT" on page 587

# CREATE DATABASE

Initializes a new database with an optional user-defined collating sequence, creates the three initial table spaces, creates the system tables, and allocates the recovery log. When you initialize a new database you can specify the AUTOCONFIGURE option to display and optionally apply the initial values for the buffer pool size, database and database manager parameters. The AUTOCONFIGURE option is not available in a partitioned database environment.

This command is not valid on a client.

**Scope:**

In a partitioned database environment, this command affects all database partitions that are listed in the `db2nodes.cfg` file.

The database partition from which this command is issued becomes the catalog database partition for the new database.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

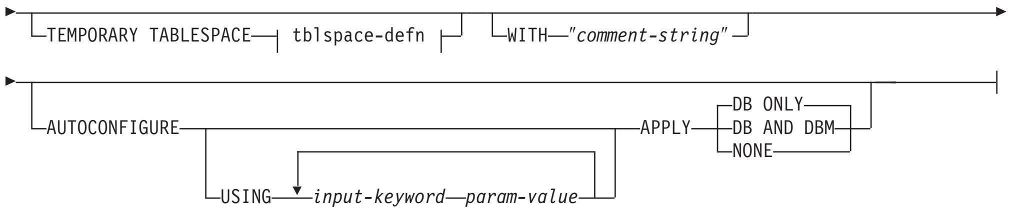
Instance. To create a database at another (remote) node, it is necessary to first attach to that node. A database connection is temporarily established by this command during processing.

**Command syntax:**

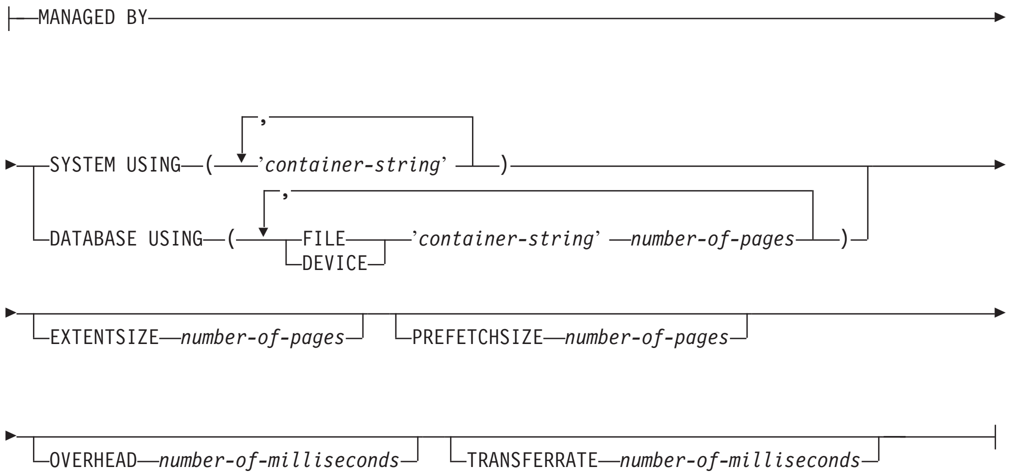
```
>>──CREATE──┬─DATABASE─┬──database-name──┬────────────────────────────┬──────────><
            └─DB───────┘                 ├─AT DBPARTITIONNUM──────────┤
                                         └─┤ Create Database options ├┘
```

## Create Database options:

```
├──┬──────────────┬──┬───────────────────────────┬──────────────────────────────────►
   └─ON──┬─path──┬┘  └─ALIAS──database-alias─────┘
         └─drive─┘

►──┬──────────────────────────────────────────────────┬──┬──────────────────────────────────────┬──►
   └─USING CODESET──codeset──TERRITORY──territory─────┘  └─COLLATE USING──┬─SYSTEM───────────┬─┘
                                                                          ├─COMPATIBILITY────┤
                                                                          ├─IDENTITY─────────┤
                                                                          ├─IDENTITY_16BIT───┤
                                                                          ├─UCA400_NO────────┤
                                                                          ├─UCA400_LTH───────┤
                                                                          └─NLSCHAR──────────┘

►──┬──────────────────┬──┬──────────────────────────────────┬──────────────────────►
   └─NUMSEGS──numsegs─┘  └─DFT_EXTENT_SZ──dft_extentsize────┘

►──┬──────────────────────────────────────────┬──┬───────────────────────────────────────┬──►
   └─CATALOG TABLESPACE──┤ tblspace-defn ├────┘  └─USER TABLESPACE──┤ tblspace-defn ├────┘
```



### tblspace-defn:



**Notes:**

1. The combination of the code set and territory values must be valid.
2. Not all collating sequences are valid with every code set and territory combination.
3. The table space definitions specified on CREATE DATABASE apply to all database partitions on which the database is being created. They cannot be specified separately for each database partition. If the table space definitions are to be created differently on particular database partitions, the CREATE TABLESPACE statement must be used.

   When defining containers for table spaces, $N can be used. $N will be replaced by the database partition number when the container is actually created. This is required if the user wants to specify containers in a multiple logical partition database.
4. In a partitioned database environment, use of the AUTOCONFIGURE option will result in failure of the CREATE DATABASE command. If you want to use the AUTOCONFIGURE option in a partitioned database environment, first create the database without specifying the AUTOCONFIGURE option, then run the AUTOCONFIGURE command on each partition.
5. The AUTOCONFIGURE option requires *sysadm* authority.

**Command parameters:**

**DATABASE database-name**

A name to be assigned to the new database. This must be a unique name that differentiates the database from any other database in either the local database directory or the system database directory. The name must conform to naming conventions for databases.

**AT DBPARTITIONNUM**

Specifies that the database is to be created only on the database partition that issues the command. You do not specify this option when you create a new database. You can use it to recreate a database partition that you dropped because it was damaged. After you use the CREATE DATABASE command with the AT DBPARITIONNUM option, the database at this partition is in the restore-pending state. You must immediately restore the database on this node. This parameter is not intended for general use. For example, it should be used with RESTORE DATABASE command if the database partition at a node was damaged and must be re-created. Improper use of this parameter can cause inconsistencies in the system, so it should only be used with caution.

**ON path/drive**

On UNIX based systems, specifies the path on which to create the database. If a path is not specified, the database is created on the default database path specified in the database manager configuration file (*dftdbpath* parameter). Maximum length is 205 characters. On the Windows operating system, specifies the letter of the drive on which to create the database.

**Note:** For MPP systems, a database should not be created in an NFS-mounted directory. If a path is not specified, ensure that the *dftdbpath* database manager configuration parameter is not set to an NFS-mounted path (for example, on UNIX based systems, it should not specify the `$HOME` directory of the instance owner). The path specified for this command in an MPP system cannot be a relative path.

**ALIAS database-alias**

An alias for the database in the system database directory. If no alias is provided, the specified database name is used.

**USING CODESET codeset**

Specifies the code set to be used for data entered into this database. After you create the database, you cannot change the specified code set.

**TERRITORY territory**

Specifies the territory to be used for data entered into this database. After you create the database, you cannot change the specified territory.

**COLLATE USING**

Identifies the type of collating sequence to be used for the database. Once the database has been created, the collating sequence cannot be changed.

**COMPATIBILITY**

The DB2 Version 2 collating sequence. Some collation tables have been enhanced. This option specifies that the previous version of these tables is to be used.

**IDENTITY**

Identity collating sequence, in which strings are compared byte for byte.

| **IDENTITY_16BIT**

| CESU-8 (Compatibility Encoding Scheme for UTF-16: 8-Bit)
| collation sequence as specified by the Unicode Technical Report
| #26, which is available at the Unicode Corsortium web site
| (www.unicode.org). This option can only be specified when
| creating a Unicode database.

**UCA400_NO**
The UCA (Unicode Collation Algorithm) collation sequence based on the Unicode Standard version 4.00 with normalization implicitly set to on. Details of the UCA can be found in the Unicode Technical Standard #10, which is available at the Unicode Consortium web site (www.unicode.org). This option can only be used when creating a Unicode database.

**UCA400_LTH**
The UCA (Unicode Collation Algorithm) collation sequence based on the Unicode Standard version 4.00, but will sort all Thai characters according to the Royal Thai Dictionary order. Details of the UCA can be found in the Unicode Technical Standard #10 available at the Unicode Consortium web site (www.unicode.org). This option can only be used when creating a Unicode database. Note that this collator might order Thai data differently from the NLSCHAR collator option.

**NLSCHAR**
System-defined collating sequence using the unique collation rules for the specific code set/territory.

**Note:** This option can only be used with the Thai code page (CP874). If this option is specified in non-Thai environments, the command will fail and return the error SQL1083N with Reason Code 4.

**SYSTEM**
Collating sequence based on the database territory. This option cannot be specified when creating a Unicode database.

**NUMSEGS numsegs**
Specifies the number of segment directories that will be created and used to store DAT, IDX, LF, LB, and LBA files for any default SMS table spaces. This parameter does not affect DMS table spaces, any SMS table spaces with explicit creation characteristics (created when the database is created), or any SMS table spaces explicitly created after the database is created.

**DFT_EXTENT_SZ dft_extentsize**
Specifies the default extent size of table spaces in the database.

**CATALOG TABLESPACE tblspace-defn**
Specifies the definition of the table space which will hold the catalog tables, SYSCATSPACE. If not specified, SYSCATSPACE will be created as a System Managed Space (SMS) table space with *numsegs* number of directories as containers, and with an extent size of *dft_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0000.0
/u/smith/smith/NODE0000/SQL00001/SQLT0000.1
/u/smith/smith/NODE0000/SQL00001/SQLT0000.2
/u/smith/smith/NODE0000/SQL00001/SQLT0000.3
/u/smith/smith/NODE0000/SQL00001/SQLT0000.4
```

In a partitioned database environment, the catalog table space is only created on the catalog database partition (the database partition on which the CREATE DATABASE command is issued).

**USER TABLESPACE tblspace-defn**
Specifies the definition of the initial user table space, USERSPACE1. If not

specified, USERSPACE1 will be created as an SMS table space with *numsegs* number of directories as containers, and with an extent size of *dft_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0001.0
/u/smith/smith/NODE0000/SQL00001/SQLT0001.1
/u/smith/smith/NODE0000/SQL00001/SQLT0001.2
/u/smith/smith/NODE0000/SQL00001/SQLT0001.3
/u/smith/smith/NODE0000/SQL00001/SQLT0001.4
```

**TEMPORARY TABLESPACE tblspace-defn**

Specifies the definition of the initial system temporary table space, TEMPSPACE1. If not specified, TEMPSPACE1 will be created as an SMS table space with *numsegs* number of directories as containers, and with an extent size of *dft_extentsize*. For example, the following containers would be created if *numsegs* were specified to be 5:

```
/u/smith/smith/NODE0000/SQL00001/SQLT0002.0
/u/smith/smith/NODE0000/SQL00001/SQLT0002.1
/u/smith/smith/NODE0000/SQL00001/SQLT0002.2
/u/smith/smith/NODE0000/SQL00001/SQLT0002.3
/u/smith/smith/NODE0000/SQL00001/SQLT0002.4
```

**WITH ″comment-string″**

Describes the database entry in the database directory. Any comment that helps to describe the database can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by single or double quotation marks.

**AUTOCONFIGURE**

Based on user input, calculates the recommended settings for buffer pool size, database configuration, and database manager configuration and optionally applies them.

**USING input-keyword param-value**

*Table 7. Valid input keywords and parameter values*

| Keyword | Valid values | Default value | Explanation |
|---|---|---|---|
| mem_percent | 1–100 | 25 | Percentage of memory to dedicate. If other applications (other than the operating system) are running on this server, set this to less than 100. |
| workload_type | simple, mixed, complex | mixed | Simple workloads tend to be I/O intensive and mostly transactions, whereas complex workloads tend to be CPU intensive and mostly queries. |
| num_stmts | 1–1 000 000 | 25 | Number of statements per unit of work |
| tpm | 1–200 000 | 60 | Transactions per minute |

*Table 7. Valid input keywords and parameter values (continued)*

| Keyword | Valid values | Default value | Explanation |
|---|---|---|---|
| admin_priority | performance, recovery, both | both | Optimize for better performance (more transactions per minute) or better recovery time |
| num_local_apps | 0–5 000 | 0 | Number of connected local applications |
| num_remote_apps | 0–5 000 | 100 | Number of connected remote applications |
| isolation | RR, RS, CS, UR | RR | Isolation level of applications connecting to this database (Repeatable Read, Read Stability, Cursor Stability, Uncommitted Read) |
| bp_resizeable | yes, no | yes | Are buffer pools resizeable? |

**APPLY**

**DB ONLY**
Displays the recommended values for the database configuration and the buffer pool settings based on the current database manager configuration. Applies the recommended changes to the database configuration and the buffer pool settings.

**DB AND DBM**
Displays and applies the recommended changes to the database manager configuration, the database configuration, and the buffer pool settings.

**NONE**
Displays the recommended changes, but does not apply them.

**Usage notes:**

The CREATE DATABASE command:

- Creates a database in the specified subdirectory. In partitioned database environment, creates the database on all database partitions listed in `db2nodes.cfg`, and creates a `$DB2INSTANCE/NODE`*xxxx* directory under the specified subdirectory at each database partition. In a non-partitioned environment, creates a `$DB2INSTANCE/NODE0000` directory under the specified subdirectory.
- Creates the system catalog tables and recovery log.
- Catalogs the database in the following database directories:
  - Server's local database directory on the path indicated by *path* or, if the path is not specified, the default database path defined in the database manager system configuration file by the *dftdbpath* parameter. A local database directory resides on each file system that contains a database.
  - Server's system database directory for the attached instance. The resulting directory entry will contain the database name and a database alias.

If the command was issued from a remote client, the client's system database directory is also updated with the database name and an alias.

Creates a system or a local database directory if neither exists. If specified, the comment and code set values are placed in both directories.

- Stores the specified code set, territory, and collating sequence. A flag is set in the database configuration file if the collating sequence consists of unique weights, or if it is the identity sequence.
- Creates the schemas called SYSCAT, SYSFUN, SYSIBM, and SYSSTAT with SYSIBM as the owner. The database partition server on which this command is issued becomes the catalog database partition for the new database. Two database partition groups are created automatically: IBMDEFAULTGROUP and IBMCATGROUP.
- Binds the previously defined database manager bind files to the database (these are listed in the utilities bind file list, `db2ubind.lst`). If one or more of these files do not bind successfully, CREATE DATABASE returns a warning in the SQLCA, and provides information about the binds that failed. If a bind fails, the user can take corrective action and manually bind the failing file. The database is created in any case. A schema called NULLID is implicitly created when performing the binds with CREATEIN privilege granted to PUBLIC.

  **Note:** The utilities bind file list contains two bind files that cannot be bound against down-level servers:

  - `db2ugtpi.bnd` cannot be bound against DB2 Version 2 servers.
  - `db2dropv.bnd` cannot be bound against DB2 Parallel Edition Version 1 servers.

  If `db2ubind.lst` is bound against a down-level server, warnings pertaining to these two files are returned, and can be disregarded.
- Creates SYSCATSPACE, TEMPSPACE1, and USERSPACE1 table spaces. The SYSCATSPACE table space is only created on the catalog database partition.
- Grants the following:
  - EXECUTE WITH GRANT privilege to PUBLIC on all functions in the SYSFUN schema
  - EXECUTE privilege to PUBLIC on all procedures in SYSIBM schema
  - DBADM authority, and CONNECT, CREATETAB, BINDADD, CREATE_NOT_FENCED, IMPLICIT_SCHEMA and LOAD privileges to the database creator
  - CONNECT, CREATETAB, BINDADD, and IMPLICIT_SCHEMA privileges to PUBLIC
  - USE privilege on the USERSPACE1 table space to PUBLIC
  - SELECT privilege on each system catalog to PUBLIC
  - BIND and EXECUTE privilege to PUBLIC for each successfully bound utility.
  - EXECUTE WITH GRANT privilege to PUBLIC on all functions in the SYSFUN schema.
  - EXECUTE privilege to PUBLIC on all procedures in SYSIBM schema.

With *dbadm* authority, one can grant these privileges to (and revoke them from) other users or PUBLIC. If another administrator with *sysadm* or *dbadm* authority over the database revokes these privileges, the database creator nevertheless retains them.

In an MPP environment, the database manager creates a subdirectory, `$DB2INSTANCE/NODE`*xxxx*, under the specified or default path on all database partitions. The *xxxx* is the database partition number as defined in the `db2nodes.cfg` file (that is, database partition 0 becomes `NODE0000`). Subdirectories `SQL00001` through `SQL`*nnnnn* will reside on this path. This ensures that the database objects associated with different database partitions are stored in different directories (even if the subdirectory `$DB2INSTANCE` under the specified or default path is shared by all database partitions).

If LDAP (Lightweight Directory Access Protocol) support is enabled on the current machine, the database will be automatically registered in the LDAP directory. If a database object of the same name already exists in the LDAP directory, the database is still created on the local machine, but a warning message is returned, indicating that there is a naming conflict. In this case, the user can manually catalog an LDAP database entry by using the CATALOG LDAP DATABASE command.

CREATE DATABASE will fail if the application is already connected to a database.

When a database is created, a detailed deadlocks event monitor is created. As with any monitor, there is some overhead associated with this event monitor. You can drop the deadlocks event monitor by issuing the DROP EVENT MONITOR command.

Use CATALOG DATABASE to define different alias names for the new database.

**Compatibilities:**

For compatibility with versions earlier than Version 8:
- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related concepts:**
- "Isolation levels" in the *SQL Reference, Volume 1*
- "Unicode implementation in DB2 Universal Database" in the *Administration Guide: Planning*

**Related tasks:**
- "Collating Thai characters" in the *Administration Guide: Planning*
- "Creating a database" in the *Administration Guide: Implementation*

**Related reference:**
- "CREATE TABLESPACE statement" in the *SQL Reference, Volume 2*
- "sqlecrea - Create Database" in the *Administrative API Reference*
- "BIND" on page 286
- "CATALOG DATABASE" on page 307
- "DROP DATABASE" on page 352
- "RESTORE DATABASE" on page 647
- "CATALOG LDAP DATABASE" on page 313
- "AUTOCONFIGURE" on page 277

# CREATE TOOLS CATALOG

Creates the DB2 tools catalog tables in a new or existing database. The database must be local.

The tools catalog contains information about the administrative tasks that you configure with such tools as the Task Center and Control Center.

**Note:** This command will optionally force all applications and stop and restart the database manager if new table spaces are created for the tools catalog. It will also update the DB2 Administration Server (DAS) configuration and activate the scheduler.

This command is not valid on a DB2 client.

**Scope:**

The node from which this command is issued becomes the catalog node for the new database.

**Authorization:**

One of the following:
- sysadm
- sysctrl

The user must also have DASADM authority to update the DB2 administration server configuration parameters.

**Required connection:**

A database connection is temporarily established by this command during processing. This command will optionally stop and restart the database manager if new table spaces are created.

**Command syntax:**

```
►►──CREATE TOOLS CATALOG──catalog-name───────────────────────────────────────────────►

►──┬─CREATE NEW DATABASE──database-name───────────────────────────────────────────┬───►
   └─USE EXISTING──┬──────────────────────────────────────┬──DATABASE──database-name─┘
                   └─TABLESPACE──tablespace-name──IN─┘

►──┬───────┬──┬───────────────┬──────────────────────────────────────────────────────►◄
   └─FORCE─┘  └─KEEP INACTIVE─┘
```

**Command parameters:**

**CATALOG catalog-name**
A name to be used to uniquely identify the DB2 tools catalog. The catalog tables are created under this schema name.

**NEW DATABASE database-name**
A name to be assigned to the new database. This must be a unique name that differentiates the database from any other database in either the local database directory or the system database directory. The name must conform to naming conventions for databases.

**EXISTING DATABASE database-name**
The name of an existing database to host the tools catalog. It must be a local database.

**EXISTING TABLESPACE tablespace-name**
A name to be used to specify the existing 32K page table space used to create the DB2 tools catalog tables. A 32K page size temporary table space must also exist for the tables to be created successfully.

**FORCE**
When you create a tools catalog in a new table space, the database manager must be restarted, which requires that no applications be connected. Use the FORCE option to ensure that no applications are connected to the database. If applications are connected, the tools catalog creation will fail unless you specify an existing table space.

**KEEP INACTIVE**
This option will not update the DB2 administration server configuration parameters or enable the scheduler.

**Examples:**

```
db2 create tools catalog cc create new database toolsdb

db2 create tools catalog use existing database toolsdb force

db2 create tools catalog foobar use existing tablespace user32Ksp
 in database toolsdb

db2 create tools catalog toolscat use existing database toolsdb keep inactive
```

**Usage notes:**

- The tools catalog tables require two 32K page table spaces (regular and temporary). In addition, unless you specify existing table spaces, a new 32K buffer pool is created for the table spaces. This requires a restart of the database manager. If the database manager must be restarted, all existing applications must be forced off. The new table spaces are created with a single container each in the default database directory path.
- If an active catalog with this name exists before you execute this command, it is deactivated and the new catalog becomes the active catalog.
- Multiple DB2 tools catalogs can be created in the same database and are uniquely identified by the catalog name.
- The *jdk_path* configuration parameter must be set in the DB2 administration server (DAS) configuration to the minimum supported level of the SDK for Java.
- Updating the DAS configuration parameters requires *dasadm* authority on the DB2 administration server.
- Unless you specify the KEEP INACTIVE option, this command updates the local DAS configuration parameters related to the DB2 tools catalog database configuration and enables the scheduler at the local DAS server.
- The *jdk_64_path* configuration parameter must be set if you are creating a tools catalog against a 64-bit instance on one of the platforms that supports both 32- and 64-bit instances (AIX, HP-UX, and the Solaris Operating Environment).

**Related concepts:**

- “DB2 Administration Server” in the *Administration Guide: Implementation*

**Related reference:**

- “jdk_path - Software Developer's Kit for Java installation path DAS configuration parameter” in the *Administration Guide: Performance*
- “jdk_64_path - 64-Bit Software Developer's Kit for Java installation path DAS configuration parameter” in the *Administration Guide: Performance*

# DEACTIVATE DATABASE

Stops the specified database.

**Scope:**

In an MPP system, this command deactivates the specified database on all database partitions in the system. If one or more of these database partitions encounters an error, a warning is returned. The database will be successfully deactivated on some database partitions, but might continue to be active on the nodes encountering the error.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

None

**Command syntax:**

```
►►──DEACTIVATE──┬─DATABASE─┬──database-alias──────────────────────────────►
                └─DB───────┘

►──┬──────────────────────────────────────┬────────────────────────────►◄
   └─USER──username──┬──────────────────┬─┘
                     └─USING──password──┘
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database to be stopped.

**USER username**
Specifies the user stopping the database.

**USING password**
Specifies the password for the user ID.

**Usage notes:**

Databases initialized by ACTIVATE DATABASE can be shut down by DEACTIVATE DATABASE or by **db2stop**. If a database was initialized by ACTIVATE DATABASE, the last application disconnecting from the database will not shut down the database, and DEACTIVATE DATABASE must be used. (In this case, **db2stop** will also shut down the database.)

**Note:** The application issuing the DEACTIVATE DATABASE command cannot have an active database connection to any database.

**Related reference:**
- "STOP DATABASE MANAGER" on page 698

- “ACTIVATE DATABASE” on page 265

# DEREGISTER

Deregisters the DB2 server from the network directory server.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──DEREGISTER──┬────────────┬──┬────┬──────────────────────────────────────►
                └─DB2 SERVER─┘  └─IN─┘

►─LDAP──NODE──nodename──┬────────────────────────────────────────────┬──────►◄
                        └─USER──username──┬───────────────────────┬─┘
                                          └─PASSWORD──password─┘
```

**Command parameters:**

**IN**
Specifies the network directory server from which to deregister the DB2 server. The valid value is LDAP for an LDAP (Lightweight Directory Access Protocol) directory server.

**USER username**
This is the user's LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to delete the object from the LDAP directory. The user name is optional when deregistering in LDAP. If the user's LDAP DN is not specified, the credentials of the current logon user will be used.

**PASSWORD password**
Account password.

**NODE nodename**
The node name is the value that was specified when the DB2 server was registered in LDAP.

**Usage notes:**

This command can only be issued for a remote machine when in the LDAP environment. When issued for a remote machine, the node name of the remote server must be specified.

The DB2 server is automatically deregistered when the instance is dropped.

**Related reference:**

- "REGISTER" on page 613
- "UPDATE LDAP NODE" on page 738

# DESCRIBE

This command:

- Displays the output SQLDA information about a SELECT or CALL statement
- Displays columns of a table or a view
- Displays indexes of a table or a view

**Authorization:**

To display the output SQLDA information about a SELECT statement, one of the privileges or authorities listed below for each table or view referenced in the SELECT statement is required.

To display the columns or indexes of a table or a view, one of the privileges or authorities listed below for the system catalogs `SYSCAT.COLUMNS` (DESCRIBE TABLE) and `SYSCAT.INDEXES` (DESCRIBE INDEXES FOR TABLE) is required:

- SELECT privilege
- CONTROL privilege
- *sysadm* or *dbadm* authority

As PUBLIC has all the privileges over declared global temporary tables, a user can use the command to display information about any declared global temporary table that exists within its connection.

To display the output SQLDA information about a CALL statement, one of the privileges or authorities listed below is required:

- EXECUTE privilege on the stored procedure
- sysadm or dbadm authority

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
            ┌─OUTPUT─┐
►►──DESCRIBE──┬─┴────────┴──┬─select-statement─┬───────────┬──────────►◄
              │             └─call-statement───┘           │
              ├─TABLE──table-name──────────────────────────┤
              └─INDEXES FOR TABLE──table-name─┘ └─SHOW DETAIL─┘
```

**Command parameters:**

**OUTPUT**
: Indicates that the output of the statement should be described. This keyword is optional.

  **select-statement or call-statement**
  : Identifies the statement about which information is wanted. The statement is automatically prepared by CLP.

**TABLE table-name**
: Specifies the table or view to be described. The fully qualified name in the

form *schema.table-name* must be used. An alias for the table cannot be used in place of the actual table. The *schema* is the user name under which the table or view was created.

The DESCRIBE TABLE command lists the following information about each column:

- Column name
- Type schema
- Type name
- Length
- Scale
- Nulls (yes/no)

**INDEXES FOR TABLE table-name**

Specifies the table or view for which indexes need to be described. The fully qualified name in the form *schema.table-name* must be used. An alias for the table cannot be used in place of the actual table. The *schema* is the user name under which the table or view was created.

The DESCRIBE INDEXES FOR TABLE command lists the following information about each index of the table or view:

- Index schema
- Index name
- Unique rule
- Column count

**SHOW DETAIL**

For the DESCRIBE TABLE command, specifies that output include the following additional information:

- Whether a CHARACTER, VARCHAR or LONG VARCHAR column was defined as FOR BIT DATA
- Column number
- Partitioning key sequence
- Code page
- Default

For the DESCRIBE INDEXES FOR TABLE command, specifies that output include the following additional information:

- Column names

**Examples:**

**Describing the output a SELECT Statement**

The following example shows how to describe a SELECT statement:

```
db2 "describe output select * from staff"
```

```
SQLDA Information

sqldaid :SQLDA sqldabc:896 sqln:20 sqld:7

Column Information

sqltype               sqllen  sqlname.data                    sqlname.length
--------------------  ------  ------------------------------  --------------
500 SMALLINT               2  ID                                           2
449 VARCHAR                9  NAME                                         4
501 SMALLINT               2  DEPT                                         4
453 CHARACTER              5  JOB                                          3
501 SMALLINT               2  YEARS                                        5
485 DECIMAL              7,2  SALARY                                       6
485 DECIMAL              7,2  COMM                                         4
```

### Describing the output a CALL Statement

Given a stored procedure created with the statement:

```
   CREATE PROCEDURE GIVE_BONUS (IN EMPNO INTEGER,
                                IN DEPTNO INTEGER,
                                OUT CHEQUE INTEGER,
                                INOUT BONUS DEC(6,0))
...
```

The following example shows how to describe the output of a CALL statement:

```
 db2 "describe output call give_bonus(123456, 987, ?, 15000.)"

SQLDA Information

sqldaid :SQLDA sqldabc:896 sqln:20 sqld:2

Column Information

sqltype               sqllen  sqlname.data                    sqlname.length
--------------------  ------  ------------------------------  --------------
497 INTEGER                4
485 DECIMAL              6,0
```

### Describing a Table

The following example shows how to describe a table:

```
   db2 describe table user1.department


Table: USER1.DEPARTMENT

Column             Type        Type
name               schema      name               Length   Scale    Nulls
------------------ ----------- ------------------ -------- -------- --------
AREA               SYSIBM      SMALLINT                  2        0 No
DEPT               SYSIBM      CHARACTER                 3        0 No
DEPTNAME           SYSIBM      CHARACTER                20        0 Yes
```

### Describing a Table Index

The following example shows how to describe a table index:

```
   db2 describe indexes for table user1.department
```

```
Table: USER1.DEPARTMENT

Index          Index              Unique         Number of
schema         name               rule           columns
-------------- ------------------ -------------- --------------
USER1          IDX1               U                           2
```

# DETACH

Removes the logical DBMS instance attachment, and terminates the physical communication connection if there are no other logical connections using this layer.

**Authorization:**

None

**Required connection:**

None. Removes an existing instance attachment.

**Command syntax:**

```
►►──DETACH──────────────────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Related reference:**

- “ATTACH” on page 275

# DROP CONTACT

Removes a contact from the list of contacts defined on the local system. A contact is a user to whom the Scheduler and Health Monitor send messages.

**Authorization:**

None.

**Required connection:**

None.

**Command syntax:**

```
►►──DROP CONTACT──name──────────────────────────────────────────────►◄
```

**Command parameters:**

**CONTACT name**
The name of the contact that will be dropped from the local system.

## DROP CONTACTGROUP

Removes a contact group from the list of contacts defined on the local system. A contact group contains a list of users to whom the Scheduler and Health Monitor send messages.

**Authorization:**

None.

**Required Connection:**

None.

**Command Syntax:**

```
►►──DROP CONTACTGROUP──name──────────────────────────────────────────────►◄
```

**Command Parameters:**

**CONTACTGROUP name**
The name of the contact group that will be dropped from the local system.

# DROP DATABASE

Deletes the database contents and all log files for the database, uncatalogs the database, and deletes the database subdirectory.

**Scope:**

By default, this command affects all database partitions that are listed in the `db2nodes.cfg` file.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

**Command syntax:**

```
►►──DROP──┬─DATABASE─┬──database-alias──┬───────────────────┬───────────────────►◄
          └─DB───────┘                  └─AT DBPARTITIONNUM─┘
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database to be dropped. The database must be cataloged in the system database directory.

**AT DBPARTITIONNUM**
Specifies that the database is to be deleted only on the database partition that issued the DROP DATABASE command. This parameter is used by utilities supplied with DB2 ESE, and is not intended for general use. Improper use of this parameter can cause inconsistencies in the system, so it should only be used with caution.

**Examples:**

The following example deletes the database referenced by the database alias SAMPLE:

```
db2 drop database sample
```

**Usage notes:**

DROP DATABASE deletes all user data and log files, as well as any back/restore history for the database. If the log files are needed for a roll-forward recovery after a restore operation, or the backup history required to restore the database, these files should be saved prior to issuing this command.

The database must not be in use; all users must be disconnected from the database before the database can be dropped.

To be dropped, a database must be cataloged in the system database directory. Only the specified database alias is removed from the system database directory. If other aliases with the same database name exist, their entries remain. If the database being dropped is the last entry in the local database directory, the local database directory is deleted automatically.

If DROP DATABASE is issued from a remote client (or from a different instance on the same machine), the specified alias is removed from the client's system database directory. The corresponding database name is removed from the server's system database directory.

This command unlinks all files that are linked through any DATALINK columns. Since the unlink operation is performed asynchronously on the DB2 Data Links Manager, its effects might not be seen immediately on the DB2 Data Links Manager, and the unlinked files might not be immediately available for other operations. When the command is issued, all the DB2 Data Links Managers configured to that database must be available; otherwise, the drop database operation will fail.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "CATALOG DATABASE" on page 307
- "CREATE DATABASE" on page 331
- "UNCATALOG DATABASE" on page 706

# DROP DATALINKS MANAGER

Drops a DB2 Data Links Manager from the list of registered DB2 Data Links Managers for a specified database.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

**Command syntax:**

```
►►──DROP DATALINKS MANAGER FOR──┬─DATABASE─┬──dbname──USING──name──────────────────►◄
                                └─DB───────┘
```

**Command parameters:**

**DATABASE dbname**
Specifies the database name.

**USING name**
Specifies the name of the DB2 Data Links Manager server as shown by the LIST DATALINKS MANAGER command.

**Examples:**

**Example 1**

To drop the DB2 Data Links Manager `micky.almaden.ibm.com` from database TEST under the instance VALIDATE which resides on host `bramha.almaden.ibm.com` when some database tables have links to `micky.almaden.ibm.com`, do the following:

1. Take a database backup for database TEST.
2. If there are any links to `micky.almaden.ibm.com`, unlink them by doing the following:
   a. Log on with a user ID belonging to SYSADM_GROUP and obtain an exclusive mode connection for the database TEST by issuing the following command:

      ```
      connect to test in exclusive mode
      ```

      Ensure that this is the only connection to TEST using that user ID. This will prevent any new links from being created.

   b. Obtain a list of all FILE LINK CONTROL DATALINK columns and the tables containing them in the database by issuing the following command:

      ```
      select tabname, colname from syscat.columns where
      substr(dl_features, 2, 1) = 'F'
      ```

   c. For each FILE LINK CONTROL DATALINK column in the list, issue `SQL SELECT` to determine if links to `micky.almaden.ibm.com` exist. For example, for DATALINK column c in table t, the SELECT statement would be:

      ```
      select count(*) from t where dlurlserver(t.c) = 'MICKY.ALMADEN.IBM.COM'
      ```

d. For each FILE LINK CONTROL DATALINK column containing links, issue SQL UPDATE to unlink values which are links to micky.almaden.ibm.com. For example, for DATALINK column c in table t, the UPDATE statement would be:

```
update t set t.c = null where dlurlserver(t.c) = 'MICKY.ALMADEN.IBM.COM'
```

If t.c is not nullable, the following can be used:

```
update t set t.c = dlvalue('') where dlurlserver(t.c)
= 'MICKY.ALMADEN.IBM.COM'
```

e. Commit the SQL UPDATE:

```
commit
```

3. Issue the DROP DATALINKS MANAGER command:

```
drop datalinks manager for db test using micky.almaden.ibm.com
```

4. Terminate the exclusive mode connection to make the changes effective and to allow other connections to the database:

```
terminate
```

5. Initiate unlink processing and garbage collection of backup information for TEST on micky.almaden.ibm.com. As DB2 Data Links Manager Administrator, issue the following command on micky.almaden.ibm.com:

```
dlfm drop_dlm test validate bramha.almaden.ibm.com
```

This will unlink any files that are still linked to database TEST if the user has missed unlinking them before invoking step 3. If micky.almaden.ibm.com has backup information (for example, archive files, metadata) for files previously linked to database TEST, this command will initiate garbage collection of that information. The actual unlinking and garbage collection will be performed asynchronously.

**Example 2**

A DB2 Data Links Manager can be re-registered after being dropped and it will be treated as a completely new DB2 Data Links Manager. If the steps in Example 1 are followed for dropping micky.almaden.ibm.com, links to the older version will not exist. Otherwise, the user will receive the error SQL0368 as seen in step 7 below. The steps for re-registering the DB2 Data Links Manager are as follows:

1. Register micky.almaden.ibm.com on database TEST:

```
add datalinks manager for db test using node micky.almaden.ibm.com
port 14578
```

2. Create links to files on micky.almaden.ibm.com:

```
connect to test
create table t(c1 int, c2 datalink linktype url file link
control mode db2options)
insert into t
values(1, dlvalue('file://micky.almaden.ibm.com/pictures/yosemite.jpg'))
commit
terminate
```

3. Drop micky.almaden.ibm.com from database TEST:

```
drop datalinks manager for db test using micky.almaden.ibm.com
```

4. Select DATALINK values:

```
connect to test
select * from t
terminate
```

The user will see:

```
SQL0368 The DB2 Data Links Manager "MICKY.ALMADEN.IBM.COM"
is not registered to the database. SQLSTATE=55022.
```

5. Register `micky.almaden.ibm.com` on database TEST again:

   ```
   add datalinks manager for db test using node micky.almaden.ibm.com
   port 14578
   ```

6. Insert more DATALINK values:

   ```
   connect to test
   insert into t
   values(2, dlvalue('file://micky.almaden.ibm.com/pictures/tahoe.jpg'))
   commit
   ```

7. Select DATALINK values:

   ```
   select c2 from t where c1 = 2
   ```

   This command will be successful because the value being selected is a link to the currently registered version of `micky.almaden.ibm.com`.

**Usage notes:**

The effects of the DROP DATALINKS MANAGER command cannot be rolled back. It is important to follow the steps outlined in Example 1 when using the DROP DATALINKS MANAGER command.

This command is effective only after all applications have been disconnected from the database.

Upon successful completion of this command, the DB210201I message will indicate that no processing has been done on the DB2 Data Links Manager.

Before dropping a DB2 Data Links Manager, ensure that the database does not have any links to files on that DB2 Data Links Manager. If links do exist after a DB2 Data Links Manager has been dropped, run the reconcile utility to remove them. This will set nullable links to NULL and non-nullable links to a zero-length DATALINK value. Any row containing these values will be inserted into the exception table. The DATALINK value will not include the original prefix name, which is no longer available after the Data Link Manager has been dropped.

Files corresponding to links between a database and a dropped DB2 Data Links Manager remain in linked state and will be inaccessible to operations such as read, write, rename, delete, change of permissions, or change of ownership.

Archived copies of unlinked files on the DB2 Data Links Manager will not be garbage collected by this command. Users can explicitly initiate unlink processing and garbage collection using the `dlfm drop_dlm` command on the DB2 Data Links Manager.

It is recommended that a database backup be taken before dropping a DB2 Data Links Manager. In addition, ensure that all replication subscriptions have replicated all changes involving this DB2 Data Links Manager.

If a backup was taken before the DB2 Data Links Manager was dropped from a database, and that backup image is used to restore after that DB2 Data Links

Manager was dropped, the restore or rollforward processing might put certain tables in datalink reconcile pending (DRP) state. This will require running the RECONCILE or the **db2_recon_aid** utility to identify and repair any inconsistencies between the DB2 database and the files stored on the Data Links Manager.

**Related reference:**

- "LIST DATALINKS MANAGERS" on page 488
- "ADD DATALINKS MANAGER" on page 269

# DROP DBPARTITIONNUM VERIFY

Verifies if a database partition exists in the database partition groups of any databases, and if an event monitor is defined on the database partition. This command should be used prior to dropping a partition from a partitioned database system.

**Scope:**

This command only affects the database partition on which it is issued.

**Authorization:**

*sysadm*

**Command syntax:**

```
►►──DROP DBPARTITIONNUM VERIFY──────────────────────────────────────►◄
```

**Command parameters:**
None
**Usage notes:**
If a message is returned, indicating that the database partition is not in use, use the STOP DATABASE MANAGER command with DROP DBPARTITIONNUM to remove the entry for the database partition from the `db2nodes.cfg` file, which removes the database partition from the database system.

If a message is returned, indicating that the database partition is in use, the following actions should be taken:

1. If the database partition contains data, redistribute the data to remove it from the database partition using REDISTRIBUTE DATABASE PARTITION GROUP. Use either the DROP DBPARTITIONNUM option on the REDISTRIBUTE DATABASE PARTITION GROUP command or on the ALTER DATABASE PARTITION GROUP statement to remove the database partition from any database partition groups for the database. This must be done for each database that contains the database partition in a database partition group.
2. Drop any event monitors that are defined on the database partition.
3. Rerun DROP DBPARTITIONNUM VERIFY to ensure that the database is no longer in use.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- “STOP DATABASE MANAGER” on page 698
- “REDISTRIBUTE DATABASE PARTITION GROUP” on page 609

# DROP TOOLS CATALOG

Drops the DB2 tools catalog tables for the specified catalog in the given database. This command is not valid on a DB2 client.

**Warning:** If you drop the active tools catalog, you can no longer schedule tasks and scheduled tasks are not executed. To activate the scheduler, you must activate a previous tools catalog or create a new one.

**Scope:**

This command affects the database.

**Authorization:**

One of the following:

- sysadm
- sysctrl

The user must also have DASADM authority to update the DB2 administration server (DAS) configuration parameters.

**Required connection:**

A database connection is temporarily established by this command during processing.

**Command syntax:**

```
►►──DROP TOOLS CATALOG──catalog-name──IN DATABASE──database-name──┬───────┬──────►◄
                                                                  └─FORCE─┘
```

**Command parameters:**

**CATALOG catalog-name**
A name to be used to uniquely identify the DB2 tools catalog. The catalog tables are dropped from this schema.

**DATABASE database-name**
A name to be used to connect to the local database containing the catalog tables.

**FORCE**
The force option is used to force the DB2 administration server's scheduler to stop. If this is not specified, the tools catalog will not be dropped if the scheduler cannot be stopped.

**Examples:**

```
db2 drop tools catalog cc in database toolsdb
    db2 drop tools catalog in database toolsdb force
```

**Usage notes:**

- The *jdk_path* configuration parameter must be set in the DB2 administration server (DAS) configuration to the minimum supported level of the SDK for Java.
- This command will disable the scheduler at the local DAS and reset the DAS configuration parameters related to the DB2 tools catalog database configuration.

# ECHO

Permits the user to write character strings to standard output.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──ECHO──┬──────────────────┬──────────────────────────────────────►◄
          └─character-string─┘
```

**Command parameters:**

**character-string**
Any character string.

**Usage notes:**

If an input file is used as standard input, or comments are to be printed without being interpreted by the command shell, the ECHO command will print character strings directly to standard output.

One line is printed each time that ECHO is issued.

The ECHO command is not affected by the verbose (-v) option.

# EDIT

Launches a user-specified editor with a specified command for editing. When the user finishes editing, saves the contents of the editor and exits the editor, permits the user to execute the command in CLP interactive mode.

**Scope**

This command can only be run within CLP interactive mode. Specifically, it cannot be run from the CLP command mode or the CLP batch mode.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──┬─EDIT─┬──┬───────────────────┬──┬─────┬──────────────────────────────────►◄
    └─E────┘  └─EDITOR──editor─┘  └─num─┘
```

**Command parameters:**

**EDITOR**
: Launch the editor specified for editing. If this parameter is not specified, the editor to be used is determined in the following order:

1. the editor specified by the DB2_CLP_EDITOR registry variable
2. the editor specified by the VISUAL environment variable
3. the editor specified by the EDITOR environment variable
4. On Windows platforms, the Notepad editor; on UNIX-based platforms, the vi editor

**num**
: If *num* is positive, launches the editor with the command corresponding to *num*. If *num* is negative, launches the editor with the command corresponding to *num*, counting backwards from the most recent command in the command history. Zero is not a valid value for *num*. If this parameter is not specified, launches the editor with the most recently run command. (This is equivalent to specifying a value of -1 for *num*.)

**Usage notes:**

1. The editor specified must be a valid editor contained in the PATH of the operating system.
2. You can view a list of the most recently run commands available for editing by executing the HISTORY command.
3. The EDIT command will never be recorded in the command history. However, if you choose to run a command that was edited using the EDIT command, this command will be recorded in the command history.

**Related reference:**

- "HISTORY" on page 448

# EXPORT

Exports data from a database to one of several external file formats. The user specifies the data to be exported by supplying an SQL SELECT statement, or by providing hierarchical information for typed tables.

**Authorization:**

One of the following:

- *sysadm*
- *dbadm*

or CONTROL or SELECT privilege on each participating table or view.

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
>>──EXPORT TO──filename──OF──filetype──┬──────────────────────────┬────────►
                                       │          ┌─,────────┐    │
                                       └─LOBS TO──▼─lob-path─┴────┘

►──┬──────────────────────────┬─┬──────────────────────────────────┬───────►
   │          ┌─,────────┐    │ │              ┌─────────────┐     │
   └─LOBFILE──▼─filename─┴────┘ └─MODIFIED BY──▼─filetype-mod─┴────┘

►──┬────────────────────────────────────┬─┬──────────────────────────┬─────►
   │              ┌─,───────────┐       │ └─MESSAGES──message-file───┘
   └─METHOD N──(──▼─column-name─┴──)────┘

►──┬─select-statement──────────────────────────────────────────────────┬───►◄
   └─HIERARCHY──┬─STARTING──sub-table-name─┬──┬──────────────┬──────────┘
                └─┤ traversal-order-list ├─┘  └─where-clause─┘
```

### traversal-order-list:

```
         ┌─,──────────────┐
├──(─────▼─sub-table-name─┴──)──────────────────────────────────────────┤
```

**Command parameters:**

**HIERARCHY traversal-order-list**
: Export a sub-hierarchy using the specified traverse order. All sub-tables must be listed in PRE-ORDER fashion. The first sub-table name is used as the target table name for the SELECT statement.

**HIERARCHY STARTING sub-table-name**

Using the default traverse order (OUTER order for ASC, DEL, or WSF files, or the order stored in PC/IXF data files), export a sub-hierarchy starting from *sub-table-name*.

**LOBFILE filename**

Specifies one or more base file names for the LOB files. When name space is exhausted for the first name, the second name is used, and so on.

When creating LOB files during an export operation, file names are constructed by appending the current base name from this list to the current path (from *lob-path*), and then appending a 3-digit sequence number. For example, if the current LOB path is the directory `/u/foo/lob/path/`, and the current LOB file name is `bar`, the LOB files created will be `/u/foo/lob/path/bar.001`, `/u/foo/lob/path/bar.002`, and so on.

**LOBS TO lob-path**

Specifies one or more paths to directories in which the LOB files are to be stored. There will be at least one file per LOB path, and each file will contain at least one LOB.

**MESSAGES message-file**

Specifies the destination for warning and error messages that occur during an export operation. If the file already exists, the export utility appends the information. If *message-file* is omitted, the messages are written to standard output.

**METHOD N column-name**

Specifies one or more column names to be used in the output file. If this parameter is not specified, the column names in the table are used. This parameter is valid only for WSF and IXF files, but is not valid when exporting hierarchical data.

**MODIFIED BY filetype-mod**

Specifies file type modifier options. See File type modifiers for export.

**OF filetype**

Specifies the format of the data in the output file:

- DEL (delimited ASCII format), which is used by a variety of database manager and file manager programs.
- WSF (work sheet format), which is used by programs such as:
  - Lotus 1-2-3
  - Lotus Symphony

  **Note:** When exporting BIGINT or DECIMAL data, only values that fall within the range of type DOUBLE can be exported accurately. Although values that do not fall within this range are also exported, importing or loading these values back might result in incorrect data, depending on the operating system.
- IXF (integrated exchange format, PC version), in which most of the table attributes, as well as any existing indexes, are saved in the IXF file, except when columns are specified in the SELECT statement. With this format, the table can be recreated, while with the other file formats, the table must already exist before data can be imported into it.

**select-statement**

Specifies the SELECT statement that will return the data to be exported. If the SELECT statement causes an error, a message is written to the message file (or to standard output). If the error code is one of SQL0012W, SQL0347W, SQL0360W, SQL0437W, or SQL1824W, the export operation continues; otherwise, it stops.

**TO filename**

Specifies the name of the file to which data is to be exported. If the complete path to the file is not specified, the export utility uses the current directory and the default drive as the destination.

If the name of a file that already exists is specified, the export utility overwrites the contents of the file; it does not append the information.

**Examples:**

The following example shows how to export information from the STAFF table in the SAMPLE database to the file `myfile.ixf`. The output will be in IXF format. Note that you must be connected to the SAMPLE database before issuing the command. The index definitions (if any) will be stored in the output file except when the database connection is made through DB2 Connect.

```
db2 export to myfile.ixf of ixf messages msgs.txt select * from staff
```

The following example shows how to export the information about employees in Department 20 from the STAFF table in the SAMPLE database. The output will be in IXF format and will go into the awards.ixf file. Note that you must first connect to the SAMPLE database before issuing the command. Also note that the actual column name in the table is ’dept’ instead of ’department’.

```
db2 export to awards.ixf of ixf messages msgs.txt select * from staff
   where dept = 20
```

The following example shows how to export LOBs to a DEL file:

```
db2 export to myfile.del of del lobs to mylobs/
   lobfile lobs1, lobs2 modified by lobsinfile
   select * from emp_photo
```

The following example shows how to export LOBs to a DEL file, specifying a second directory for files that might not fit into the first directory:

```
db2 export to myfile.del of del
   lobs to /db2exp1/, /db2exp2/ modified by lobsinfile
   select * from emp_photo
```

The following example shows how to export data to a DEL file, using a single quotation mark as the string delimiter, a semicolon as the column delimiter, and a comma as the decimal point. The same convention should be used when importing data back into the database:

```
db2 export to myfile.del of del
   modified by chardel'' coldel; decpt,
   select * from staff
```

**Usage notes:**

Be sure to complete all table operations and release all locks before starting an export operation. This can be done by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK.

Table aliases can be used in the SELECT statement.

The messages placed in the message file include the information returned from the message retrieval service. Each message begins on a new line.

The export utility produces a warning message whenever a character column with a length greater than 254 is selected for export to DEL format files.

PC/IXF import should be used to move data between databases. If character data containing row separators is exported to a delimited ASCII (DEL) file and processed by a text transfer program, fields containing the row separators will shrink or expand.

The file copying step is not necessary if the source and the target databases are both accessible from the same client.

DB2 Connect can be used to export tables from DRDA servers such as DB2 for OS/390, DB2 for VM and VSE, and DB2 for OS/400. Only PC/IXF export is supported.

The export utility will not create multiple-part PC/IXF files when invoked from an AIX system.

The export utility will store the NOT NULL WITH DEFAULT attribute of the table in an IXF file if the SELECT statement provided is in the form `SELECT * FROM tablename`.

When exporting typed tables, subselect statements can only be expressed by specifying the target table name and the WHERE clause. Fullselect and *select-statement* cannot be specified when exporting a hierarchy.

For file formats other than IXF, it is recommended that the traversal order list be specified, because it tells DB2 how to traverse the hierarchy, and what sub-tables to export. If this list is not specified, all tables in the hierarchy are exported, and the default order is the OUTER order. The alternative is to use the default order, which is the order given by the OUTER function.

**Note:** Use the same traverse order during an import operation. The load utility does not support loading hierarchies or sub-hierarchies.

**DB2 Data Links Manager considerations:**

To ensure that a consistent copy of the table and the corresponding files referenced by the DATALINK columns are copied for export, do the following:

1. Issue the command: QUIESCE TABLESPACES FOR TABLE tablename SHARE.

   This ensures that no update transactions are in progress when EXPORT is run.
2. Issue the EXPORT command.
3. Run the **dlfm_export** utility at each Data Links server. Input to the **dlfm_export** utility is the control file name, which is generated by the export utility. This produces a tar (or equivalent) archive of the files listed within the control file.
4. Issue the command: QUIESCE TABLESPACES FOR TABLE tablename RESET.

   This makes the table available for updates.

EXPORT is executed as an SQL application. The rows and columns satisfying the SELECT statement conditions are extracted from the database. For the DATALINK columns, the SELECT statement should not specify any scalar function.

Successful execution of EXPORT results in generation of the following files:

- An export data file as specified in the EXPORT command. A DATALINK column value in this file has the same format as that used by the IMPORT and LOAD utilities. When the DATALINK column value is the SQL NULL value, handling is the same as that for other data types.
- Control files *server_name*, which are generated for each Data Links server. On Windows operating systems, a single control file, `ctrlfile.lst`, is used by all Data Links servers. These control files are placed in the directory <data-file path>/dlfm/YYYYMMDD/HHMMSS (on the Windows NT operating system, `ctrlfile.lst` is placed in the directory <data-file path>\dlfm\YYYYMMDD\HHMMSS). YYYYMMDD represents the date (year month day), and HHMMSS represents the time (hour minute second).

The **dlfm_export** utility is provided to export files from a Data Links server. This utility generates an archive file, which can be used to restore files in the target Data Links server.

**Related concepts:**

- "Export Overview" in the *Data Movement Utilities Guide and Reference*
- "Privileges, authorities and authorization required to use export" in the *Data Movement Utilities Guide and Reference*

**Related tasks:**

- "Using Export" in the *Data Movement Utilities Guide and Reference*

**Related reference:**

- "db2Export - Export" in the *Administrative API Reference*
- "Export Sessions - CLP Examples" in the *Data Movement Utilities Guide and Reference*
- "File type modifiers for export" on page 367
- "Delimiter restrictions for moving data" on page 370

## File type modifiers for export

*Table 8. Valid file type modifiers for export: All file formats*

| Modifier | Description |
|---|---|
| lobsinfile | *lob-path* specifies the path to the files containing LOB data.<br>Each path contains at least one file that contains at least one LOB pointed to by a Lob Location Specifier (LLS) in the data file. The LLS is a string representation of the location of a LOB in a file stored in the LOB file path. The format of an LLS is *filename.ext.nnn.mmm/*, where *filename.ext* is the name of the file that contains the LOB, *nnn* is the offset in bytes of the LOB within the file, and *mmm* is the length of the LOB in bytes. For example, if the string `db2exp.001.123.456/` is stored in the data file, the LOB is located at offset 123 in the file db2exp.001, and is 456 bytes long.<br>If you specify the "lobsinfile" modifier when using EXPORT, the LOB data is placed in the locations specified by the LOBS TO clause. Otherwise the LOB data is sent to the current working directory. The LOBS TO clause specifies one or more paths to directories in which the LOB files are to be stored. There will be at least one file per LOB path, and each file will contain at least one LOB.<br>To indicate a null LOB , enter the size as -1. If the size is specified as 0, it is treated as a 0 length LOB. For null LOBS with length of -1, the offset and the file name are ignored. For example, the LLS of a null LOB might be db2exp.001.7.-1/. |

*Table 9. Valid file type modifiers for export: DEL (delimited ASCII) file format*

| Modifier | Description |
|---|---|
| chardel*x* | *x* is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.[2] If you want to explicitly specify the double quotation mark as the character string delimiter, it should be specified as follows:<br>`modified by chardel""`<br>The single quotation mark (') can also be specified as a character string delimiter as follows:<br>`modified by chardel''` |
| codepage=*x* | *x* is an ASCII character string. The value is interpreted as the code page of the data in the output data set. Converts character data to this code page from the application code page during the export operation.<br>For pure DBCS (graphic), mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive.<br>**Note:** The `codepage` modifier cannot be used with the `lobsinfile` modifier. |
| coldel*x* | *x* is a single character column delimiter. The default value is a comma (,). The specified character is used in place of a comma to signal the end of a column.[2]<br>In the following example, `coldel;` causes the export utility to interpret any semicolon (;) it encounters as a column delimiter:<br>`db2 "export to temp of del modified by coldel;`<br>`   select * from staff where dept = 20"` |
| datesiso | Date format. Causes all date data values to be exported in ISO format (*"YYYY-MM-DD"*).[3] |
| decplusblank | Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign. |

*Table 9. Valid file type modifiers for export: DEL (delimited ASCII) file format (continued)*

| **Modifier** | **Description** |
|---|---|
| decpt*x* | *x* is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.[2] |
| dldel*x* | *x* is a single character DATALINK delimiter. The default value is a semicolon (;). The specified character is used in place of a semicolon as the inter-field separator for a DATALINK value. It is needed because a DATALINK value can have more than one sub-value. [2]<br>**Note:** *x* must not be the same character specified as the row, column, or character string delimiter. |
| nochardel | Column data will not be surrounded by character delimiters. This option should not be specified if the data is intended to be imported or loaded using DB2. It is provided to support vendor data files that do not have character delimiters. Improper usage might result in data loss or corruption.<br>This option cannot be specified with `chardelx` or `nodoubledel`. These are mutually exclusive options. |
| nodoubledel | Suppresses recognition of double character delimiters.[2] |
| striplzeros | Removes the leading zeros from all exported decimal columns.<br>Consider the following example:<br>`db2 create table decimalTable ( c1 decimal( 31, 2 ) )`<br>`db2 insert into decimalTable values ( 1.1 )`<br>`db2 export to data of del select * from decimalTable`<br>`db2 export to data of del modified by STRIPLZEROS`<br>`   select * from decimalTable`<br>In the first export operation, the content of the exported file data will be +00000000000000000000000000001.10. In the second operation, which is identical to the first except for the `striplzeros` modifier, the content of the exported file data will be +1.10. |

*Table 9. Valid file type modifiers for export: DEL (delimited ASCII) file format (continued)*

| Modifier | Description |
|---|---|
| timestampformat="*x*" | *x* is the format of the time stamp in the source file.[4] Valid time stamp elements are:<br>`YYYY   - Year (four digits ranging from 0000 - 9999)`<br>`M      - Month (one or two digits ranging from 1 - 12)`<br>`MM     - Month (two digits ranging from 01 - 12;`<br>`            mutually exclusive with M and MMM)`<br>`MMM    - Month (three-letter case-insensitive abbreviation for`<br>`            the month name; mutually exclusive with M and MM)`<br>`D      - Day (one or two digits ranging from 1 - 31)`<br>`DD     - Day (two digits ranging from 1 - 31; mutually exclusive with D)`<br>`DDD    - Day of the year (three digits ranging from 001 - 366;`<br>`            mutually exclusive with other day or month elements)`<br>`H      - Hour (one or two digits ranging from 0 - 12`<br>`            for a 12 hour system, and 0 - 24 for a 24 hour system)`<br>`HH     - Hour (two digits ranging from 0 - 12`<br>`            for a 12 hour system, and 0 - 24 for a 24 hour system;`<br>`            mutually exclusive with H)`<br>`M      - Minute (one or two digits ranging from 0 - 59)`<br>`MM     - Minute (two digits ranging from 0 - 59;`<br>`            mutually exclusive with M, minute)`<br>`S      - Second (one or two digits ranging from 0 - 59)`<br>`SS     - Second (two digits ranging from 0 - 59;`<br>`            mutually exclusive with S)`<br>`SSSSS  - Second of the day after midnight (5 digits`<br>`            ranging from 00000 - 86399; mutually`<br>`            exclusive with other time elements)`<br>`UUUUUU - Microsecond (6 digits ranging from 000000 - 999999;`<br>`            mutually exclusive with all other microsecond elements)`<br>`UUUUU  - Microsecond (5 digits ranging  from 00000 - 99999,`<br>`            maps to range from 000000 - 999990;`<br>`            mutually exclusive with all other microseond elements)`<br>`UUUU   - Microsecond (4 digits ranging from 0000 - 9999,`<br>`            maps to range from 000000 - 999900;`<br>`            mutually exclusive with all other microseond elements)`<br>`UUU    - Microsecond (3 digits ranging from 000 - 999,`<br>`            maps to range from 000000 - 999000;`<br>`            mutually exclusive with all other microseond elements)`<br>`UU     - Microsecond (2 digits ranging from 00 - 99,`<br>`            maps to range from 000000 - 990000;`<br>`            mutually exclusive with all other microseond elements)`<br>`U      - Microsecond (1 digit ranging from 0 - 9,`<br>`            maps to range from 000000 - 900000;`<br>`            mutually exclusive with all other microseond elements)`<br>`TT     - Meridian indicator (AM or PM)`<br>Following is an example of a time stamp format:<br>`"YYYY/MM/DD HH:MM:SS.UUUUUU"`<br>The MMM element will produce the following values: 'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', and 'Dec'. 'Jan' is equal to month 1, and 'Dec' is equal to month 12.<br>The following example illustrates how to export data containing user-defined time stamp formats from a table called 'schedule':<br>`db2 export to delfile2 of del`<br>`   modified by timestampformat="yyyy.mm.dd hh:mm tt"`<br>`   select * from schedule` |

*Table 10. Valid file type modifiers for export: WSF file format*

| Modifier | Description |
|---|---|
| 1 | Creates a WSF file that is compatible with Lotus 1-2-3 Release 1, or Lotus 1-2-3 Release 1a.[5] This is the default. |
| 2 | Creates a WSF file that is compatible with Lotus Symphony Release 1.0.[5] |
| 3 | Creates a WSF file that is compatible with Lotus 1-2-3 Version 2, or Lotus Symphony Release 1.1.[5] |
| 4 | Creates a WSF file containing DBCS characters. |

**Notes:**

1. The export utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the export operation fails, and an error code is returned.
2. Delimiter restrictions for moving data lists restrictions that apply to the characters that can be used as delimiter overrides.
3. The export utility normally writes
   - date data in *YYYYMMDD* format
   - char(date) data in "*YYYY-MM-DD*" format
   - time data in "*HH.MM.SS*" format
   - time stamp data in "*YYYY-MM-DD-HH. MM.SS.uuuuuu*" format

   Data contained in any datetime columns specified in the SELECT statement for the export operation will also be in these formats.
4. For time stamp formats, care must be taken to avoid ambiguity between the month and the minute descriptors, since they both use the letter M. A month field must be adjacent to other date fields. A minute field must be adjacent to other time fields. Following are some ambiguous time stamp formats:

   ```
   "M" (could be a month, or a minute)
   "M:M" (Which is which?)
   "M:YYYY:M" (Both are interpreted as month.)
   "S:M:YYYY" (adjacent to both a time value and a date value)
   ```

   In ambiguous cases, the utility will report an error message, and the operation will fail.

   Following are some unambiguous time stamp formats:

   ```
   "M:YYYY" (Month)
   "S:M" (Minute)
   "M:YYYY:S:M" (Month....Minute)
   "M:H:YYYY:M:D" (Minute....Month)
   ```
5. These files can also be directed to a specific product by specifying an `L` for Lotus 1-2-3, or an `S` for Symphony in the *filetype-mod* parameter string. Only one value or product designator can be specified.

**Related reference:**

- "db2Export - Export" in the *Administrative API Reference*
- "EXPORT" on page 362
- "Delimiter restrictions for moving data" on page 370

## Delimiter restrictions for moving data

**Delimiter restrictions:**

It is the user's responsibility to ensure that the chosen delimiter character is not part of the data to be moved. If it is, unexpected errors might occur. The following restrictions apply to column, string, DATALINK, and decimal point delimiters when moving data:

- Delimiters are mutually exclusive.
- A delimiter cannot be binary zero, a line-feed character, a carriage-return, or a blank space.
- The default decimal point (.) cannot be a string delimiter.
- The following characters are specified differently by an ASCII-family code page and an EBCDIC-family code page:
  - The Shift-In (0x0F) and the Shift-Out (0x0E) character cannot be delimiters for an EBCDIC MBCS data file.
  - Delimiters for MBCS, EUC, or DBCS code pages cannot be greater than 0x40, except the default decimal point for EBCDIC MBCS data, which is 0x4b.
  - Default delimiters for data files in ASCII code pages or EBCDIC MBCS code pages are:

    ```
    " (0x22, double quotation mark; string delimiter)
    , (0x2c, comma; column delimiter)
    ```
  - Default delimiters for data files in EBCDIC SBCS code pages are:

    ```
    " (0x7F, double quotation mark; string delimiter)
    , (0x6B, comma; column delimiter)
    ```
  - The default decimal point for ASCII data files is 0x2e (period).
  - The default decimal point for EBCDIC data files is 0x4B (period).
  - If the code page of the server is different from the code page of the client, it is recommended that the hex representation of non-default delimiters be specified. For example,

    ```
    db2 load from ... modified by chardel0x0C coldelX1e ...
    ```

The following information about support for double character delimiter recognition in DEL files applies to the export, import, and load utilities:

- Character delimiters are permitted within the character-based fields of a DEL file. This applies to fields of type CHAR, VARCHAR, LONG VARCHAR, or CLOB (except when `lobsinfile` is specified). Any pair of character delimiters found between the enclosing character delimiters is imported or loaded into the database. For example,

  ```
  "What a ""nice"" day!"
  ```

  will be imported as:

  ```
  What a "nice" day!
  ```

  In the case of export, the rule applies in reverse. For example,

  ```
  I am 6" tall.
  ```

  will be exported to a DEL file as:

  ```
  "I am 6"" tall."
  ```
- In a DBCS environment, the pipe (|) character delimiter is not supported.

# FORCE APPLICATION

Forces local or remote users or applications off the system to allow for maintenance on a server.

**Attention:** If an operation that cannot be interrupted (RESTORE DATABASE, for example) is forced, the operation must be successfully re-executed before the database becomes available.

**Scope:**

This command affects all database partitions that are listed in the `$HOME/sqllib/db2nodes.cfg` file.

In a partitioned database environment, this command does not have to be issued from the coordinator database partition of the application being forced. It can be issued from any node (database partition server) in the partitioned database environment.

**Authorization:**

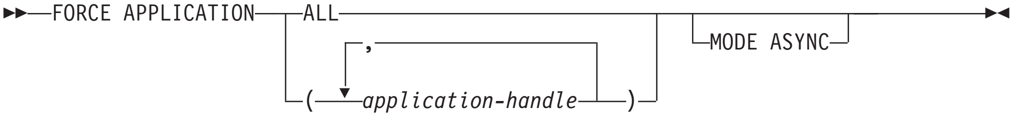
One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance. To force users off a remote server, it is first necessary to attach to that server. If no attachment exists, this command is executed locally.

**Command syntax:**



**Command parameters:**

**APPLICATION**

**ALL** All applications will be disconnected from the database.

**application-handle**
Specifies the agent to be terminated. List the values using the LIST APPLICATIONS command.

**MODE ASYNC**
The command does not wait for all specified users to be terminated before returning; it returns as soon as the function has been successfully issued or an error (such as invalid syntax) is discovered.

This is the only mode that is currently supported.

**Examples:**

The following example forces two users, with *application-handle* values of 41408 and 55458, to disconnect from the database:

```
   db2 force application ( 41408, 55458 )
```

**Usage notes:**

**db2stop** cannot be executed during a force. The database manager remains active so that subsequent database manager operations can be handled without the need for **db2start**.

To preserve database integrity, only users who are idling or executing interruptible database operations can be terminated.

Users creating a database cannot be forced.

After a FORCE has been issued, the database will still accept requests to connect. Additional forces might be required to completely force all users off.

**Related reference:**

- "LIST APPLICATIONS" on page 480
- "ATTACH" on page 275

# GET ADMIN CONFIGURATION

Returns the values of individual DB2 Administration Server (DAS) configuration parameter values on the administration node of the system. The DAS is a special administrative tool that enables remote administration of DB2 servers For a list of the DAS configuration parameters, see the description of the UPDATE ADMIN CONFIGURATION command.

**Scope:**

This command returns information about DAS configuration parameters on the administration node of the system to which you are attached or that you specify in the FOR NODE option.

**Authorization:**

None

**Required connection:**

Node. To display the DAS configuration for a remote system, first connect to that system or use the FOR NODE option to specify the administration node of the system.

**Command syntax:**

```
►►──GET ADMIN──┬─CONFIGURATION─┬──────────────────────────────────────────────►
               ├─CONFIG────────┤
               └─CFG───────────┘

►──┬──────────────────────────────────────────────────────────┬───────────────►◄
   └─FOR NODE──node-name──┬────────────────────────────────┬──┘
                          └─USER──username──USING──password─┘
```

**Command parameters:**

**FOR NODE**
: Enter the name of a the administration node to view DAS configuration parameters there.

**USER username USING** *password*
: If connection to the node requires user name and password, enter this information.

**Examples:**

The following is sample output from GET ADMIN CONFIGURATION:

```
            Admin Server Configuration

 Authentication Type DAS                (AUTHENTICATION) = SERVER_ENCRYPT

 DAS Administration Authority Group Name  (DASADM_GROUP) = ADMINISTRATORS

 DAS Discovery Mode                           (DISCOVER) = SEARCH
 Name of the DB2 Server System               (DB2SYSTEM) = swalkty

 Java Development Kit Installation Path DAS   (JDK_PATH) = e:\sqllib\java\jdk

 DAS Code Page                            (DAS_CODEPAGE) = 0
 DAS Territory                           (DAS_TERRITORY) = 0

 Location of Contact List                 (CONTACT_HOST) = hostA.ibm.ca
 Execute Expired Tasks                   (EXEC_EXP_TASK) = NO
 Scheduler Mode                           (SCHED_ENABLE) = ON
 SMTP Server                               (SMTP_SERVER) = smtp1.ibm.ca
 Tools Catalog Database                    (TOOLSCAT_DB) = CCMD
 Tools Catalog Database Instance         (TOOLSCAT_INST) = DB2
 Tools Catalog Database Schema         (TOOLSCAT_SCHEMA) = TOOLSCAT
 Scheduler User ID                                       = db2admin
```

**Usage notes:**

If an error occurs, the information returned is not valid. If the configuration file is invalid, an error message is returned. The user must install the DAS again to recover.

To set the configuration parameters to the default values shipped with the DAS, use the RESET ADMIN CONFIGURATION command.

**Related reference:**

- "RESET ADMIN CONFIGURATION" on page 635
- "UPDATE ADMIN CONFIGURATION" on page 715
- "Configuration parameters summary" in the *Administration Guide: Performance*

# GET ALERT CONFIGURATION

Returns the alert configuration settings for health indicators for a particular instance.

**Authorization:**

None.

**Required connection:**

Instance. An explicit attachment is not required.

**Command syntax:**

```
►►─GET ALERT──┬─CONFIGURATION─┬──FOR──────────────────────────────────────►
              ├─CONFIG────────┤
              └─CFG───────────┘

►──┬─┬─┬─DATABASE MANAGER─┬─┬──┬─────────┬─┬──────────────────────────────►
   │ │ ├─DB MANAGER───────┤ │  └─DEFAULT─┘ │
   │ │ └─DBM──────────────┘ │              │
   │ ├─DATABASES────────────┤              │
   │ ├─CONTAINERS───────────┤              │
   │ └─TABLESPACES──────────┘              │
   └─┬─DATABASE─────────────────────────┬─ON─database alias─┘
     ├─TABLESPACE─name──────────────────┤
     └─CONTAINER─name─FOR─tablespace-id─┘

►──┬───────────────────────────────────┬──────────────────────────────►◄
   │          ┌─,─────────────────────┐ │
   │          ▼                       │ │
   └─USING────┴─health indicator name─┴─┘
```

**Command parameters:**

**DATABASE MANAGER**
Retrieves alert settings for the database manager.

**DATABASES**
Retrieves alert settings for all databases managed by the database manager. These are the settings that apply to all databases that do not have custom settings. Custom settings are defined using the DATABASE ON *database alias* clause.

**CONTAINERS**
Retrieves alert settings for all table space containers managed by the database manager. These are the settings that apply to all table space containers that do not have custom settings. Custom settings are defined using the "CONTAINER *name* ON *database alias*" clause.

**TABLESPACES**
Retrieves alert settings for all table spaces managed by the database manager. These are the settings that apply to all table spaces that do not have custom settings. Custom settings are defined using the TABLESPACE *name* ON *database alias* clause.

**DEFAULT**
Specifies that the install defaults are to be retrieved.

**DATABASE ON database alias**

Retrieves the alert settings for the database specified using the ON *database alias* clause. If this database does not have custom settings, then the settings for all databases for the instance will be returned, which is equivalent to using the DATABASES parameter.

**CONTAINER** *name* **FOR** *tablespace-id* **ON** *database alias*

Retrieves the alert settings for the table space container called *name*, for the table space specified using the ″FOR tablespace-id″ clause, on the database specified using the ″ON database alias″ clause. If this table space container does not have custom settings, then the settings for all table space containers for the database will be returned, which is equivalent to using the CONTAINERS parameter.

**TABLESPACE** *name* **ON** *database alias*

Retrieves the alert settings for the table space called *name*, on the database specified using the ON *database alias* clause. If this table space does not have custom settings, then the settings for all table spaces for the database will be returned, which is equivalent to using the TABLESPACES parameter.

**USING health indicator name**

Specifies the set of health indicators for which alert configuration information will be returned. Health indicator names consist of a two-letter object identifier followed by a name that describes what the indicator measures. For example: db.sort_privmem_util. This is an optional clause, meaning that if it is not used, all health indicators for the specified object or object type will be returned.

**Examples:**

The following is typical output resulting from a request for database manager information:

```
DB2 GET ALERT CFG FOR DBM

            Alert Configuration
  Indicator Name                    = db2.db2_op_status
      Default                       = Yes
      Type                          = State-based
      Sensitivity                   = 0
      Formula                       = db2.db2_status;
      Actions                       = Disabled
      Threshold or State checking   = Enabled

  Indicator Name                    = db2.sort_privmem_util
      Default                       = Yes
      Type                          = Threshold-based
      Warning                       = 90
      Alarm                         = 100
      Unit                          = %
      Sensitivity                   = 0
      Formula                       = ((db2.sort_heap_allocated/sheapthres)
                                       *100);
      Actions                       = Disabled
      Threshold or State checking   = Enabled

  Indicator Name                    = db2.mon_heap_util
      Default                       = Yes
      Type                          = Threshold-based
      Warning                       = 85
      Alarm                         = 95
      Unit                          = %
      Sensitivity                    = 0
      Formula                        = ((db2.mon_heap_cur_size/
                                         db2.mon_heap_max_size)*100);
      Actions                        = Disabled
      Threshold or State checking    = Enabled
```

The following is typical output resulting from a request for configuration information:

```
DB2 GET ALERT CFG FOR DATABASES

            Alert Configuration
  Indicator Name                     = db.db_op_status
      Default                        = Yes
      Type                           = State-based
      Sensitivity                    = 0
      Formula                        = db.db_status;
      Actions                        = Disabled
      Threshold or State checking    = Enabled

  Indicator Name                     = db.sort_shrmem_util
      Default                        = Yes
      Type                           = Threshold-based
      Warning                        = 70
      Alarm                          = 85
      Unit                           = %
      Sensitivity                    = 0
      Formula                        = ((db.sort_shrheap_allocated/sheapthres_shr)
                                        *100);
      Actions                        = Disabled
      Threshold or State checking    = Enabled

  Indicator Name                     = db.spilled_sorts
      Default                        = Yes
      Type                           = Threshold-based
      Warning                        = 30
      Alarm                          = 50
      Unit                           = %
      Sensitivity                    = 0
      Formula                        = ((delta(db.sort_overflows,10))/
                                       (delta(db.total_sorts,10)+1)*100);
      Actions                        = Disabled
      Threshold or State checking    = Enabled

  Indicator Name                     = db.max_sort_shrmem_util
      Default                        = Yes
      Type                           = Threshold-based
      Warning                        = 60
      Alarm                          = 30
      Unit                           = %
      Sensitivity                    = 0
      Formula                        = ((db.max_shr_sort_mem/
                                         sheapthres_shr)*100);
      Actions                        = Disabled
      Threshold or State checking    = Enabled

  Indicator Name                     = db.log_util
      Default                        = Yes
      Type                           = Threshold-based
      Warning                        = 75
      Alarm                          = 85
      Unit                           = %
      Sensitivity                    = 0
      Formula                        = (db.total_log_used/
                                        (db.total_log_used+db.total_log_available)
                                         )*100;
      Actions                        = Disabled
```

```
    Threshold or State checking    = Enabled

Indicator Name                     = db.log_fs_util
    Default                        = Yes
    Type                           = Threshold-based
    Warning                        = 75
    Alarm                          = 85
    Unit                           = %
    Sensitivity                    = 0
    Formula                        = ((os.fs_used/os.fs_total)*100);
    Actions                        = Disabled
    Threshold or State checking    = Enabled

Indicator Name                     = db.deadlock_rate
    Default                        = Yes
    Type                           = Threshold-based
    Warning                        = 5
    Alarm                          = 10
    Unit                           = Deadlocks per hour
    Sensitivity                    = 0
    Formula                        = delta(db.deadlocks);
    Actions                        = Disabled
    Threshold or State checking    = Enabled

Indicator Name                     = db.locklist_util
    Default                        = Yes
    Type                           = Threshold-based
    Warning                        = 75
    Alarm                          = 85
    Unit                           = %
    Sensitivity                    = 0
    Formula                        = (db.lock_list_in_use/(locklist*4096))
                                      *100;
    Actions                        = Disabled
    Threshold or State checking    = Enabled

Indicator Name                     = db.lock_escal_rate
    Default                        = Yes
    Type                           = Threshold-based
    Warning                        = 5
    Alarm                          = 10
    Unit                           = Lock escalations per hour
    Sensitivity                    = 0
    Formula                        = delta(db.lock_escals);
    Actions                        = Disabled
    Threshold or State checking    = Enabled

Indicator Name                     = db.apps_waiting_locks
    Default                        = Yes
    Type                           = Threshold-based
    Warning                        = 50
    Alarm                          = 70
    Unit                           = %
    Sensitivity                    = 0
    Formula                        = (db.locks_waiting/db.appls_cur_cons)*100;

    Actions                        = Disabled
    Threshold or State checking    = Enabled

Indicator Name                     = db.pkgcache_hitratio
    Default                        = Yes
    Type                           = Threshold-based
    Warning                        = 80
    Alarm                          = 70
    Unit                           = %
    Sensitivity                    = 0
    Formula                        = (1-
                                      (db.pkg_cache_inserts/db.pkg_cache_lookups)
                                      )*100;
    Actions                         = Disabled
    Threshold or State checking     = Disabled

Indicator Name                      = db.catcache_hitratio
    Default                         = Yes
    Type                            = Threshold-based
    Warning                         = 80
    Alarm                           = 70
    Unit                            = %
    Sensitivity                     = 0
    Formula                         = (1-
                                      (db.cat_cache_inserts/db.cat_cache_lookups)
                                      )*100;
    Actions                         = Disabled
    Threshold or State checking     = Disabled

Indicator Name                      = db.shrworkspace_hitratio
    Default                         = Yes
    Type                            = Threshold-based
    Warning                         = 80
    Alarm                           = 70
    Unit                            = %
    Sensitivity                     = 0
    Formula                         = ((1-
                                      (db.shr_workspace_section_inserts/
                                      db.shr_workspace_section_lookups))
                                      *100);
    Actions                         = Disabled
    Threshold or State checking     = Disabled

Indicator Name                      = db.db_heap_util
    Default                         = Yes
    Type                            = Threshold-based
    Warning                         = 85
    Alarm                           = 95
    Unit                            = %
    Sensitivity                     = 0
    Formula                         = ((db.db_heap_cur_size/
                                       db.db_heap_max_size)*100);
    Actions                         = Disabled
    Threshold or State checking     = Enabled

Indicator Name                      = db.tb_reorg_req
    Default                         = Yes
    Type                            = Collection state-based
    Sensitivity                     = 0
    Actions                         = Disabled
    Threshold or State checking     = Disabled

Indicator Name                      = db.hadr_op_status
    Default                         = Yes
    Type                            = State-based
    Sensitivity                     = 0
    Formula                         = db.hadr_connect_status;
    Actions                         = Disabled
    Threshold or State checking     = Enabled

Indicator Name                      = db.hadr_delay
    Default                         = Yes
    Type                            = Threshold-based
    Warning                         = 10
    Alarm                           = 15
    Unit                            = Minutes
    Sensitivity                     = 0
    Formula                         = (db.hadr_log_gap*var.refresh_rate/60)
```

```
                                       DIV(delta(db.hadr_secondary_log_pos));
    Actions                          = Disabled
    Threshold or State checking      = Enabled

Indicator Name                       = db.db_backup_req
    Default                          = Yes
    Type                             = State-based
    Sensitivity                      = 0
    Actions                          = Disabled
    Threshold or State checking      = Disabled

Indicator Name                       = db.fed_nicknames_op_status
    Default                          = Yes
    Type                             = Collection state-based
    Sensitivity                      = 0
    Actions                          = Disabled
    Threshold or State checking      = Disabled

Indicator Name                       = db.fed_servers_op_status
    Default                          = Yes
    Type                             = Collection state-based
    Sensitivity                      = 0
    Actions                          = Disabled
    Threshold or State checking      = Disabled

Indicator Name                       = db.tb_runstats_req
    Default                          = Yes
    Type                             = Collection state-based
    Sensitivity                      = 0
    Actions                          = Disabled
    Threshold or State checking      = Disabled
```

# GET AUTHORIZATIONS

Reports the authorities of the current user from values found in the database configuration file and the authorization system catalog view (SYSCAT.DBAUTH).

**Authorization:**

None

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
►►──GET AUTHORIZATIONS─────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Examples:**
The following is sample output from GET AUTHORIZATIONS:

```
 Administrative Authorizations for Current User

 Direct SYSADM authority                    = NO
 Direct SYSCTRL authority                   = NO
 Direct SYSMAINT authority                  = NO
 Direct DBADM authority                     = YES
 Direct CREATETAB authority                 = YES
 Direct BINDADD authority                   = YES
 Direct CONNECT authority                   = YES
 Direct CREATE_NOT_FENC authority           = YES
 Direct IMPLICIT_SCHEMA authority           = YES
 Direct LOAD authority                      = YES
 Direct QUIESCE_CONNECT authority           = YES
 Direct CREATE_EXTERNAL_ROUTINE authority   = YES

 Indirect SYSADM authority                  = YES
 Indirect SYSCTRL authority                 = NO
 Indirect SYSMAINT authority                = NO
 Indirect DBADM authority                   = NO
 Indirect CREATETAB authority               = YES
 Indirect BINDADD authority                 = YES
 Indirect CONNECT authority                 = YES
 Indirect CREATE_NOT_FENC authority         = NO
 Indirect IMPLICIT_SCHEMA authority         = YES
 Indirect LOAD authority                    = NO
 Indirect QUIESCE_CONNECT authority         = NO
 Indirect CREATE_EXTERNAL_ROUTINE authority = NO
```

**Usage notes:**

Direct authorities are acquired by explicit commands that grant the authorities to a user ID. Indirect authorities are based on authorities acquired by the groups to which a user belongs.

**Note:** PUBLIC is a special group to which all users belong.

# GET CLI CONFIGURATION

Lists the contents of the db2cli.ini file. This command can list the entire file, or a specified section.

The db2cli.ini file is used as the DB2 call level interface (CLI) configuration file. It contains various keywords and values that can be used to modify the behavior of the DB2 CLI and the applications using it. The file is divided into sections, each section corresponding to a database alias name.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──GET CLI──┬─CONFIGURATION─┬──┬─────────────────┬──────────────────────────►
             ├─CONFIG────────┤  └─AT GLOBAL LEVEL─┘
             └─CFG───────────┘

►──┬──────────────────────────────┬─────────────────────────────────────────►◄
   └─FOR SECTION──section-name─┘
```

**Command parameters:**

**AT GLOBAL LEVEL**
Displays the default CLI configuration parameters in the LDAP directory.

**Note:** This parameter is only valid on Windows operating systems.

**FOR SECTION section-name**
Name of the section whose keywords are to be listed. If not specified, all sections are listed.

**Examples:**

The following sample output represents the contents of a db2cli.ini file that has two sections:

```
[tstcli1x]
uid=userid
pwd=password
autocommit=0
TableType="'TABLE','VIEW','SYSTEM TABLE'"

[tstcli2x]
SchemaList="'OWNER1','OWNER2',CURRENT SQLID"
```

**Usage notes:**

The section name specified on this command is not case sensitive. For example, if the section name in the db2cli.ini file (delimited by square brackets) is in lowercase, and the section name specified on the command is in uppercase, the correct section will be listed.

The value of the PWD (password) keyword is never listed; instead, five asterisks (*****) are listed.

When LDAP (Lightweight Directory Access Protocol) is enabled, the CLI configuration parameters can be set at the user level, in addition to the machine level. The CLI configuration at the user level is maintained in the LDAP directory. If the specified section exists at the user level, the CLI configuration for that section at the user level is returned; otherwise, the CLI configuration at the machine level is returned.

The CLI configuration at the user level is maintained in the LDAP directory and cached on the local machine. When reading the CLI configuration at the user level, DB2 always reads from the cache. The cache is refreshed when:

- The user updates the CLI configuration.
- The user explicitly forces a refresh of the CLI configuration using the REFRESH LDAP command.

In an LDAP environment, users can configure a set of default CLI settings for a database catalogued in the LDAP directory. When an LDAP catalogued database is added as a Data Source Name (DSN), either by using the Client Configuration Assistant (CCA) or the CLI/ODBC configuration utility, any default CLI settings, if they exist in the LDAP directory, will be configured for that DSN on the local machine. The AT GLOBAL LEVEL clause must be specified to display the default CLI settings.

**Related reference:**

- “UPDATE CLI CONFIGURATION” on page 724
- “REFRESH LDAP” on page 612

# GET CONNECTION STATE

Displays the connection state. Possible states are:

- Connectable and connected
- Connectable and unconnected
- Unconnectable and connected
- Implicitly connectable (if implicit connect is available).

This command also returns information about:

- the database connection mode (SHARE or EXCLUSIVE)
- the alias and name of the database to which a connection exists (if one exists)
- the host name and service name of the connection if the connection is using TCP/IP

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──GET CONNECTION STATE──────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Examples:**
The following is sample output from GET CONNECTION STATE:

```
   Database Connection State

 Connection state       = Connectable and Connected
 Connection mode        = SHARE
 Local database alias   = SAMPLE
 Database name          = SAMPLE
 Hostname               = montero
 Service name           = 29384
```

**Usage notes:**

This command does not apply to type 2 connections.

**Related reference:**

- "SET CLIENT" on page 678
- "UPDATE ALTERNATE SERVER FOR DATABASE" on page 721

# GET CONTACTGROUP

Returns the contacts included in a single contact group that is defined on the local system. A contact is a user to whom the Scheduler and Health Monitor send messages. You create named groups of contacts with the ADD CONTACTGROUP command.

**Authorization:**

None.

**Required connection:**

None. Local execution only: this command cannot be used with a remote connection.

**Command syntax:**

```
►►──GET CONTACTGROUP──name──────────────────────────────────────────────────►◄
```

**Command parameters:**

**CONTACTGROUP name**
The name of the group for which you would like to retrieve the contacts.

**Examples:**

GET CONTACTGROUP support

```
Description
-------------
Foo Widgets broadloom support unit

Name           Type
-------------  --------------
joe            contact
support        contact group
joline         contact
```

# GET CONTACTGROUPS

The command provides a list of contact groups, which can be either defined locally on the system or in a global list. A contact group is a list of addresses to which monitoring processes such as the Scheduler and Health Monitor can send messages. The setting of the Database Administration Server (DAS) *contact_host* configuration parameter determines whether the list is local or global. You create named groups of contacts with the ADD CONTACTGROUP command.

**Authorization:**

None

**Required Connection:**

None

**Command Syntax:**

```
►►──GET CONTACTGROUPS──────────────────────────────────────────────────────►◄
```

**Command Parameters:**
None
**Examples:**
In the following example, the command GET CONTACTGROUPS is issued. The result is as follows:

```
Name        Description
---------   --------------
support     Foo Widgets broadloom support unit
service     Foo Widgets service and support unit
```

# GET CONTACTS

Returns the list of contacts defined on the local system. Contacts are users to whom the monitoring processes such as the Scheduler and Health Monitor send notifications or messages.

To create a contact, use the ADD CONTACT command.

**Authorization:**

None.

**Required connection:**

None.

**Command syntax:**

```
►►──GET CONTACTS──────────────────────────────────────────────────────►◄
```

**Examples:**
GET CONTACTS

```
Name    Type    Address           Max Page Length  Description
------- ------  ----------        ---------------  ------------
joe     e-mail  joe@somewhere.com -                -
joline  e-mail  joline@           -                -
                somewhereelse.com
john    page    john@relay.org    50               Support 24x7
```

# GET DATABASE CONFIGURATION

Returns the values of individual entries in a specific database configuration file.

**Scope:**

This command returns information only for the partition on which it is executed.

**Authorization:**

None

**Required connection:**

Instance. An explicit attachment is not required, but a connection to the database is required when using the SHOW DETAIL clause. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

**Command syntax:**

```
►►──GET──┬─DATABASE─┬──┬─CONFIGURATION─┬──┬──────────────────────────┬──────────►
         └─DB───────┘  ├─CONFIG────────┤  └─FOR──database-alias──┘
                       └─CFG───────────┘

►──┬─────────────┬───────────────────────────────────────────────────────────►◄
   └─SHOW DETAIL─┘
```

**Command parameters:**

**FOR database-alias**
Specifies the alias of the database whose configuration is to be displayed. You do not need to specify the alias if a connection to the database already exists.

**SHOW DETAIL**
Displays detailed information showing the current value of database configuration parameters as well as the value of the parameters the next time you activate the database. This option lets you see the result of dynamic changes to configuration parameters.

**Examples:**

**Notes:**

1. Output on different platforms might show small variations reflecting platform-specific parameters.
2. Parameters with keywords enclosed by parentheses can be changed by the UPDATE DATABASE CONFIGURATION command.
3. Fields that do not contain keywords are maintained by the database manager and cannot be updated.

The following is sample output from GET DATABASE CONFIGURATION (issued on AIX):

```
                    Database Configuration for Database mick

 Database configuration release level                    = 0x0a00
 Database release level                                  = 0x0a00
```

```
Database territory                                      = en_US
Database code page                                      = 819
Database code set                                       = ISO8859-1
Database country/region code                            = 1
Database collating sequence                             = UNIQUE
Alternate collating sequence              (ALT_COLLATE) =

Dynamic SQL Query management           (DYN_QUERY_MGMT) = DISABLE

Discovery support for this database       (DISCOVER_DB) = ENABLE

Default query optimization class         (DFT_QUERYOPT) = 5
Degree of parallelism                      (DFT_DEGREE) = 1
Continue upon arithmetic exceptions   (DFT_SQLMATHWARN) = NO
Default refresh age                   (DFT_REFRESH_AGE) = 0
Default maintained table types for opt (DFT_MTTB_TYPES) = SYSTEM
Number of frequent values retained     (NUM_FREQVALUES) = 10
Number of quantiles retained            (NUM_QUANTILES) = 20

Backup pending                                          = NO

Database is consistent                                  = YES
Rollforward pending                                     = NO
Restore pending                                         = NO

Multi-page file allocation enabled                      = YES

Log retain for recovery status                          = NO
User exit for logging status                            = NO

Data Links Token Expiry Interval (sec)      (DL_EXPINT) = 60
Data Links Write Token Init Expiry Intvl(DL_WT_IEXPINT) = 60
Data Links Number of Copies             (DL_NUM_COPIES) = 1
Data Links Time after Drop (days)        (DL_TIME_DROP) = 1
Data Links Token in Uppercase                (DL_UPPER) = NO
Data Links Token Algorithm                   (DL_TOKEN) = MAC0

Database heap (4KB)                            (DBHEAP) = 1200
Size of database shared memory (4KB)  (DATABASE_MEMORY) = AUTOMATIC
Catalog cache size (4KB)              (CATALOGCACHE_SZ) = 64
Log buffer size (4KB)                        (LOGBUFSZ) = 8
Utilities heap size (4KB)                (UTIL_HEAP_SZ) = 5000
Buffer pool size (pages)                     (BUFFPAGE) = 1000
Extended storage segments size (4KB)    (ESTORE_SEG_SZ) = 16000
Number of extended storage segments   (NUM_ESTORE_SEGS) = 0
Max storage for lock list (4KB)              (LOCKLIST) = 128

Max size of appl. group mem set (4KB) (APPGROUP_MEM_SZ) = 30000
Percent of mem for appl. group heap   (GROUPHEAP_RATIO) = 70
Max appl. control heap size (4KB)     (APP_CTL_HEAP_SZ) = 128

Sort heap thres for shared sorts (4KB) (SHEAPTHRES_SHR) = (SHEAPTHRES)
Sort list heap (4KB)                         (SORTHEAP) = 256
SQL statement heap (4KB)                     (STMTHEAP) = 2048
Default application heap (4KB)             (APPLHEAPSZ) = 128
Package cache size (4KB)                   (PCKCACHESZ) = (MAXAPPLS*8)
Statistics heap size (4KB)               (STAT_HEAP_SZ) = 4384

Interval for checking deadlock (ms)         (DLCHKTIME) = 10000
Percent. of lock lists per application       (MAXLOCKS) = 10
Lock timeout (sec)                        (LOCKTIMEOUT) = -1

Changed pages threshold                (CHNGPGS_THRESH) = 60
Number of asynchronous page cleaners   (NUM_IOCLEANERS) = 1
Number of I/O servers                   (NUM_IOSERVERS) = 3
Index sort flag                             (INDEXSORT) = YES
```

```
Sequential detect flag                      (SEQDETECT) = YES
Default prefetch size (pages)         (DFT_PREFETCH_SZ) = AUTOMATIC

Track modified pages                         (TRACKMOD) = OFF

Default number of containers                            = 1
Default tablespace extentsize (pages)   (DFT_EXTENT_SZ) = 32

Max number of active applications            (MAXAPPLS) = AUTOMATIC
Average number of active applications       (AVG_APPLS) = 1
Max DB files open per application            (MAXFILOP) = 64

Log file size (4KB)                         (LOGFILSIZ) = 1000
Number of primary log files                (LOGPRIMARY) = 3
Number of secondary log files               (LOGSECOND) = 2
Changed path to log files                  (NEWLOGPATH) =
Path to log files                                       = /home/db2inst/db2inst
                                                           /NODE0000/SQL00001
                                                           /SQLOGDIR/
Overflow log path                     (OVERFLOWLOGPATH) =
Mirror log path                         (MIRRORLOGPATH) =
First active log file                                   =
Block log on disk full                (BLK_LOG_DSK_FUL) = NO
Percent of max active log space by transaction(MAX_LOG) = 0
Num. of active log files for 1 active UOW(NUM_LOG_SPAN) = 0

Group commit count                          (MINCOMMIT) = 1
Percent log file reclaimed before soft chckpt (SOFTMAX) = 100
Log retain for recovery enabled             (LOGRETAIN) = OFF
User exit for logging enabled                (USEREXIT) = OFF

HADR database role                                      = STANDARD
HADR local host name                  (HADR_LOCAL_HOST) =
HADR local service name                (HADR_LOCAL_SVC) =
HADR remote host name                (HADR_REMOTE_HOST) =
HADR remote service name              (HADR_REMOTE_SVC) =
HADR instance name of remote server  (HADR_REMOTE_INST) =
HADR timeout value                       (HADR_TIMEOUT) = 120
HADR log write synchronization mode     (HADR_SYNCMODE) = NEARSYNC

First log archive method                 (LOGARCHMETH1) = OFF
Options for logarchmeth1                  (LOGARCHOPT1) =
Second log archive method                (LOGARCHMETH2) = OFF
Options for logarchmeth2                  (LOGARCHOPT2) =
Failover log archive path                (FAILARCHPATH) =
Number of log archive retries on error   (NUMARCHRETRY) = 5
Log archive retry Delay (secs)         (ARCHRETRYDELAY) = 20
Vendor options                              (VENDOROPT) =

Auto restart enabled                      (AUTORESTART) = ON
Index re-creation time and redo index build  (INDEXREC) = SYSTEM (RESTART)
Log pages during index build            (LOGINDEXBUILD) = OFF
Default number of loadrec sessions    (DFT_LOADREC_SES) = 1
Number of database backups to retain   (NUM_DB_BACKUPS) = 12
Recovery history retention (days)     (REC_HIS_RETENTN) = 366

TSM management class                    (TSM_MGMTCLASS) =
TSM node name                            (TSM_NODENAME) =
TSM owner                                   (TSM_OWNER) =
TSM password                             (TSM_PASSWORD) =

Automatic maintenance                      (AUTO_MAINT) = OFF
  Automatic database backup            (AUTO_DB_BACKUP) = OFF
  Automatic table maintenance          (AUTO_TBL_MAINT) = OFF
    Automatic runstats                  (AUTO_RUNSTATS) = OFF
    Automatic statistics profiling    (AUTO_STATS_PROF) = OFF
      Automatic profile updates         (AUTO_PROF_UPD) = OFF
    Automatic reorganization               (AUTO_REORG) = OFF
```

The following example shows a portion of the output of the command when you specify the SHOW DETAIL option. The value in the **Delayed Value** column is the value that will be applied the next time you start the instance.

```
                    Database Configuration for Database mick
Description                                Parameter    Current Value  Delayed
                                                                        Value

Database configuration release level                    = 0x0a00
Database release level                                  = 0x0a00

Database territory                                      = en_US
Database code page                                      = 819
Database code set                                       = ISO8859-1
Database country/region code                            = 1
Database collating sequence                             = UNIQUE      UNIQUE
Alternate collating sequence              (ALT_COLLATE) =
Dynamic SQL Query management           (DYN_QUERY_MGMT) = DISABLE     DISABLE
Discovery support for this database       (DISCOVER_DB) = ENABLE      ENABLE
Default query optimization class         (DFT_QUERYOPT) = 5           5
Degree of parallelism                      (DFT_DEGREE) = 1           1
Continue upon arithmetic exceptions   (DFT_SQLMATHWARN) = NO          NO
Default refresh age                   (DFT_REFRESH_AGE) = 0           0
Default maintained table types for opt (DFT_MTTB_TYPES) = SYSTEM      SYSTEM
Number of frequent values retained     (NUM_FREQVALUES) = 10          10
Number of quantiles retained            (NUM_QUANTILES) = 20          20

Backup pending                                          = NO

Database is consistent                                  = YES
Rollforward pending                                     = NO
Restore pending                                         = NO

Multi-page file allocation enabled                      = YES

Log retain for recovery status                          = NO
User exit for logging status                            = NO

Data Links Token Expiry Interval (sec)      (DL_EXPINT) = 60          60
Data Links Write Token Init Expiry Intvl(DL_WT_IEXPINT) = 60          60
Data Links Number of Copies             (DL_NUM_COPIES) = 1           1
Data Links Time after Drop (days)        (DL_TIME_DROP) = 1           1
Data Links Token in Uppercase                (DL_UPPER) = NO          NO
Data Links Token Algorithm                   (DL_TOKEN) = MAC0        MAC0

Database heap (4KB)                            (DBHEAP) = 1200         1200
Size of database shared memory (4KB)  (DATABASE_MEMORY) = AUTOMATIC   AUTOMATIC
                                                          (11516)      (11516)
Catalog cache size (4KB)              (CATALOGCACHE_SZ) = 64           64
Log buffer size (4KB)                        (LOGBUFSZ) = 8            8
Utilities heap size (4KB)                (UTIL_HEAP_SZ) = 5000         5000
Buffer pool size (pages)                     (BUFFPAGE) = 1000         1000
Extended storage segments size (4KB)    (ESTORE_SEG_SZ) = 16000        16000
Number of extended storage segments   (NUM_ESTORE_SEGS) = 0            0
Max storage for lock list (4KB)              (LOCKLIST) = 128          128

Max size of appl. group mem set (4KB) (APPGROUP_MEM_SZ) = 30000        30000
Percent of mem for appl. group heap   (GROUPHEAP_RATIO) = 70           70
Max appl. control heap size (4KB)     (APP_CTL_HEAP_SZ) = 128          128
Sort heap thres for shared sorts (4KB) (SHEAPTHRES_SHR) = (SHEAPTHRES) (SHEAPTHRES)
Sort list heap (4KB)                         (SORTHEAP) = 256          256
SQL statement heap (4KB)                     (STMTHEAP) = 2048         2048
Default application heap (4KB)             (APPLHEAPSZ) = 128          128
```

```
Package cache size (4KB)                   (PCKCACHESZ) = (MAXAPPLS*8) (MAXAPPLS*8)
Statistics heap size (4KB)               (STAT_HEAP_SZ) = 4384         4384

Interval for checking deadlock (ms)         (DLCHKTIME) = 10000        10000
Percent. of lock lists per application       (MAXLOCKS) = 10           10
Lock timeout (sec)                        (LOCKTIMEOUT) = -1           -1

Changed pages threshold                (CHNGPGS_THRESH) = 60           60
Number of asynchronous page cleaners   (NUM_IOCLEANERS) = 1            1
Number of I/O servers                   (NUM_IOSERVERS) = 3            3
Index sort flag                             (INDEXSORT) = YES          YES
Sequential detect flag                      (SEQDETECT) = YES          YES
Default prefetch size (pages)         (DFT_PREFETCH_SZ) = AUTOMATIC    AUTOMATIC

Track modified pages                         (TRACKMOD) = NO           NO

Default number of containers                            = 1            1
Default tablespace extentsize (pages)   (DFT_EXTENT_SZ) = 32           32

Max number of active applications            (MAXAPPLS) = AUTOMATIC    AUTOMATIC
                                                           (40)         (40)
Average number of active applications       (AVG_APPLS) = 1            1
Max DB files open per application            (MAXFILOP) = 64           64

Log file size (4KB)                         (LOGFILSIZ) = 1000         1000
Number of primary log files                (LOGPRIMARY) = 3            3
Number of secondary log files               (LOGSECOND) = 2            2
Changed path to log files                  (NEWLOGPATH) =
Path to log files                                       = home/db2inst /home
                                                          /db2inst      /db2inst
                                                          /NODE0000     /db2inst
                                                          /SQL00001     /NODE0000
                                                          /SQLOGDIR/    /SQL00001
                                                                        /SQLOGDIR/
Overflow log path                     (OVERFLOWLOGPATH) =
Mirror log path                         (MIRRORLOGPATH) =
First active log file                                   =
Block log on disk full                (BLK_LOG_DSK_FUL) = NO           NO
Percent of max active log space by transaction(MAX_LOG) = 0            0
Num. of active log files for 1 active UOW(NUM_LOG_SPAN) = 0            0
Group commit count                          (MINCOMMIT) = 1            1
Percent log file reclaimed before soft chckpt (SOFTMAX) = 100          100
Log retain for recovery enabled             (LOGRETAIN) = OFF          OFF
User exit for logging enabled                (USEREXIT) = OFF          OFF

HADR database role                                      = STANDARD     STANDARD
HADR local host name                  (HADR_LOCAL_HOST) =
HADR local service name                (HADR_LOCAL_SVC) =
HADR remote host name                (HADR_REMOTE_HOST) =
HADR remote service name              (HADR_REMOTE_SVC) =
HADR instance name of remote server  (HADR_REMOTE_INST) =
HADR timeout value                       (HADR_TIMEOUT) = 120          120
HADR log write synchronization mode     (HADR_SYNCMODE) = NEARSYNC     NEARSYNC

First log archive method                 (LOGARCHMETH1) = OFF          OFF
Options for logarchmeth1                  (LOGARCHOPT1) =
Second log archive method                (LOGARCHMETH2) = OFF          OFF
Options for logarchmeth2                  (LOGARCHOPT2) =
Failover log archive path                (FAILARCHPATH) =
Number of log archive retries on error   (NUMARCHRETRY) = 5            5
Log archive retry Delay (secs)         (ARCHRETRYDELAY) = 20           20
Vendor options                              (VENDOROPT) =
Auto restart enabled                      (AUTORESTART) = ON           ON
Index re-creation time and redo index build  (INDEXREC) = SYSTEM       SYSTEM
                                                          (RESTART)    (RESTART)
Log pages during index build            (LOGINDEXBUILD) = OFF          OFF
Default number of loadrec sessions    (DFT_LOADREC_SES) = 1            1
```

```
Number of database backups to retain   (NUM_DB_BACKUPS) = 12          12
Recovery history retention (days)     (REC_HIS_RETENTN) = 366         366

TSM management class                    (TSM_MGMTCLASS) =
TSM node name                            (TSM_NODENAME) =
TSM owner                                   (TSM_OWNER) =
TSM password                             (TSM_PASSWORD) =

Automatic maintenance                      (AUTO_MAINT) = OFF         OFF
  Automatic database backup            (AUTO_DB_BACKUP) = OFF         OFF
  Automatic table maintenance          (AUTO_TBL_MAINT) = OFF         OFF
    Automatic runstats                  (AUTO_RUNSTATS) = OFF         OFF
    Automatic statistics profiling    (AUTO_STATS_PROF) = OFF         OFF
    Automatic profile updates           (AUTO_PROF_UPD) = OFF         OFF
    Automatic reorganization               (AUTO_REORG) = OFF          OFF
```

**Usage notes:**

If an error occurs, the information returned is not valid. If the configuration file is invalid, an error message is returned. The database must be restored from a backup version.

To set the database configuration parameters to the database manager defaults, use the RESET DATABASE CONFIGURATION command.

**Related tasks:**

- "Changing node and database configuration files" in the *Administration Guide: Implementation*
- "Configuring DB2 with configuration parameters" in the *Administration Guide: Performance*

**Related reference:**

- "RESET DATABASE CONFIGURATION" on page 639
- "UPDATE DATABASE CONFIGURATION" on page 730
- "Configuration parameters summary" in the *Administration Guide: Performance*

# GET DATABASE MANAGER CONFIGURATION

Returns the values of individual entries in the database manager configuration file.

**Authorization:**

None

**Required connection:**

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To display the database manager configuration for a remote instance, it is necessary to first attach to that instance. The SHOW DETAIL clause requires an instance attachment.

**Command syntax:**

```
►►──GET───┬─DATABASE MANAGER─┬───┬─CONFIGURATION─┬──┬─────────────┬──────────►◄
          ├─DB MANAGER───────┤   ├─CONFIG────────┤  └─SHOW DETAIL─┘
          └─DBM──────────────┘   └─CFG───────────┘
```

**Command parameters:**

**SHOW DETAIL**
: Displays detailed information showing the current value of database manager configuration parameters as well as the value of the parameters the next time you start the database manager. This option lets you see the result of dynamic changes to configuration parameters.

**Examples:**

**Note:** Both node type and platform determine which configuration parameters are listed.

The following is sample output from GET DATABASE MANAGER CONFIGURATION (issued on AIX):

```
          Database Manager Configuration

     Node type = Database Server with local clients

 Database manager configuration release level            = 0x0a00

 CPU speed (millisec/instruction)             (CPUSPEED) = 4.000000e-05

 Max number of concurrently active databases     (NUMDB) = 8
 Data Links support                          (DATALINKS) = NO
 Federated Database System Support           (FEDERATED) = NO
 Transaction processor monitor name        (TP_MON_NAME) =

 Default charge-back account           (DFT_ACCOUNT_STR) =

 Java Development Kit installation path       (JDK_PATH) = /usr/java131

 Diagnostic error capture level              (DIAGLEVEL) = 3
 Notify Level                              (NOTIFYLEVEL) = 3
 Diagnostic data directory path               (DIAGPATH) =

 Default database monitor switches
```

```
  Buffer pool                         (DFT_MON_BUFPOOL) = OFF
  Lock                                   (DFT_MON_LOCK) = OFF
  Sort                                   (DFT_MON_SORT) = OFF
  Statement                              (DFT_MON_STMT) = OFF
  Table                                 (DFT_MON_TABLE) = OFF
  Timestamp                         (DFT_MON_TIMESTAMP) = ON
  Unit of work                            (DFT_MON_UOW) = OFF
Monitor health of instance and databases   (HEALTH_MON) = ON

SYSADM group name                        (SYSADM_GROUP) =
SYSCTRL group name                      (SYSCTRL_GROUP) =
SYSMAINT group name                    (SYSMAINT_GROUP) =
SYSMON group name                        (SYSMON_GROUP) =

Client Userid-Password Plugin          (CLNT_PW_PLUGIN) =
Client Kerberos Plugin                (CLNT_KRB_PLUGIN) =
Group Plugin                             (GROUP_PLUGIN) =
GSS Plugin for Local Authorization    (LOCAL_GSSPLUGIN) =
Server Plugin Mode                    (SRV_PLUGIN_MODE) = UNFENCED
Server List of GSS Plugins      (SRVCON_GSSPLUGIN_LIST) =
Server Userid-Password Plugin        (SRVCON_PW_PLUGIN) =
Server Connection Authentication          (SRVCON_AUTH) = NOT_SPECIFIED
Database manager authentication        (AUTHENTICATION) = SERVER
Cataloging allowed without authority   (CATALOG_NOAUTH) = YES
Trust all clients                      (TRUST_ALLCLNTS) = YES
Trusted client authentication          (TRUST_CLNTAUTH) = CLIENT
Bypass federated authentication            (FED_NOAUTH) = NO

Default database path                       (DFTDBPATH) = /home/db2inst

Database monitor heap size (4KB)          (MON_HEAP_SZ) = 90
Java Virtual Machine heap size (4KB)     (JAVA_HEAP_SZ) = 512
Audit buffer size (4KB)                  (AUDIT_BUF_SZ) = 0
Size of instance shared memory (4KB)  (INSTANCE_MEMORY) = AUTOMATIC
Backup buffer default size (4KB)            (BACKBUFSZ) = 1024
Restore buffer default size (4KB)           (RESTBUFSZ) = 1024

Sort heap threshold (4KB)                  (SHEAPTHRES) = 20000

Directory cache support                     (DIR_CACHE) = YES

Application support layer heap size (4KB)   (ASLHEAPSZ) = 15
Max requester I/O block size (bytes)         (RQRIOBLK) = 32767
Query heap size (4KB)                   (QUERY_HEAP_SZ) = 1000

Workload impact by throttled utilities(UTIL_IMPACT_LIM) = 10

Priority of agents                           (AGENTPRI) = SYSTEM
Max number of existing agents               (MAXAGENTS) = 200
Agent pool size                        (NUM_POOLAGENTS) = 100(calculated)
Initial number of agents in pool       (NUM_INITAGENTS) = 0
Max number of coordinating agents     (MAX_COORDAGENTS) = MAXAGENTS
Max no. of concurrent coordinating agents  (MAXCAGENTS) = MAX_COORDAGENTS
Max number of client connections      (MAX_CONNECTIONS) = MAX_COORDAGENTS

Keep fenced process                        (KEEPFENCED) = YES
Number of pooled fenced processes         (FENCED_POOL) = MAX_COORDAGENTS
Initial number of fenced processes     (NUM_INITFENCED) = 0

Index re-creation time and redo index build  (INDEXREC) = RESTART

Transaction manager database name         (TM_DATABASE) = 1ST_CONN
Transaction resync interval (sec)     (RESYNC_INTERVAL) = 180

SPM name                                     (SPM_NAME) =
SPM log size                          (SPM_LOG_FILE_SZ) = 256
SPM resync agent limit                 (SPM_MAX_RESYNC) = 20
SPM log path                             (SPM_LOG_PATH) =

TCP/IP Service name                          (SVCENAME) =
Discovery mode                               (DISCOVER) = SEARCH
Discover server instance                (DISCOVER_INST) = ENABLE

Maximum query degree of parallelism   (MAX_QUERYDEGREE) = ANY
Enable intra-partition parallelism     (INTRA_PARALLEL) = NO

No. of int. communication buffers(4KB)(FCM_NUM_BUFFERS) = 512
Number of FCM request blocks              (FCM_NUM_RQB) = AUTOMATIC
Number of FCM connection entries      (FCM_NUM_CONNECT) = AUTOMATIC
Number of FCM message anchors         (FCM_NUM_ANCHORS) = AUTOMATIC
```

The following output sample shows the information displayed when you specify the WITH DETAIL option. The value that appears in the **Delayed Value** is the value that will be in effect the next time you start the database manager instance.

```
                         Database Manager Configuration
 Node type = Database Server with local clients
 Description                                  Parameter   Current Value  Delayed
                                                                          Value

Database manager configuration release level            = 0x0a00

 CPU speed (millisec/instruction)             (CPUSPEED) = 4.000000e     4.000000e
                                                            -05           -05
 Max number of concurrently active databases     (NUMDB) = 8             8
 Data Links support                          (DATALINKS) = NO            NO
 Federated Database System Support           (FEDERATED) = NO            NO
 Transaction processor monitor name        (TP_MON_NAME) =

 Default charge-back account           (DFT_ACCOUNT_STR) =

 Java Development Kit installation path       (JDK_PATH) = /wsdb/v81     /usr
                                                            /bldsupp      /java131
                                                            /AIX/jdk1.3.1

 Diagnostic error capture level              (DIAGLEVEL) = 3            3
 Notify Level                              (NOTIFYLEVEL) = 3            3
 Diagnostic data directory path               (DIAGPATH) =

 Default database monitor switches
   Buffer pool                         (DFT_MON_BUFPOOL) = OFF         OFF
   Lock                                   (DFT_MON_LOCK) = OFF         OFF
   Sort                                   (DFT_MON_SORT) = OFF         OFF
   Statement                              (DFT_MON_STMT) = OFF         OFF
   Table                                 (DFT_MON_TABLE) = OFF         OFF
   Timestamp                         (DFT_MON_TIMESTAMP) = ON          ON
   Unit of work                            (DFT_MON_UOW) = OFF         OFF
 Monitor health of instance and databases   (HEALTH_MON) = ON          ON

 SYSADM group name                        (SYSADM_GROUP) = BUILD
 SYSCTRL group name                      (SYSCTRL_GROUP) =
 SYSMAINT group name                    (SYSMAINT_GROUP) =
 SYSMON group name                        (SYSMON_GROUP) =

 Client Userid-Password Plugin          (CLNT_PW_PLUGIN) =

 Client Kerberos Plugin                (CLNT_KRB_PLUGIN) =

 Group Plugin                             (GROUP_PLUGIN) =

 GSS Plugin for Local Authorization    (LOCAL_GSSPLUGIN) =

 Server Plugin Mode                    (SRV_PLUGIN_MODE) = UNFENCED    UNFENCED
 Server List of GSS Plugins      (SRVCON_GSSPLUGIN_LIST) =

 Server Userid-Password Plugin        (SRVCON_PW_PLUGIN) =

 Server Connection Authentication          (SRVCON_AUTH) = NOT_          NOT_
                                                           SPECIFIED    SPECIFIED
 Database manager authentication        (AUTHENTICATION) = SERVER      SERVER
 Cataloging allowed without authority   (CATALOG_NOAUTH) = YES         YES
 Trust all clients                      (TRUST_ALLCLNTS) = YES         YES
 Trusted client authentication          (TRUST_CLNTAUTH) = CLIENT      CLIENT
 Bypass federated authentication            (FED_NOAUTH) = NO          NO

 Default database path                       (DFTDBPATH) = /home       /home
                                                            /db2inst    /db2inst

 Database monitor heap size (4KB)          (MON_HEAP_SZ) = 90          90
 Java Virtual Machine heap size (4KB)     (JAVA_HEAP_SZ) = 512         512
 Audit buffer size (4KB)                  (AUDIT_BUF_SZ) = 0           0
 Size of instance shared memory (4KB)  (INSTANCE_MEMORY) = AUTOMATIC   AUTOMATIC
                                                           (5386)       (20)
 Backup buffer default size (4KB)            (BACKBUFSZ) = 1024        1024
 Restore buffer default size (4KB)           (RESTBUFSZ) = 1024        1024

 Sort heap threshold (4KB)                  (SHEAPTHRES) = 20000       20000

 Directory cache support                     (DIR_CACHE) = YES         YES

 Application support layer heap size (4KB)   (ASLHEAPSZ) = 15          15
 Max requester I/O block size (bytes)         (RQRIOBLK) = 32767       32767
 Query heap size (4KB)                   (QUERY_HEAP_SZ) = 1000        1000

 Workload impact by throttled utilities(UTIL_IMPACT_LIM) = 10          10

 Priority of agents                           (AGENTPRI) = SYSTEM      SYSTEM
 Max number of existing agents               (MAXAGENTS) = 200         200
 Agent pool size                        (NUM_POOLAGENTS) = 100         100
                                                                       (calculated)
 Initial number of agents in pool       (NUM_INITAGENTS) = 0           0
 Max number of coordinating agents     (MAX_COORDAGENTS) = 200         MAXAGENTS
 Max no. of concurrent coordinating agents  (MAXCAGENTS) = 200         MAX_COORDAGENTS
 Max number of client connections      (MAX_CONNECTIONS) = 200         MAX_COORDAGENTS

 Keep fenced process                        (KEEPFENCED) = YES         YES
 Number of pooled fenced processes         (FENCED_POOL) = MAX_        MAX_
                                                           COORDAGENTS  COORDAGENTS
 Initial number of fenced processes     (NUM_INITFENCED) = 0           0

 Index re-creation time and redo index build  (INDEXREC) = RESTART     RESTART

 Transaction manager database name         (TM_DATABASE) = 1ST_CONN    1ST_CONN
 Transaction resync interval (sec)     (RESYNC_INTERVAL) = 180         180

 SPM name                                     (SPM_NAME) =
 SPM log size                          (SPM_LOG_FILE_SZ) = 256         256
 SPM resync agent limit                 (SPM_MAX_RESYNC) = 20          20
 SPM log path                             (SPM_LOG_PATH) =

 TCP/IP Service name                          (SVCENAME) =
 Discovery mode                               (DISCOVER) = SEARCH      SEARCH
 Discover server instance                (DISCOVER_INST) = ENABLE      ENABLE

 Maximum query degree of parallelism   (MAX_QUERYDEGREE) = ANY         ANY
 Enable intra-partition parallelism     (INTRA_PARALLEL) = NO          NO

 No. of int. communication buffers(4KB)(FCM_NUM_BUFFERS) = 0           512
 Number of FCM request blocks              (FCM_NUM_RQB) = AUTOMATIC   AUTOMATIC
                                                            (0)         (256)
```

```
Number of FCM connection entries      (FCM_NUM_CONNECT) = AUTOMATIC   AUTOMATIC
                                                          (-1)         (-1)
Number of FCM message anchors         (FCM_NUM_ANCHORS) = AUTOMATIC   AUTOMATIC
                                                           (-1)          (-1)
```

**Usage notes:**

If an attachment to a remote instance or a different local instance exists, the database manager configuration parameters for the attached server are returned; otherwise, the local database manager configuration parameters are returned.

If an error occurs, the information returned is invalid. If the configuration file is invalid, an error message is returned. The user must install the database manager again to recover.

To set the configuration parameters to the default values shipped with the database manager, use the RESET DATABASE MANAGER CONFIGURATION command.

**Related tasks:**

- "Changing node and database configuration files" in the *Administration Guide: Implementation*
- "Configuring DB2 with configuration parameters" in the *Administration Guide: Performance*

**Related reference:**

- "RESET DATABASE MANAGER CONFIGURATION" on page 641
- "UPDATE DATABASE MANAGER CONFIGURATION" on page 733
- "Configuration parameters summary" in the *Administration Guide: Performance*

# GET DATABASE MANAGER MONITOR SWITCHES

Displays the status of the database system monitor switches. Monitor switches instruct the database system manager to collect database activity information. Each application using the database system monitor interface has its own set of monitor switches. A database manager-level switch is on when any of the monitoring applications has turned it on. This command is used to determine if the database system monitor is currently collecting data for any monitoring application.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance or database:

- If there is neither an attachment to an instance, nor a connection to a database, a default instance attachment is created.
- If there is both an attachment to an instance, and a database connection, the instance attachment is used.

To display the settings for a remote instance, or for a different local instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──GET──┬─DATABASE MANAGER─┬──MONITOR SWITCHES──────────────────────────────►
         ├─DB MANAGER───────┤
         └─DBM──────────────┘

►──┬────────────────────────────────────────┬─────────────────────────────►◄
   ├─AT DBPARTITIONNUM──db-partition-number─┤
   └─GLOBAL─────────────────────────────────┘
```

**Command parameters:**

**AT DBPARTITIONNUM db-partition-number**
: Specifies the database partition for which the status of the database manager monitor switches is to be displayed.

**GLOBAL**
: Returns an aggregate result for all database partitions in a partitioned database system.

**Examples:**

The following is sample output from GET DATABASE MANAGER MONITOR SWITCHES:

```
            DBM System Monitor Information Collected

Switch list for db partition number 1
Buffer Pool Activity Information (BUFFERPOOL) = ON   06-11-2003 10:11:01.738377
Lock Information                       (LOCK) = OFF
Sorting Information                    (SORT) = ON   06-11-2003 10:11:01.738400
SQL Statement Information         (STATEMENT) = OFF
Table Activity Information            (TABLE) = OFF
Take Timestamp Information        (TIMESTAMP) = ON   06-11-2003 10:11:01.738525
Unit of Work Information                (UOW) = ON   06-11-2003 10:11:01.738353
```

**Usage notes:**

The recording switches BUFFERPOOL, LOCK, SORT, STATEMENT, TABLE, and UOW are off by default, but can be switched on using the UPDATE MONITOR SWITCHES command. If any of these switches are on, this command also displays the time stamp for when the switch was turned on.

The recording switch TIMESTAMP is on by default, but can be switched off using UPDATE MONITOR SWITCHES. When this switch is on the system issues timestamp calls when collecting information for timestamp monitor elements. Examples of these elements are:

- agent_sys_cpu_time
- agent_usr_cpu_time
- appl_con_time
- con_elapsed_time
- con_response_time
- conn_complete_time
- db_conn_time
- elapsed_exec_time
- gw_comm_error_time
- gw_con_time
- gw_exec_time
- host_response_time
- last_backup
- last_reset
- lock_wait_start_time
- network_time_bottom
- network_time_top
- prev_uow_stop_time
- rf_timestamp
- ss_sys_cpu_time
- ss_usr_cpu_time
- status_change_time
- stmt_elapsed_time
- stmt_start
- stmt_stop
- stmt_sys_cpu_time
- stmt_usr_cpu_time
- uow_elapsed_time

- uow_start_time
- uow_stop_time

If the TIMESTAMP switch is off, timestamp operating system calls are not issued to determine these elements and these elements will contain zero. Note that turning this switch off becomes important as CPU utilization approaches 100%; when this occurs, the CPU time required for issuing timestamps increases dramatically.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "GET SNAPSHOT" on page 419
- "GET MONITOR SWITCHES" on page 410
- "RESET MONITOR" on page 643
- "UPDATE MONITOR SWITCHES" on page 740

## GET DESCRIPTION FOR HEALTH INDICATOR

Returns a description for the specified health indicator. A Health Indicator measures the healthiness of a particular state, capacity, or behavior of the database system. The state defines whether or not the database object or resource is operating normally.

**Authorization:**

None.

**Required connection:**

Instance. If there is no instance attachment, a default instance attachment is created.

To obtain a snapshot of a remote instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──GET DESCRIPTION FOR HEALTH INDICATOR──shortname──────────────────────────────►◄
```

**Command parameters:**

**HEALTH INDICATOR** *shortname*

The name of the health indicator for which you would like to retrieve the description. Health indicator names consist of a two- or three-letter object identifier followed by a name which describes what the indicator measures. For example:

```
db.sort_privmem_util
```

**Examples:**

The following is sample output from the GET DESCRIPTION FOR HEALTH INDICATOR command.

```
GET DESCRIPTION FOR HEALTH INDICATOR db2.sort_privmem_util

DESCRIPTION FOR db2.sort_privmem_util

Sorting is considered healthy if there is sufficient heap space in which to
perform sorting and sorts do not overflow unnecessarily. This indicator
tracks the utilization of the private sort memory. If db2.sort_heap_allocated
(system monitor data element) >= SHEAPTHRES (DBM configuration parameter), sorts
may not be getting full sort heap as defined by the SORTHEAP parameter and an
alert may be generated. The indicator is calculated using the formula:
(db2.sort_heap_allocated / SHEAPTHRES) * 100. The Post Threshold Sorts snapshot
monitor element measures the number of sorts that have requested heaps after the
sort heap threshold has been exceeded. The value of this indicator, shown in the
Additional Details, indicates the degree of severity of the problem for this
health indicator. The Maximum Private Sort Memory Used snapshot monitor element
maintains a private sort memory high-water mark for the instance.  The value of
this indicator, shown in the Additional Information, indicates the maximum amount
of private sort memory that has been in use at any one point in time since the
instance was last recycled.  This value can be used to help determine an
appropriate value for SHEAPTHRES.
```

**Related reference:**

- “Health indicators” in the *System Monitor Guide and Reference*

# GET HEALTH NOTIFICATION CONTACT LIST

Returns the list of contacts and contact groups that are notified about the health of an instance. A contact list consists of e-mail addresses or pager Internet addresses of individuals who are to be notified when non-normal health conditions are present for an instance or any of its database objects.

**Authorization:**

None.

**Required Connection:**

Instance. An explicit attachment is not required.

**Command Syntax:**

```
►►──GET──┬─HEALTH NOTIFICATION CONTACT─┬──LIST──────────────────────────────►◄
         └─NOTIFICATION────────────────┘
```

**Command Parameters:**
None.
**Examples:**
Issuing the command GET NOTIFICATION LIST results in a report similar to the following:

```
 Name                           Type
------------------------------ -------------
Joe Brown                      Contact
Support                        Contact group
```

# GET HEALTH SNAPSHOT

Retrieves the health status information for the database manager and its databases. The information returned represents a snapshot of the health state at the time the command was issued.

**Scope:**

In a partitioned database environment, this command can be invoked from any database partition defined in the `db2nodes.cfg` file. By default it acts on the partition from which it was invoked. If you use the GLOBAL option, it will extract consolidated information from all of the partitions.

**Authorization:**

None.

**Required connection:**

Instance. If there is no instance attachment, a default instance attachment is created.

To obtain a snapshot of a remote instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──GET HEALTH SNAPSHOT FOR──┬──┬─DATABASE MANAGER─┬─────────────────┬──────────►
                             │  ├─DB MANAGER───────┤                 │
                             │  └─DBM──────────────┘                 │
                             ├─ALL DATABASES─────────────────────────┤
                             └──┬─ALL──────────┬──ON──database alias─┘
                                ├──┬─DATABASE─┬┤
                                │  └─DB───────┘│
                                └─TABLESPACES──┘

►──┬─────────────────────────────────────────┬──┬─────────────┬───────────────►
   ├─AT DBPARTITIONNUM──db partition number──┤  └─SHOW DETAIL─┘
   └─GLOBAL──────────────────────────────────┘

►──┬──────────────────────┬───────────────────────────────────────────────────►◄
   └─WITH FULL COLLECTION─┘
```

**Command parameters:**

**DATABASE MANAGER**
Provides statistics for the active database manager instance.

**ALL DATABASES**
Provides health states for all active databases on the current database partition.

**ALL ON database-alias**
Provides health states and information about all table spaces and buffer pools for a specified database.

**DATABASE ON database-alias**

**TABLESPACES ON database-alias**

Provides information about table spaces for a specified database.

**AT DBPARTITIONNUM db-partition-number**

Returns results for the database partition specified.

**GLOBAL**

Returns an aggregate result for all database partitions in a partitioned database system.

**SHOW DETAIL**

Specifies that the output should include the historical data for each health monitor data element in the form of {(Timestamp, Value, Formula)}, where the bracketed parameters (Timestamp, Value, Formula), will be repeated for each history record that is returned. For example,

```
(03-19-2002 13:40:24.138865,50,((1-(4/8))*100)),
(03-19-2002 13:40:13.1386300,50,((1-(4/8))*100)),
(03-19-2002 13:40:03.1988858,0,((1-(3/3))*100))
```

Collection object history is returned for all collection objects in ATTENTION or AUTOMATE FAILED state.

The SHOW DETAIL option also provides additional contextual information that can be useful to understanding the value and alert state of the associated Health Indicator. For example, if the table space storage utilization Health Indicator is being used to determine how full the table space is, the rate at which the table space is growing will also be provided by SHOW DETAIL.

**WITH FULL COLLECTION**

Specifies that full collection information for all collection state-based health indicators is to be returned. The output returned when this option is specified is for collection objects in NORMAL, AUTOMATED, ATTENTION, or AUTOMATE FAILED state. This option can be specified in conjunction with the SHOW DETAIL option.

**Examples:**

The following is typical output resulting from a request for database manager information:

```
D:\>DB2 GET HEALTH SNAPSHOT FOR DBM

             Database Manager Health Snapshot

Node name                                      =
Node type                                      = Enterprise Server Edition
                                                 with local and remote clients
Instance name                                  = DB2
Snapshot timestamp                             = 02/17/2004 12:39:44.818949

Number of database partitions in DB2 instance  = 1
Start Database Manager timestamp               = 02/17/2004 12:17:21.000119
Instance highest severity alert state          = Normal

Health Indicators:

    Indicator Name                             = db2.db2_op_status
       Value                                   = 0
       Evaluation timestamp                    = 02/17/2004 12:37:23.393000
       Alert state                             = Normal

```

```
Indicator Name                            = db2.sort_privmem_util
   Value                                  = 0
   Unit                                   = %
   Evaluation timestamp                   = 02/17/2004 12:37:23.393000
   Alert state                            = Normal

Indicator Name                            = db2.mon_heap_util
   Value                                  = 6
   Unit                                   = %
   Evaluation timestamp                   = 02/17/2004 12:37:23.393000
   Alert state                            = Normal
```

# GET INSTANCE

Returns the value of the **DB2INSTANCE** environment variable.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──GET INSTANCE─────────────────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Examples:**
The following is sample output from GET INSTANCE:

```
 The current database manager instance is:  smith
```

# GET MONITOR SWITCHES

Displays the status of the database system monitor switches for the current session. Monitor switches instruct the database system manager to collect database activity information. Each application using the database system monitor interface has its own set of monitor switches. This command displays them. To display the database manager-level switches, use the GET DBM MONITOR SWITCHES command.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance. If there is no instance attachment, a default instance attachment is created.

To display the settings for a remote instance, or for a different local instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──GET MONITOR SWITCHES──┬─────────────────────────────────────────┬──────────────►◄
                          ├─AT DBPARTITIONNUM──db-partition-number──┤
                          └─GLOBAL──────────────────────────────────┘
```

**Command parameters:**

**AT DBPARTITIONNUM db-partition-number**
: Specifies the database partition for which the status of the monitor switches is to be displayed.

**GLOBAL**
: Returns an aggregate result for all database partitions in a partitioned database system.

**Examples:**

The following is sample output from GET MONITOR SWITCHES:

```
            Monitor Recording Switches

Switch list for db partition number 1
Buffer Pool Activity Information  (BUFFERPOOL) = ON  02-20-2003 16:04:30.070073
Lock Information                        (LOCK) = OFF
Sorting Information                     (SORT) = OFF
SQL Statement Information          (STATEMENT) = ON  02-20-2003 16:04:30.070073
Table Activity Information             (TABLE) = OFF
Take Timestamp Information         (TIMESTAMP) = ON  02-20-2003 16:04:30.070073
Unit of Work Information                 (UOW) = ON  02-20-2003 16:04:30.070073
```

**Usage notes:**

The recording switch TIMESTAMP is on by default, but can be switched off using UPDATE MONITOR SWITCHES. When this switch is on the system issues timestamp calls when collecting information for timestamp monitor elements.

The recording switch TIMESTAMP is on by default, but can be switched off using UPDATE MONITOR SWITCHES. If this switch is off, this command also displays the time stamp for when the switch was turned off. When this switch is on the system issues timestamp calls when collecting information for timestamp monitor elements. Examples of these elements are:

- agent_sys_cpu_time
- agent_usr_cpu_time
- appl_con_time
- con_elapsed_time
- con_response_time
- conn_complete_time
- db_conn_time
- elapsed_exec_time
- gw_comm_error_time
- gw_con_time
- gw_exec_time
- host_response_time
- last_backup
- last_reset
- lock_wait_start_time
- network_time_bottom
- network_time_top
- prev_uow_stop_time
- rf_timestamp
- ss_sys_cpu_time
- ss_usr_cpu_time
- status_change_time
- stmt_elapsed_time
- stmt_start
- stmt_stop
- stmt_sys_cpu_time
- stmt_usr_cpu_time
- uow_elapsed_time
- uow_start_time
- uow_stop_time

If the TIMESTAMP switch is off, timestamp operating system calls are not issued to determine these elements and these elements will contain zero. Note that turning this switch off becomes important as CPU utilization approaches 100%; when this occurs, the CPU time required for issuing timestamps increases dramatically.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "GET SNAPSHOT" on page 419
- "GET DATABASE MANAGER MONITOR SWITCHES" on page 400
- "RESET MONITOR" on page 643
- "UPDATE MONITOR SWITCHES" on page 740

# GET RECOMMENDATIONS

Returns descriptions of recommendations for improving the health of the aspect of the database system that is monitored by the specified health indicator. Recommendations can be returned for a health indicator that is in an alert state on a specific object, or the full set of recommendations for a given health indicator can be queried.

**Scope:**

In a partitioned database environment, this command can be invoked from any database partition defined in the `db2nodes.cfg` file. It acts only on that partition unless the GLOBAL parameter is specified.

**Authorization:**

None.

**Required connection:**

Instance. If there is no instance attachment, a default instance attachment is created. To retrieve recommendations for a remote instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──GET RECOMMENDATIONS FOR HEALTH INDICATOR──health-indicator-name─────────────────────►

►─┬──────────────────────────────────────────────────────────────────────────────┬────►
  └─FOR───┬─DBM──────────────────────────────────────────────────────────────────┘
          └─┬─TABLESPACE──tblspacename───────────────────────────┬─ON──database-alias─┘
            ├─CONTAINER──containername──FOR TABLESPACE──tblspacename─┤
            └─DATABASE───────────────────────────────────────────┘

►─┬─────────────────────────────────────────┬──────────────────────────────────────►◄
  ├─AT DBPARTITIONNUM──db-partition-number──┤
  └─GLOBAL──────────────────────────────────┘
```

**Command parameters:**

**HEALTH INDICATOR** *health-indicator-name*
: The name of the health indicator for which you would like to retrieve the recommendations. Health indicator names consist of a two- or three-letter object identifier followed by a name that describes what the indicator measures.

**DBM** Returns recommendations for a database manager health indicator that has entered an alert state.

**TABLESPACE**
: Returns recommendation for a health indicator that has entered an alert state on the specified table space and database.

**CONTAINER**
: Returns recommendation for a health indicator that has entered an alert state on the specified container in the specified table space and database.

**DATABASE**
: Returns recommendations for a health indicator that has entered an alert state on the specified database.

**ON** *database-alias*

Specifies a database.

**AT DBPARTITIONNUM**

Specifies the partition number at which the health indicator has entered an alert state. If a partition number is not specified and GLOBAL is not specified, the command will return information for the currently connected partition.

**GLOBAL**

Retrieves recommendations for the specified health indicator across all partitions. In cases where the recommendations are the same on different partitions, those recommendations are returned as a single set of recommendations that solve the health indicator on the affected partitions.

**Examples:**

```
   db2 get recommendations for health indicator db.db_heap_util
    for database on sample

Problem:

    Indicator Name                              = db.db_heap_util
       Value                                    = 42
       Evaluation timestamp                     = 11/25/2003 19:04:54
       Alert state                              = Alarm
       Additional information                   =

Recommendations:

Recommendation: Increase the database heap size.
Rank: 1

Increase the database configuration parameter dbheap sufficiently
to move utilization to normal operating levels. To increase the
value, set the new value of dbheap to be equal to
(pool_cur_size / (4096*U)) where U is the desired utilization rate.
For example, if your desired utilization rate is 60% of the warning
threshold level, which you have set at 75%, then
U = 0.6 * 0.75 = 0.45 (or 45%).

Take one of the following actions:

Execute the following scripts at the DB2 server (this can be done using
the EXEC_DB2_CMD stored procedure):

CONNECT TO DATABASE SAMPLE;
UPDATE DB CFG USING DBHEAP 149333;
CONNECT_RESET;


Launch DB2 tool: Database Configuration Window

The Database Configuration window can be used to view and update
database configuration parameters.

To open the Database Configuration window:
1.  From the Control Center, expand the object tree until you find
    the databases folder.
2.  Click the databases folder. Any existing database are displayed
    in the contents pane on the right side of the window.
3.  Right-click the database that you want in the contents pane, and
    click Configure Parameters in the pop-up menu. The Database
    Configuration window opens.

On the Performance tab, update the database heap size parameter as
```

```
suggested and click OK to apply the update.


Recommendation: Investigate memory usage of database heap.
Rank: 2

There is one database heap per database and the database manager uses
it on behalf of all applications connected to the database. The data
area is expanded as needed up to the maximum specified by dbheap.

For more information on the database heap, refer to the DB2
Information Center.

Investigate the amount of memory that was used for the database heap
over time to determine the most appropriate value for the database
heap configuration parameter. The database system monitor tracks the
highest amount of memory that was used for the database heap.


Take one of the following actions:

Launch DB2 tool: Memory Visualizer

The Memory Visualizer is used to monitor memory allocation within
a DB2 instance. It can be used to monitor overall memory usage, and
to update configuration parameters for individual memory components.

To open the Memory Visualizer:
1.  From the Control Center, expand the object tree until you find
    the instances folder.
2.  Click the instances folder. Any existing instances are displayed
    in the contents pane on the right side of the window.
3.  Right-click the instance that you want in the contents pane, and
    click View Memory Usage in the pop-up menu. The Memory Visualizer
    opens.

To start the Memory Visualizer from the command line issue the
db2memvis command.

The Memory Visualizer displays a hierarchical list of memory pools
for the database manager. Database Heap is listed under the Database
Manager Memory group for each database. On Windows, it is listed
under the Database Manager Shared Memory group.

Click the check box on the Show Plot column for the Database Heap
row to add the element to the plot.
```

**Usage notes:**

The GET RECOMMENDATIONS command can be used in two different ways:

- Specify only the health indicator to get an informational list of all possible recommendations. If no object is specified, the command will return a full listing of all recommendations that can be used to resolve an alert on the given health indicator.
- Specify an object to resolve a specific alert on that object. If an object (for example, a database or a table space) is specified, the recommendations returned will be specific to an alert on the object identified. In this case, the recommendations will be more specific and will contain more information about resolving the alert. If the health indicator identified is not in an alert state on the specified object, no recommendations will be returned.

**Related reference:**

- “Health indicators” in the *System Monitor Guide and Reference*

- “Health indicators summary” in the *System Monitor Guide and Reference*

# GET ROUTINE

Retrieves a routine SQL Archive (SAR) file for a specified SQL routine.

**Authorization:**

*dbadm*

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
►►──GET ROUTINE──INTO──file_name──FROM──┬──────────┬──PROCEDURE──routine_name──────►
                                        └─SPECIFIC─┘

►──┬───────────┬─────────────────────────────────────────────────────────────────►◄
   └─HIDE BODY─┘
```

**Command parameters:**

**INTO file_name**
Names the file where routine SQL archive (SAR) is stored.

**FROM**
Indicates the start of the specification of the routine to be retrieved.

**SPECIFIC**
The specified routine name is given as a specific name.

**PROCEDURE**
The routine is an SQL procedure.

**routine_name**
The name of the procedure. If SPECIFIC is specified then it is the specific name of the procedure. If the name is not qualified with a schema name, the CURRENT SCHEMA is used as the schema name of the routine. The *routine-name* must be an existing procedure that is defined as an SQL procedure.

**HIDE BODY**
Specifies that the body of the routine must be replaced by an empty body when the routine text is extracted from the catalogs.

This does not affect the compiled code; it only affects the text.

**Examples:**

```
GET ROUTINE INTO procs/proc1.sar FROM PROCEDURE myappl.proc1;
```

**Usage Notes:**

If a GET ROUTINE or a PUT ROUTINE operation (or their corresponding procedure) fails to execute successfully, it will always return an error (SQLSTATE 38000), along with diagnostic text providing information about the cause of the failure. For example, if the procedure name provided to GET ROUTINE does not identify an SQL procedure, diagnostic ″-204, 42704″ text will be returned, where ″-204″ and ″42704″ are the SQLCODE and SQLSTATE, respectively, that identify the cause of the problem. The SQLCODE and SQLSTATE in this example indicate that the procedure name provided in the GET ROUTINE command is undefined.

# GET SNAPSHOT

Collects status information and formats the output for the user. The information returned represents a *snapshot* of the database manager operational status at the time the command was issued.

**Scope:**

In a partitioned database environment, this command can be invoked from any database partition defined in the `db2nodes.cfg` file. It acts only on that partition.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance. If there is no instance attachment, a default instance attachment is created.

To obtain a snapshot of a remote instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──GET SNAPSHOT FOR──────────────────────────────────────────────────────►
```

```
►──┬─┬─DATABASE MANAGER─┬──────────────────────────────────────┬──────────────────────────►
   │ ├─DB MANAGER───────┤                                      │
   │ └─DBM──────────────┘                                      │
   ├─ALL──┬─────┬─DATABASES────────────────────────────────────┤
   │      └─DCS─┘                                              │
   ├─ALL──┬─────┬─APPLICATIONS─────────────────────────────────┤
   │      └─DCS─┘                                              │
   ├─ALL BUFFERPOOLS───────────────────────────────────────────┤
   ├─┬─────┬─APPLICATION──┬─APPLID──appl-id──────┬─────────────┤
   │ └─DCS─┘              └─AGENTID──appl-handle─┘             │
   ├─FCM FOR ALL DBPARTITIONNUMS───────────────────────────────┤
   ├─LOCKS FOR APPLICATION──┬─APPLID──appl-id──────┬───────────┤
   │                        └─AGENTID──appl-handle─┘           │
   ├─ALL REMOTE_DATABASES──────────────────────────────────────┤
   ├─ALL REMOTE_APPLICATIONS───────────────────────────────────┤
   └─┬─ALL─────────────────────────────┬─ON──database-alias────┘
     ├─┬─────┬─┬─DATABASE─┬────────────┤
     │ └─DCS─┘ └─DB───────┘            │
     ├─┬─────┬─APPLICATIONS────────────┤
     │ └─DCS─┘                         │
     ├─TABLES──────────────────────────┤
     ├─TABLESPACES─────────────────────┤
     ├─LOCKS───────────────────────────┤
     ├─BUFFERPOOLS─────────────────────┤
     ├─REMOTE_DATABASES────────────────┤
     ├─REMOTE_APPLICATIONS─────────────┤
     └─DYNAMIC SQL──┬───────────────┬──┘
                    └─WRITE TO FILE─┘

►──┬─────────────────────────────────────────┬──────────────────────────────────────►◄
   ├─AT DBPARTITIONNUM──db-partition-number──┤
   └─GLOBAL──────────────────────────────────┘
```

**Notes:**

1. The monitor switches must be turned on in order to collect some statistics.

**Command parameters:**

**DATABASE MANAGER**
Provides statistics for the active database manager instance.

**ALL DATABASES**
Provides general statistics for all active databases on the current database partition.

**ALL APPLICATIONS**
Provides information about all active applications that are connected to a database on the current database partition.

**ALL BUFFERPOOLS**
Provides information about buffer pool activity for all active databases.

**APPLICATION APPLID appl-id**
Provides information only about the application whose ID is specified. To get a specific application ID, use the LIST APPLICATIONS command.

**APPLICATION AGENTID appl-handle**
Provides information only about the application whose application handle is specified. The application handle is a 32-bit number that uniquely identifies an application that is currently running. Use the LIST APPLICATIONS command to get a specific application handle.

**FCM FOR ALL DBPARTITIONNUMS**
Provides Fast Communication Manager (FCM) statistics between the database partition against which the GET SNAPSHOT command was issued and the other database partitions in the partitioned database environment.

**LOCKS FOR APPLICATION APPLID appl-id**
Provides information about all locks held by the specified application, identified by application ID.

**LOCKS FOR APPLICATION AGENTID appl-handle**
Provides information about all locks held by the specified application, identified by application handle.

**ALL REMOTE_DATABASES**
Provides general statistics about all active remote databases on the current database partition.

**ALL REMOTE_APPLICATIONS**
Provides information about all active remote applications that are connected to the current database partition.

**ALL ON database-alias**
Provides general statistics and information about all applications, tables, table spaces, buffer pools, and locks for a specified database.

**DATABASE ON database-alias**
Provides general statistics for a specified database.

**APPLICATIONS ON database-alias**
Provides information about all applications connected to a specified database.

**TABLES ON database-alias**
Provides information about tables in a specified database. This will include only those tables that have been accessed since the TABLE recording switch was turned on.

**TABLESPACES ON database-alias**
Provides information about table spaces for a specified database.

**LOCKS ON database-alias**
Provides information about every lock held by each application connected to a specified database.

**BUFFERPOOLS ON database-alias**
Provides information about buffer pool activity for the specified database.

**REMOTE_DATABASES ON database-alias**
Provides general statistics about all active remote databases for a specified database.

**REMOTE_APPLICATIONS ON database-alias**
Provides information about remote applications for a specified database.

**DYNAMIC SQL ON database-alias**
Returns a point-in-time picture of the contents of the SQL statement cache for the database.

**WRITE TO FILE**
Specifies that snapshot results are to be stored in a file at the server, as well as being passed back to the client. This command is valid only over a

database connection. The snapshot data can then be queried through the table function SYSFUN.SQLCACHE_SNAPSHOT over the same connection on which the call was made.

**DCS** Depending on which clause it is specified, this keyword requests statistics about:

- A specific DCS application currently running on the DB2 Connect Gateway
- All DCS applications
- All DCS applications currently connected to a specific DCS database
- A specific DCS database
- All DCS databases.

**AT DBPARTITIONNUM db-partition-number**
Returns results for the database partition specified.

**GLOBAL**
Returns an aggregate result for all database partitions in a partitioned database system.

**Examples:**

In the following sample output listings, some of the information might not be available, depending on whether or not the appropriate database system monitor recording switch is turned on. If the information is unavailable, `Not Collected` appears in the output.

The following is typical output resulting from a request for database manager information:

```
            Database Manager Snapshot

Node type                                      = Database Server with local clients
Instance name                                  = minweiw2
Number of database partitions in DB2 instance  = 1
Database manager status                        = Active

Product name                                   = DB2 v8.1.0.64
Service level                                  = n040215 (U488485)

Private Sort heap allocated                    = 0
Private Sort heap high water mark              = 0
Post threshold sorts                           = 0
Piped sorts requested                          = 0
Piped sorts accepted                           = 0

Start Database Manager timestamp               = 02/17/2004 12:17:21.493836
Last reset timestamp                           =
Snapshot timestamp                             = 02/17/2004 12:19:12.210537

Remote connections to db manager               = 0
Remote connections executing in db manager     = 0
Local connections                              = 2
Local connections executing in db manager      = 1
Active local databases                         = 1

High water mark for agents registered          = 3
High water mark for agents waiting for a token = 0
Agents registered                              = 3
Agents waiting for a token                     = 0
Idle agents                                    = 0
```

```
Committed private Memory (Bytes)               = 835584

Switch list for db partition number 0
Buffer Pool Activity Information  (BUFFERPOOL) = ON  02/17/2004 12:17:21.493836
Lock Information                        (LOCK) = ON  02/17/2004 12:17:21.493836
Sorting Information                     (SORT) = ON  02/17/2004 12:17:21.493836
SQL Statement Information          (STATEMENT) = ON  02/17/2004 12:17:21.493836
Table Activity Information             (TABLE) = ON  02/17/2004 12:17:21.493836
Take Timestamp Information         (TIMESTAMP) = ON  02/17/2004 12:17:21.493836
Unit of Work Information                 (UOW) = ON  02/17/2004 12:17:21.493836

Agents assigned from pool                      = 1
Agents created from empty pool                 = 4
Agents stolen from another application         = 0
High water mark for coordinating agents        = 3
Max agents overflow                            = 0
Hash joins after heap threshold exceeded       = 0

Total number of gateway connections            = 0
Current number of gateway connections          = 0
Gateway connections waiting for host reply     = 0
Gateway connections waiting for client request = 0
Gateway connection pool agents stolen          = 0

Memory usage for database manager:

    Memory Pool Type                           = Database Monitor Heap
       Current size (bytes)                    = 180224
       High water mark (bytes)                 = 180224
       Configured size (bytes)                 = 376832

    Memory Pool Type                           = Other Memory
       Current size (bytes)                    = 4751360
       High water mark (bytes)                 = 4751360
       Configured size (bytes)                 = 18071552
```

The following is typical output resulting from a request for database information:

```

              Database Snapshot

Database name                              = SAMPLE
Database path                              = /home/minweiw2/minweiw2
                                              /NODE0000/SQL00001/
Input database alias                       = SAMPLE
Database status                            = Active
Catalog database partition number          = 0
Catalog network node name                  =
Operating system running at database server= AIX
Location of the database                   = Local
First database connect timestamp           = 02/17/2004 12:17:25.076527
Last reset timestamp                       =
Last backup timestamp                      =
Snapshot timestamp                         = 02/17/2004 12:19:11.548218

High water mark for connections            = 2
Application connects                       = 2
Secondary connects total                   = 0
Applications connected currently           = 2
Appls. executing in db manager currently   = 1
Agents associated with applications        = 2
Maximum agents associated with applications= 2
Maximum coordinating agents                = 2

Locks held currently                       = 7
Lock waits                                 = 1
Time database waited on locks (ms)         = 26039
```

```
Lock list memory in use (Bytes)            = 2304
Deadlocks detected                         = 0
Lock escalations                           = 0
Exclusive lock escalations                 = 0
Agents currently waiting on locks          = 1
Lock Timeouts                              = 0
Number of indoubt transactions             = 0

Total Private Sort heap allocated          = 0
Total Shared Sort heap allocated           = 0
Shared Sort heap high water mark           = 0
Total sorts                                = 0
Total sort time (ms)                       = 0
Sort overflows                             = 0
Active sorts                               = 0

Buffer pool data logical reads             = 98
Buffer pool data physical reads            = 27
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Asynchronous pool data page reads          = 0
Buffer pool data writes                    = 2
Asynchronous pool data page writes         = 0
Buffer pool index logical reads            = 214
Buffer pool index physical reads           = 91
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Asynchronous pool index page reads         = 0
Buffer pool index writes                   = 0
Asynchronous pool index page writes        = 0
Total buffer pool read time (ms)           = 947
Total buffer pool write time (ms)          = 3
Total elapsed asynchronous read time       = 0
Total elapsed asynchronous write time      = 0
Asynchronous data read requests            = 0
Asynchronous index read requests           = 0
No victim buffers available                = 0
LSN Gap cleaner triggers                   = 0
Dirty page steal cleaner triggers          = 0
Dirty page threshold cleaner triggers      = 0
Time waited for prefetch (ms)              = 0
Unread prefetch pages                      = 0
Direct reads                               = 42
Direct writes                              = 4
Direct read requests                       = 7
Direct write requests                      = 2
Direct reads elapsed time (ms)             = 0
Direct write elapsed time (ms)             = 1
Database files closed                      = 0
Data pages copied to extended storage      = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0

Host execution elapsed time                = 0.069848

Commit statements attempted                = 2
Rollback statements attempted              = 0
Dynamic statements attempted               = 8
Static statements attempted                = 2
Failed statement operations                = 0
Select SQL statements executed             = 1
Update/Insert/Delete statements executed   = 1
DDL statements executed                    = 2

Internal automatic rebinds                 = 0
Internal rows deleted                      = 0
Internal rows inserted                     = 0
Internal rows updated                      = 0
Internal commits                           = 2
Internal rollbacks                         = 0
Internal rollbacks due to deadlock         = 0

Rows deleted                               = 0
Rows inserted                              = 1
Rows updated                               = 0
Rows selected                              = 0
Rows read                                  = 31
Binds/precompiles attempted                = 0

Log space available to the database (Bytes)= 20395444
Log space used by the database (Bytes)     = 4556
Maximum secondary log space used (Bytes)   = 0
Maximum total log space used (Bytes)       = 6031
Secondary logs allocated currently         = 0
Log pages read                             = 0
Log read time (sec.ns)                     = 0.000000004
Log pages written                          = 6
Log write time (sec.ns)                    = 0.000000004
Number write log IOs                       = 6
Number read log IOs                        = 0
Number partial page log IOs                = 4
Number log buffer full                     = 0
Log data found in buffer                   = 0
Appl id holding the oldest transaction     = 7
Log to be redone for recovery (Bytes)      = 4464
Log accounted for by dirty pages (Bytes)   = 4424

File number of first active log            = 0
File number of last active log             = 2
File number of current active log          = 0
File number of log being archived          = Not applicable

Package cache lookups                      = 10
Package cache inserts                      = 8
Package cache overflows                    = 0
Package cache high water mark (Bytes)      = 207369
Application section lookups                = 8
Application section inserts                = 5

Catalog cache lookups                      = 20
Catalog cache inserts                      = 6
Catalog cache overflows                    = 0
Catalog cache high water mark              = 0

Workspace Information

 Shared high water mark                    = 0
 Corresponding shared overflows            = 0
 Total shared section inserts              = 0
 Total shared section lookups              = 0
 Private high water mark                   = 17692
 Corresponding private overflows           = 0
 Total private section inserts             = 5
 Total private section lookups             = 5

Number of hash joins                       = 0
Number of hash loops                       = 0
Number of hash join overflows              = 0
Number of small hash join overflows        = 0


Memory usage for database:

```

```
Memory Pool Type                           = Backup/Restore/Util Heap
   Current size (bytes)                    = 16384
   High water mark (bytes)                 = 16384
   Configured size (bytes)                 = 20496384

Memory Pool Type                           = Package Cache Heap
   Current size (bytes)                    = 262144
   High water mark (bytes)                 = 262144
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Catalog Cache Heap
   Current size (bytes)                    = 65536
   High water mark (bytes)                 = 65536
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Buffer Pool Heap
   Current size (bytes)                    = 4259840
   High water mark (bytes)                 = 4259840
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Buffer Pool Heap
   Current size (bytes)                    = 540672
   High water mark (bytes)                 = 540672
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Buffer Pool Heap
   Current size (bytes)                    = 278528
   High water mark (bytes)                 = 278528
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Buffer Pool Heap
   Current size (bytes)                    = 147456
   High water mark (bytes)                 = 147456
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Buffer Pool Heap
   Current size (bytes)                    = 81920
   High water mark (bytes)                 = 81920
   Configured size (bytes)                 = 4294950912

Memory Pool Type                           = Lock Manager Heap
   Current size (bytes)                    = 507904
   High water mark (bytes)                 = 507904
   Configured size (bytes)                 = 507904

Memory Pool Type                           = Database Heap
   Current size (bytes)                    = 3637248
   High water mark (bytes)                 = 3637248
   Configured size (bytes)                 = 8339456

Memory Pool Type                           = Other Memory
   Current size (bytes)                    = 0
   High water mark (bytes)                 = 0
   Configured size (bytes)                 = 12353536
```

The following is typical output resulting from a request for application information (by specifying either an application ID, an application handle, all applications, or all applications on a database):

```
            Application Snapshot

Application handle                         = 9
Application status                         = Lock-wait
Status change time                         = 02/17/2004 12:18:45.508734
Application code page                      = 850
Application country/region code            = 1
DUOW correlation token                     = *LOCAL.minweiw2.0F7397171829
```

```
Application name                           = db2bp
Application ID                             = *LOCAL.minweiw2.0F7397171829
Sequence number                            = 0001
TP Monitor client user ID                  =
TP Monitor client workstation name         =
TP Monitor client application name         =
TP Monitor client accounting string        =

Connection request start timestamp         = 02/17/2004 12:18:29.718212
Connect request completion timestamp       = 02/17/2004 12:18:29.735915
Application idle time                      = 0
CONNECT Authorization ID                   = MINWEIW2
Client login ID                            = minweiw2
Configuration NNAME of client              =
Client database manager product ID         = SQL08020
Process ID of client application           = 194360
Platform of client application             = AIX
Communication protocol of client           = Local Client

Inbound communication address              = *LOCAL.minweiw2

Database name                              = SAMPLE
Database path                              = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/
Client database alias                      = SAMPLE
Input database alias                       =
Last reset timestamp                       =
Snapshot timestamp                         = 02/17/2004 12:19:13.260464
The highest authority level granted        =
        Direct DBADM authority
        Direct CREATETAB authority
        Direct BINDADD authority
        Direct CONNECT authority
        Direct CREATE_NOT_FENC authority
        Direct LOAD authority
        Direct IMPLICIT_SCHEMA authority
        Direct CREATE_EXT_RT authority
        Direct QUIESCE_CONN authority
        Indirect SYSADM authority
        Indirect CREATETAB authority
        Indirect BINDADD authority
        Indirect CONNECT authority
        Indirect IMPLICIT_SCHEMA authority
Coordinating database partition number     = 0
Current database partition number          = 0
Coordinator agent process or thread ID     = 35384
Agents stolen                              = 0
Agents waiting on locks                    = 1
Maximum associated agents                  = 1
Priority at which application agents work  = 0
Priority type                              = Dynamic

Lock timeout (seconds)                     = -1
Locks held by application                  = 4
Lock waits since connect                   = 1
Time application waited on locks (ms)      = 27751
Deadlocks detected                         = 0
Lock escalations                           = 0
Exclusive lock escalations                 = 0
Number of Lock Timeouts since connected    = 0
Total time UOW waited on locks (ms)        = 27751

Total sorts                                = 0
Total sort time (ms)                       = 0
Total sort overflows                       = 0

Data pages copied to extended storage      = 0
```

```
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0
Buffer pool data logical reads             = 0
Buffer pool data physical reads            = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool data writes                    = 0
Buffer pool index logical reads            = 0
Buffer pool index physical reads           = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Buffer pool index writes                   = 0
Total buffer pool read time (ms)           = 0
Total buffer pool write time (ms)          = 0
Time waited for prefetch (ms)              = 0
Unread prefetch pages                      = 0
Direct reads                               = 0
Direct writes                              = 0
Direct read requests                       = 0
Direct write requests                      = 0
Direct reads elapsed time (ms)             = 0
Direct write elapsed time (ms)             = 0

Number of SQL requests since last commit   = 3
Commit statements                          = 0
Rollback statements                        = 0
Dynamic SQL statements attempted           = 3
Static SQL statements attempted            = 0
Failed statement operations                = 0
Select SQL statements executed             = 1
Update/Insert/Delete statements executed   = 0
DDL statements executed                    = 0
Internal automatic rebinds                 = 0
Internal rows deleted                      = 0
Internal rows inserted                     = 0
Internal rows updated                      = 0
Internal commits                           = 1
Internal rollbacks                         = 0
Internal rollbacks due to deadlock         = 0
Binds/precompiles attempted                = 0
Rows deleted                               = 0
Rows inserted                              = 0
Rows updated                               = 0
Rows selected                              = 0
Rows read                                  = 0
Rows written                               = 0

UOW log space used (Bytes)                 = 0
Previous UOW completion timestamp          = 02/17/2004 12:18:29.735915
Elapsed time of last completed uow (sec.ms)= 0.000000
UOW start timestamp                        = 02/17/2004 12:18:45.394125
UOW stop timestamp                         =
UOW completion status                      =

Open remote cursors                        = 0
Open remote cursors with blocking          = 0
Rejected Block Remote Cursor requests      = 0
Accepted Block Remote Cursor requests      = 1
Open local cursors                         = 1
Open local cursors with blocking           = 1
Total User CPU Time used by agent (s)      = 0.020000
Total System CPU Time used by agent (s)    = 0.100000
Host execution elapsed time                = 0.001853

Package cache lookups                      = 2
Package cache inserts                      = 1
```

```
Application section lookups                = 3
Application section inserts                = 1
Catalog cache lookups                      = 6
Catalog cache inserts                      = 0
Catalog cache overflows                    = 0
Catalog cache high water mark              = 0

Workspace Information

 Shared high water mark                    = 0
 Total shared overflows                    = 0
 Total shared section inserts              = 0
 Total shared section lookups              = 0
 Private high water mark                   = 14976
 Total private overflows                   = 0
 Total private section inserts             = 1
 Total private section lookups             = 1

Most recent operation                      = Fetch
Cursor name                                = SQLCUR201
Most recent operation start timestamp      = 02/17/2004 12:18:45.504828
Most recent operation stop timestamp       =
Agents associated with the application     = 1
Number of hash joins                       = 0
Number of hash loops                       = 0
Number of hash join overflows              = 0
Number of small hash join overflows        = 0
Statement type                             = Dynamic SQL Statement
Statement                                  = Fetch
Section number                             = 201
Application creator                        = NULLID
Package name                               = SQLC2E03
Consistency Token                          = AAAAAJHR
Package Version ID                         =
Cursor name                                = SQLCUR201
Statement database partition number        = 0
Statement start timestamp                  = 02/17/2004 12:18:45.504828
Statement stop timestamp                   =
Elapsed time of last completed stmt(sec.ms)= 0.001853
Total Statement user CPU time              = 0.000000
Total Statement system CPU time            = 0.000000
SQL compiler cost estimate in timerons     = 27
SQL compiler cardinality estimate          = 180
Degree of parallelism requested            = 1
Number of agents working on statement      = 1
Number of subagents created for statement  = 1
Statement sorts                            = 0
Total sort time                            = 0
Sort overflows                             = 0
Rows read                                  = 0
Rows written                               = 0
Rows deleted                               = 0
Rows updated                               = 0
Rows inserted                              = 0
Rows fetched                               = 0
Buffer pool data logical reads             = 0
Buffer pool data physical reads            = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads            = 0
Buffer pool index physical reads           = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Blocking cursor                            = YES
Dynamic SQL statement text:
select * from t1
```

```
    Agent process/thread ID                       = 35384

Agent process/thread ID                    = 35384
  Agent Lock timeout (seconds)             = -1
  Memory usage for agent:

    Memory Pool Type                       = Application Heap
       Current size (bytes)                = 147456
       High water mark (bytes)             = 147456
       Configured size (bytes)             = 1277952

    Memory Pool Type                       = Application Control Heap
       Current size (bytes)                = 16384
       High water mark (bytes)             = 16384
       Configured size (bytes)             = 704512

  ID of agent holding lock                 = 7
  Application ID holding lock              = *LOCAL.minweiw2.0307B7171724
  Lock name                                = 0x0002000D000000000000000054
  Lock attributes                          = 0x00000000
  Release flags                            = 0x00000001
  Lock object type                         = Table
  Lock mode                                = Exclusive Lock (X)
  Lock mode requested                      = Intention Share Lock (IS)
  Name of tablespace holding lock          = USERSPACE1
  Schema of table holding lock             = MINWEIW2
  Name of table holding lock               = T1
  Lock wait start timestamp                = 02/17/2004 12:18:45.508738

            Application Snapshot

Application handle                         = 7
Application status                         = UOW Waiting
Status change time                         = 02/17/2004 12:18:24.237397
Application code page                      = 850
Application country/region code            = 1
DUOW correlation token                     = *LOCAL.minweiw2.0307B7171724
Application name                           = db2bp
Application ID                             = *LOCAL.minweiw2.0307B7171724
Sequence number                            = 0003
TP Monitor client user ID                  =
TP Monitor client workstation name         =
TP Monitor client application name         =
TP Monitor client accounting string        =

Connection request start timestamp         = 02/17/2004 12:17:25.076527
Connect request completion timestamp       = 02/17/2004 12:17:27.198920
Application idle time                      = 49
CONNECT Authorization ID                   = MINWEIW2
Client login ID                            = minweiw2
Configuration NNAME of client              =
Client database manager product ID         = SQL08020
Process ID of client application           = 209018
Platform of client application             = AIX
Communication protocol of client           = Local Client

Inbound communication address              = *LOCAL.minweiw2

Database name                              = SAMPLE
Database path                              = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/
Client database alias                      = SAMPLE
Input database alias                       =
Last reset timestamp                       =
Snapshot timestamp                         = 02/17/2004 12:19:13.260464
The highest authority level granted        =
        Direct DBADM authority
        Direct CREATETAB authority
        Direct BINDADD authority
        Direct CONNECT authority
        Direct CREATE_NOT_FENC authority
        Direct LOAD authority
        Direct IMPLICIT_SCHEMA authority
        Direct CREATE_EXT_RT authority
        Direct QUIESCE_CONN authority
        Indirect SYSADM authority
        Indirect CREATETAB authority
        Indirect BINDADD authority
        Indirect CONNECT authority
        Indirect IMPLICIT_SCHEMA authority
Coordinating database partition number     = 0
Current database partition number          = 0
Coordinator agent process or thread ID     = 167996
Agents stolen                              = 0
Agents waiting on locks                    = 0
Maximum associated agents                  = 1
Priority at which application agents work  = 0
Priority type                              = Dynamic

Lock timeout (seconds)                     = -1
Locks held by application                  = 3
Lock waits since connect                   = 0
Time application waited on locks (ms)      = 0
Deadlocks detected                         = 0
Lock escalations                           = 0
Exclusive lock escalations                 = 0
Number of Lock Timeouts since connected    = 0
Total time UOW waited on locks (ms)        = 0

Total sorts                                = 0
Total sort time (ms)                       = 0
Total sort overflows                       = 0

Data pages copied to extended storage      = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0
Buffer pool data logical reads             = 98
Buffer pool data physical reads            = 27
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool data writes                    = 2
Buffer pool index logical reads            = 214
Buffer pool index physical reads           = 91
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Buffer pool index writes                   = 0
Total buffer pool read time (ms)           = 947
Total buffer pool write time (ms)          = 3
Time waited for prefetch (ms)              = 0
Unread prefetch pages                      = 0
Direct reads                               = 42
Direct writes                              = 4
Direct read requests                       = 7
Direct write requests                      = 2
Direct reads elapsed time (ms)             = 0
Direct write elapsed time (ms)             = 1

Number of SQL requests since last commit   = 2
Commit statements                          = 2
Rollback statements                        = 0
Dynamic SQL statements attempted           = 5
Static SQL statements attempted            = 2
Failed statement operations                = 0
```

```
Select SQL statements executed             = 0
Update/Insert/Delete statements executed   = 1
DDL statements executed                    = 2
Internal automatic rebinds                 = 0
Internal rows deleted                      = 0
Internal rows inserted                     = 0
Internal rows updated                      = 0
Internal commits                           = 1
Internal rollbacks                         = 0
Internal rollbacks due to deadlock         = 0
Binds/precompiles attempted                = 0
Rows deleted                               = 0
Rows inserted                              = 1
Rows updated                               = 0
Rows selected                              = 0
Rows read                                  = 31
Rows written                               = 9

UOW log space used (Bytes)                 = 159
Previous UOW completion timestamp          = 02/17/2004 12:18:13.052905
Elapsed time of last completed uow (sec.ms)= 0.137336
UOW start timestamp                        = 02/17/2004 12:18:18.844035
UOW stop timestamp                         =
UOW completion status                      =

Open remote cursors                        = 0
Open remote cursors with blocking          = 0
Rejected Block Remote Cursor requests      = 0
Accepted Block Remote Cursor requests      = 0
Open local cursors                         = 0
Open local cursors with blocking           = 0
Total User CPU Time used by agent (s)      = 0.300000
Total System CPU Time used by agent (s)    = 0.150000
Host execution elapsed time                = 0.067995

Package cache lookups                      = 8
Package cache inserts                      = 7
Application section lookups                = 5
Application section inserts                = 4
Catalog cache lookups                      = 14
Catalog cache inserts                      = 6
Catalog cache overflows                    = 0
Catalog cache high water mark              = 0

Workspace Information

 Shared high water mark                    = 0
 Total shared overflows                    = 0
 Total shared section inserts              = 0
 Total shared section lookups              = 0
 Private high water mark                   = 17692
 Total private overflows                   = 0
 Total private section inserts             = 4
 Total private section lookups             = 4

Most recent operation                      = Execute Immediate
Most recent operation start timestamp      = 02/17/2004 12:18:24.169317
Most recent operation stop timestamp       = 02/17/2004 12:18:24.237312
Agents associated with the application     = 1
Number of hash joins                       = 0
Number of hash loops                       = 0
Number of hash join overflows              = 0
Number of small hash join overflows        = 0
Statement type                             = Dynamic SQL Statement
Statement                                  = Execute Immediate
Section number                             = 203
Application creator                        = NULLID
```

```
Package name                               = SQLC2E03
Consistency Token                          = AAAAAJHR
Package Version ID                         =
Cursor name                                =
Statement database partition number        = 0
Statement start timestamp                  = 02/17/2004 12:18:24.169317
Statement stop timestamp                   = 02/17/2004 12:18:24.237312
Elapsed time of last completed stmt(sec.ms)= 0.067995
Total Statement user CPU time              = 0.010000
Total Statement system CPU time            = 0.060000
SQL compiler cost estimate in timerons     = 13
SQL compiler cardinality estimate          = 1
Degree of parallelism requested            = 1
Number of agents working on statement      = 0
Number of subagents created for statement  = 1
Statement sorts                            = 0
Total sort time                            = 0
Sort overflows                             = 0
Rows read                                  = 0
Rows written                               = 1
Rows deleted                               = 0
Rows updated                               = 0
Rows inserted                              = 0
Rows fetched                               = 0
Buffer pool data logical reads             = 1
Buffer pool data physical reads            = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads            = 0
Buffer pool index physical reads           = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Blocking cursor                            = NO
Dynamic SQL statement text:
insert into t1 values(1)


Agent process/thread ID                    = 167996
  Agent Lock timeout (seconds)             = -1
  Memory usage for agent:

    Memory Pool Type                       = Application Heap
       Current size (bytes)                = 212992
       High water mark (bytes)             = 212992
       Configured size (bytes)             = 1277952

    Memory Pool Type                       = Application Control Heap
       Current size (bytes)                = 16384
       High water mark (bytes)             = 16384
       Configured size (bytes)             = 704512
```

The following is typical output resulting from a request for buffer pool information:

```
             Bufferpool Snapshot

Bufferpool name                            = IBMDEFAULTBP
Database name                              = SAMPLE
Database path                              = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/
Input database alias                       = SAMPLE
Snapshot timestamp                         = 02/17/2004 12:19:14.265625

Buffer pool data logical reads             = 98
Buffer pool data physical reads            = 27
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool data writes                    = 2
Buffer pool index logical reads            = 214
Buffer pool index physical reads           = 91
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Total buffer pool read time (ms)           = 947
Total buffer pool write time (ms)          = 3
Asynchronous pool data page reads          = 0
Asynchronous pool data page writes         = 0
Buffer pool index writes                   = 0
Asynchronous pool index page reads         = 0
Asynchronous pool index page writes        = 0
Total elapsed asynchronous read time       = 0
Total elapsed asynchronous write time      = 0
Asynchronous data read requests            = 0
Asynchronous index read requests           = 0
No victim buffers available                = 0
Direct reads                               = 42
Direct writes                              = 4
Direct read requests                       = 7
Direct write requests                      = 2
Direct reads elapsed time (ms)             = 0
Direct write elapsed time (ms)             = 1
Database files closed                      = 0
Data pages copied to extended storage      = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0
Unread prefetch pages                      = 0
Vectored IOs                               = 0
Pages from vectored IOs                    = 0
Block IOs                                  = 0
Pages from block IOs                       = 0
Physical page maps                         = 0

Node number                                = 0
Tablespaces using bufferpool               = 3
Alter bufferpool information:
 Pages left to remove                      = 0
 Current size                              = 1000
 Post-alter size                           = 1000
```

The following is typical output resulting from a request for table information:

```
            Table Snapshot

First database connect timestamp     = 02/17/2004 12:17:25.076527
Last reset timestamp                 =
Snapshot timestamp                   = 02/17/2004 12:19:10.785689
Database name                        = SAMPLE
Database path                        = /home/minweiw2/minweiw2/
                                          NODE0000/SQL00001/
Input database alias                 = SAMPLE
Number of accessed tables            = 14

Table List
 Table Schema        = SYSIBM
 Table Name          = SYSTABLES
 Table Type          = Catalog
 Data Object Pages   = 27
 Index Object Pages  = 17
 LOB Object pages    = 256
 Rows Read           = 11
 Rows Written        = 2
 Overflows           = 0
 Page Reorgs         = 0
```

```
Table Schema        = SYSIBM
Table Name          = SYSCOLUMNS
Table Type          = Catalog
Data Object Pages   = 144
Index Object Pages  = 71
Rows Read           = 2
Rows Written        = 2
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSPLAN
Table Type          = Catalog
Data Object Pages   = 9
Index Object Pages  = 5
LOB Object pages    = 320
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSDBAUTH
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 3
Rows Read           = 3
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSTABAUTH
Table Type          = Catalog
Data Object Pages   = 5
Index Object Pages  = 13
Rows Read           = 1
Rows Written        = 2
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSEVENTMONITORS
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 3
LOB Object pages    = 64
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSTABLESPACES
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 7
Rows Read           = 5
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSSCHEMATA
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 3
```

```
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSUSERAUTH
Table Type          = Catalog
Data Object Pages   = 8
Index Object Pages  = 7
LOB Object pages    = 64
Rows Read           = 1
Rows Written        = 2
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSNODEGROUPS
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 3
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSBUFFERPOOLS
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 4
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSTBSPACEAUTH
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 4
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = SYSIBM
Table Name          = SYSVERSIONS
Table Type          = Catalog
Data Object Pages   = 1
Index Object Pages  = 3
Rows Read           = 1
Rows Written        = 0
Overflows           = 0
Page Reorgs         = 0

Table Schema        = MINWEIW2
Table Name          = T1
Table Type          = User
Data Object Pages   = 1
Rows Read           = 0
Rows Written        = 1
Overflows           = 0
Page Reorgs         = 0
```

The following is typical output resulting from a request for table space information:

```
             Tablespace Snapshot

First database connect timestamp           = 02/17/2004 12:17:25.076527
Last reset timestamp                       =
Snapshot timestamp                         = 02/17/2004 12:19:10.105473
Database name                              = SAMPLE
Database path                              = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/
Input database alias                       = SAMPLE
Number of accessed tablespaces             = 3


Tablespace name                            = SYSCATSPACE
  Tablespace ID                            = 0
  Tablespace Type                          = System managed space
  Tablespace Content Type                  = Any data
  Tablespace Page size (bytes)             = 4096
  Tablespace Extent size (pages)           = 32
  Automatic Prefetch size enabled          = Yes
  Buffer pool ID currently in use          = 1
  Buffer pool ID next startup              = 1
  File system caching                      = No
  Tablespace State                         = 0x'00000000'
   Detailed explanation:
     Normal
  Tablespace Prefetch size (pages)         = 32
  Total number of pages                    = 4475
  Number of usable pages                   = 4475
  Number of used pages                     = 4475
  Minimum Recovery Time                    =
  Number of quiescers                      = 0
  Number of containers                     = 1

  Container Name                           = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/SQLT0000.0
      Container ID                         = 0
      Container Type                       = Path
      Total Pages in Container             = 4475
      Usable Pages in Container            = 4475
      Stripe Set                           = 0
      Container is accessible              = Yes

  Buffer pool data logical reads           = 93
  Buffer pool data physical reads          = 26
  Buffer pool temporary data logical reads   = 0
  Buffer pool temporary data physical reads  = 0
  Asynchronous pool data page reads        = 0
  Buffer pool data writes                  = 0
  Asynchronous pool data page writes       = 0
  Buffer pool index logical reads          = 214
  Buffer pool index physical reads         = 91
  Buffer pool temporary index logical reads  = 0
  Buffer pool temporary index physical reads = 0
  Asynchronous pool index page reads       = 0
  Buffer pool index writes                 = 0
  Asynchronous pool index page writes      = 0
  Total buffer pool read time (ms)         = 946
  Total buffer pool write time (ms)        = 0
  Total elapsed asynchronous read time     = 0
  Total elapsed asynchronous write time    = 0
  Asynchronous data read requests          = 0
  Asynchronous index read requests         = 0
  No victim buffers available              = 0
  Direct reads                             = 42
  Direct writes                            = 4
  Direct read requests                     = 7
  Direct write requests                    = 2
  Direct reads elapsed time (ms)           = 0
  Direct write elapsed time (ms)           = 1
  Number of files closed                   = 0
  Data pages copied to extended storage    = 0
  Index pages copied to extended storage   = 0
  Data pages copied from extended storage  = 0
  Index pages copied from extended storage = 0


Tablespace name                            = TEMPSPACE1
  Tablespace ID                            = 1
  Tablespace Type                          = System managed space
  Tablespace Content Type                  = System Temporary data
  Tablespace Page size (bytes)             = 4096
  Tablespace Extent size (pages)           = 32
  Automatic Prefetch size enabled          = Yes
  Buffer pool ID currently in use          = 1
  Buffer pool ID next startup              = 1
  File system caching                      = No
  Tablespace State                         = 0x'00000000'
   Detailed explanation:
     Normal
  Tablespace Prefetch size (pages)         = 32
  Total number of pages                    = 1
  Number of usable pages                   = 1
  Number of used pages                     = 1
  Minimum Recovery Time                    =
  Number of quiescers                      = 0
  Number of containers                     = 1

  Container Name                           = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/SQLT0001.0
      Container ID                         = 0
      Container Type                       = Path
      Total Pages in Container             = 1
      Usable Pages in Container            = 1
      Stripe Set                           = 0
      Container is accessible              = Yes

  Buffer pool data logical reads           = 0
  Buffer pool data physical reads          = 0
  Buffer pool temporary data logical reads   = 0
  Buffer pool temporary data physical reads  = 0
  Asynchronous pool data page reads        = 0
  Buffer pool data writes                  = 0
  Asynchronous pool data page writes       = 0
  Buffer pool index logical reads          = 0
  Buffer pool index physical reads         = 0
  Buffer pool temporary index logical reads  = 0
  Buffer pool temporary index physical reads = 0
  Asynchronous pool index page reads       = 0
  Buffer pool index writes                 = 0
  Asynchronous pool index page writes      = 0
  Total buffer pool read time (ms)         = 0
  Total buffer pool write time (ms)        = 0
  Total elapsed asynchronous read time     = 0
  Total elapsed asynchronous write time    = 0
  Asynchronous data read requests          = 0
  Asynchronous index read requests         = 0
  No victim buffers available              = 0
  Direct reads                             = 0
  Direct writes                            = 0
  Direct read requests                     = 0
  Direct write requests                    = 0
  Direct reads elapsed time (ms)           = 0
  Direct write elapsed time (ms)           = 0
  Number of files closed                   = 0
  Data pages copied to extended storage    = 0
  Index pages copied to extended storage   = 0
  Data pages copied from extended storage  = 0
  Index pages copied from extended storage = 0


Tablespace name                            = USERSPACE1
  Tablespace ID                            = 2
  Tablespace Type                          = System managed space
  Tablespace Content Type                  = Any data
  Tablespace Page size (bytes)             = 4096
  Tablespace Extent size (pages)           = 32
  Automatic Prefetch size enabled          = Yes
  Buffer pool ID currently in use          = 1
  Buffer pool ID next startup              = 1
  File system caching                      = No
  Tablespace State                         = 0x'00000000'
   Detailed explanation:
     Normal
  Tablespace Prefetch size (pages)         = 32
  Total number of pages                    = 408
  Number of usable pages                   = 408
  Number of used pages                     = 408
  Minimum Recovery Time                    =
  Number of quiescers                      = 0
  Number of containers                     = 1

  Container Name                           = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/SQLT0002.0
      Container ID                         = 0
      Container Type                       = Path
      Total Pages in Container             = 408
      Usable Pages in Container            = 408
      Stripe Set                           = 0
      Container is accessible              = Yes

  Buffer pool data logical reads           = 5
  Buffer pool data physical reads          = 1
  Buffer pool temporary data logical reads   = 0
  Buffer pool temporary data physical reads  = 0
  Asynchronous pool data page reads        = 0
  Buffer pool data writes                  = 2
  Asynchronous pool data page writes       = 0
  Buffer pool index logical reads          = 0
  Buffer pool index physical reads         = 0
  Buffer pool temporary index logical reads  = 0
  Buffer pool temporary index physical reads = 0
  Asynchronous pool index page reads       = 0
  Buffer pool index writes                 = 0
  Asynchronous pool index page writes      = 0
  Total buffer pool read time (ms)         = 1
  Total buffer pool write time (ms)        = 3
  Total elapsed asynchronous read time     = 0
  Total elapsed asynchronous write time    = 0
  Asynchronous data read requests          = 0
  Asynchronous index read requests         = 0
  No victim buffers available              = 0
  Direct reads                             = 0
  Direct writes                            = 0
  Direct read requests                     = 0
  Direct write requests                    = 0
  Direct reads elapsed time (ms)           = 0
  Direct write elapsed time (ms)           = 0
  Number of files closed                   = 0
  Data pages copied to extended storage    = 0
  Index pages copied to extended storage   = 0
  Data pages copied from extended storage  = 0
  Index pages copied from extended storage = 0
```

The following sample shows output that results from a request for lock information. Ellipsis (...) replaces internal lock information that has been removed for clarity. Information for one internal lock remains.

```

            Database Lock Snapshot

Database name                              = SAMPLE
Database path                              = /home/minweiw2/minweiw2/
                                              NODE0000/SQL00001/
Input database alias                       = SAMPLE
Locks held                                 = 6
Applications currently connected           = 2
Agents currently waiting on locks          = 1
Snapshot timestamp                         = 02/17/2004 12:19:09.013588

Application handle                         = 9
Application ID                             = *LOCAL.minweiw2.0F7397171829
Sequence number                            = 0001
Application name                           = db2bp
CONNECT Authorization ID                   = MINWEIW2
Application status                         = Lock-wait
Status change time                         = 02/17/2004 12:18:45.508734
Application code page                      = 850
Locks held                                 = 3
Total wait time (ms)                       = 23504

  ID of agent holding lock                 = 7
  Application ID holding lock              = *LOCAL.minweiw2.0307B7171724
  Lock name                                = 0x0002000D0000000000000000054
  Lock attributes                          = 0x00000000
  Release flags                            = 0x00000001
  Lock object type                         = Table
  Lock mode                                = Exclusive Lock (X)
  Lock mode requested                      = Intention Share Lock (IS)
  Name of tablespace holding lock          = USERSPACE1
  Schema of table holding lock             = MINWEIW2
  Name of table holding lock               = T1
  Lock wait start timestamp                = 02/17/2004 12:18:45.508738

List Of Locks
 Lock Name                   = 0x00000001000000010001680056
 Lock Attributes             = 0x00000000
 Release Flags               = 0x40000000
 Lock Count                  = 1
 Hold Count                  = 0
 Lock Object Name            = 0
 Object Type                 = Internal Variation Lock
 Mode                        = S

 Lock Name                   = 0x53514C4332453033A95B579A41
 Lock Attributes             = 0x00000000
 Release Flags               = 0x40000000
 Lock Count                  = 1
 Hold Count                  = 0
 Lock Object Name            = 0
 Object Type                 = Internal Plan Lock
 Mode                        = S

 Lock Name                   = 0x53514C4445464C540763DD2841
 Lock Attributes             = 0x00000000
 Release Flags               = 0x40000000
 Lock Count                  = 1
 Hold Count                 = 0
 Lock Object Name           = 0
 Object Type                = Internal Plan Lock
 Mode                       = S


Application handle                         = 7
Application ID                             = *LOCAL.minweiw2.0307B7171724
Sequence number                            = 0003
Application name                           = db2bp
CONNECT Authorization ID                   = MINWEIW2
Application status                         = UOW Waiting
Status change time                         = 02/17/2004 12:18:24.237397
Application code page                      = 850
Locks held                                 = 3
Total wait time (ms)                       = 0

List Of Locks
 Lock Name                  = 0x0000000200001A09409060F043
 Lock Attributes            = 0x00000000
 Release Flags              = 0x40000000
 Lock Count                 = 4
 Hold Count                 = 0
 Lock Object Name           = 0
 Object Type                = Internal Catalog Cache Lock
 Mode                       = S

 Lock Name                  = 0x53514C4332453033A95B579A41
 Lock Attributes            = 0x00000000
 Release Flags              = 0x40000000
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 0
 Object Type                = Internal Plan Lock
 Mode                       = S

 Lock Name                  = 0x0002000D000000000000000054
 Lock Attributes            = 0x00000000
 Release Flags              = 0x40000000
 Lock Count                 = 255
 Hold Count                 = 0
 Lock Object Name           = 13
 Object Type                = Table
 Tablespace Name            = USERSPACE1
 Table Schema               = MINWEIW2
 Table Name                 = T1
 Mode                       = X
```

Additional application information appears when the LOCK switch is on, as shown in the following sample excerpt:

```
   ...
Application handle                         = 2
Application ID                             = *LOCAL.mikew.07B492160951
Sequence number                            = 0001
Application name                           = db2bp
Authorization ID                           = MIKEW
Application status                         = Lock-wait
Status change time                         = Not Collected
Application code page                      = 819
Locks held                                 = 9
Total wait time (ms)                       = 0

  Subsection waiting for lock              = 0
  ID of agent holding lock                 = 3
  Application ID holding lock              = *LOCAL.mikew.016A92161122
  Lock name                                = 0x00020002000000000000000054
  Lock attributes                          = 0x00
  Release flags                            = 0x40000001
  Lock object type                         = Table
  Lock mode                                = Intention Exclusive Lock (IX)
  Lock mode held                           = Intention Exclusive Lock (IX)
  Lock mode requested                      = Exclusive Lock (X)
  Name of tablespace holding lock          = USERSPACE1
  Schema of table holding lock             = MIKEW
  Name of table holding lock               = SNAPSHOT
  Lock wait start timestamp                = Not Collected
  Lock is a result of escalation           = NO
    ...
```

The following is typical output resulting from a request for dynamic SQL information:

```
     Dynamic SQL Snapshot Result

Database name                      = SAMPLE

Database path                      = /home/minweiw2/minweiw2/
                                      NODE0000/SQL00001/

Number of executions               = 1
Number of compilations             = 2
Worst preparation time (ms)        = 12
Best preparation time (ms)         = 12
Internal rows deleted              = 0
Internal rows inserted             = 0
Rows read                          = 16
Internal rows updated              = 0
Rows written                       = 4
Statement sorts                    = 0
Statement sort overflows           = 0
Total sort time                    = 0
Buffer pool data logical reads     = 46
Buffer pool data physical reads    = 17
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads    = 124
Buffer pool index physical reads   = 58
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Total execution time (sec.ms)      = 0.894210
Total user cpu time (sec.ms)       = 0.120000
Total system cpu time (sec.ms)     = 0.050000
Statement text                     = drop table t1


Number of executions               = 0
Number of compilations             = 0
Worst preparation time (ms)        = 0
Best preparation time (ms)         = 0
Internal rows deleted              = 0
Internal rows inserted             = 0
Rows read                          = 0
Internal rows updated              = 0
Rows written                       = 0
Statement sorts                    = 0
Buffer pool data logical reads     = 0
Buffer pool data physical reads    = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads    = 0
Buffer pool index physical reads   = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
```

```
Total execution time (sec.ms)      = 0.000000
Total user cpu time (sec.ms)       = 0.000000
Total system cpu time (sec.ms)     = 0.000000
Statement text                     = SET CURRENT LOCALE LC_CTYPE = 'en_US'


Number of executions               = 1
Number of compilations             = 1
Worst preparation time (ms)        = 73
Best preparation time (ms)         = 73
Internal rows deleted              = 0
Internal rows inserted             = 0
Rows read                          = 0
Internal rows updated              = 0
Rows written                       = 0
Statement sorts                    = 0
Buffer pool data logical reads     = 0
Buffer pool data physical reads    = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads    = 0
Buffer pool index physical reads   = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Total execution time (sec.ms)      = 0.000000
Total user cpu time (sec.ms)       = 0.000000
Total system cpu time (sec.ms)     = 0.000000
Statement text                     = select * from t1


Number of executions               = 1
Number of compilations             = 2
Worst preparation time (ms)        = 6
Best preparation time (ms)         = 6
Internal rows deleted              = 0
Internal rows inserted             = 0
Rows read                          = 1
Internal rows updated              = 0
Rows written                       = 0
Statement sorts                    = 0
Statement sort overflows           = 0
Total sort time                    = 0
Buffer pool data logical reads     = 1
Buffer pool data physical reads    = 0
Buffer pool temporary data logical reads   = 0
Buffer pool temporary data physical reads  = 0
Buffer pool index logical reads    = 2
Buffer pool index physical reads   = 0
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Total execution time (sec.ms)      = 0.011801
Total user cpu time (sec.ms)       = 0.010000
Total system cpu time (sec.ms)     = 0.000000
Statement text                     = lock table t1 in exclusive mode


Number of executions               = 1
Number of compilations             = 2
Worst preparation time (ms)        = 3
Best preparation time (ms)         = 3
Internal rows deleted              = 0
Internal rows inserted             = 0
Rows read                          = 4
Internal rows updated              = 0
Rows written                       = 4
Statement sorts                    = 0
Statement sort overflows           = 0
```

```
 Total sort time                    = 0
 Buffer pool data logical reads     = 26
 Buffer pool data physical reads    = 3
 Buffer pool temporary data logical reads   = 0
 Buffer pool temporary data physical reads  = 0
 Buffer pool index logical reads    = 44
 Buffer pool index physical reads   = 10
 Buffer pool temporary index logical reads  = 0
 Buffer pool temporary index physical reads = 0
 Total execution time (sec.ms)      = 0.129477
 Total user cpu time (sec.ms)       = 0.090000
 Total system cpu time (sec.ms)     = 0.000000
 Statement text                     = create table t1 (c1 int)


 Number of executions               = 1
 Number of compilations             = 1
 Worst preparation time (ms)        = 64
 Best preparation time (ms)         = 64
 Internal rows deleted              = 0
 Internal rows inserted             = 0
 Rows read                          = 0
 Internal rows updated              = 0
 Rows written                       = 1
 Statement sorts                    = 0
 Statement sort overflows           = 0
 Total sort time                    = 0
 Buffer pool data logical reads     = 1
 Buffer pool data physical reads    = 0
 Buffer pool temporary data logical reads   = 0
 Buffer pool temporary data physical reads  = 0
 Buffer pool index logical reads    = 0
 Buffer pool index physical reads   = 0
 Buffer pool temporary index logical reads  = 0
 Buffer pool temporary index physical reads = 0
 Total execution time (sec.ms)      = 0.067995
 Total user cpu time (sec.ms)       = 0.010000
 Total system cpu time (sec.ms)     = 0.060000
 Statement text                     = insert into t1 values(1)
```

The following is typical output resulting from a request for DCS application information (by specifying either a DCS application ID, a DCS application handle, all DCS applications, or all DCS applications on a database):

```
           DCS Application Snapshot

Client application ID                      = *LOCAL.andrewkm.010613200844
  Sequence number                          = 0001
  Authorization ID                         = AMURCHIS
  Application name                         = db2bp
  Application handle                       = 5
  Application status                       = waiting for request
  Status change time                       = 12-31-1969 19:00:00.000000
  Client node                              =
  Client release level                     = SQL07021
  Client platform                          = AIX
  Client protocol                          = Local Client
  Client codepage                          = 850
  Process ID of client application         = 36034
  Client login ID                          = andrewkm
  Host application ID                      = G9158067.CDF2.010613200845
  Sequence number                          = 0000
  Database alias at the gateway            = GSAMPLE
  DCS database name                        = SAMPLE
  Host database name                       = SAMPLE
  Host release level                       = SQL07021
  Host CCSID                               = 850
```

```
Outbound communication address             = 9.21.115.179 17336
Outbound communication protocol            = TCP/IP
Inbound communication address              = *LOCAL.andrewkm
First database connect timestamp           = 06-13-2001 16:08:44.142656
Host response time (sec.ms)                = 0.271230
Time spent on gateway processing           = 0.000119
Last reset timestamp                       =
Rows selected                              = 0
Number of SQL statements attempted         = 1
Failed statement operations                = 0
Commit statements                          = 1
Rollback statements                        = 0
Inbound bytes received                     = 184
Outbound bytes sent                        = 10
Outbound bytes received                    = 32
Inbound bytes sent                         = 0
Number of open cursors                     = 0
Application idle time                      = 1 minute and 33 seconds

UOW completion status                      = Committed - Commit Statement
Previous UOW completion timestamp          =
UOW start timestamp                        = 06-13-2001 16:08:44.716911
UOW stop timestamp                         = 06-13-2001 16:08:44.852730
Elapsed time of last completed uow (sec.ms)= 0.135819

Most recent operation                      = Static Commit
Most recent operation start timestamp      = 06-13-2001 16:08:44.716911
Most recent operation stop timestamp       = 06-13-2001 16:08:44.852730
Host execution elapsed time                = 0.000000
Statement                                  = Static Commit
Section number                             = 0
Application creator                        = NULLID
Package name                               = SQLC2D02
SQL compiler cost estimate in timerons     = 0
SQL compiler cardinality estimate          = 0
Statement start timestamp                  = 06-13-2001 16:08:44.716911
Statement stop timestamp                   = 06-13-2001 16:08:44.852730
Host response time (sec.ms)                = 0.271230
Elapsed time of last completed stmt(sec.ms)= 0.135819
Rows fetched                               = 0
Time spent on gateway processing           = 0.000119
Inbound bytes received for statement       = 184
Outbound bytes sent for statement          = 10
Outbound bytes received for statement      = 32
Inbound bytes sent for statement           = 0
Blocking cursor                            = NO
Outbound blocking cursor                   = NO
Host execution elapsed time                = 0.000000
```

The following is typical output resulting from a request for DCS database information:

```
             DCS Database Snapshot

DCS database name                          = SAMPLE

Host database name                         = SAMPLE
First database connect timestamp           = 06-13-2001 16:08:44.142656
Most recent elapsed time to connect        = 0.569354
Most recent elapsed connection duration    = 0.000000
Host response time (sec.ms)                = 0.271230
Last reset timestamp                       =
Number of SQL statements attempted         = 1
Commit statements attempted                = 1
Rollback statements attempted              = 0
Failed statement operations                = 0
```

```
Total number of gateway connections        = 1
Current number of gateway connections      = 1
Gateway conn. waiting for host reply       = 0
Gateway conn. waiting for client request   = 1
Gateway communication errors to host       = 0
Timestamp of last communication error      = None
High water mark for gateway connections    = 1
Rows selected                              = 0
Outbound bytes sent                        = 10
Outbound bytes received                    = 32
Host execution elapsed time                = 0.000000
```

**Usage notes:**

To obtain a snapshot from a remote instance (or a different local instance), it is necessary to first attach to that instance. If an alias for a database residing at a different instance is specified, an error message is returned.

To obtain some statistics, it is necessary that the database system monitor switches are turned on. If the recording switch TIMESTAMP has been set to off, timestamp related elements will report ″Not Collected″.

No data is returned following a request for table information if any of the following is true:

- The TABLE recording switch is turned off.
- No tables have been accessed since the switch was turned on.
- No tables have been accessed since the last RESET MONITOR command was issued.

However, if a REORG TABLE is being performed or has been performed during this period, some information is returned although some fields are not displayed.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.
- The keyword NODES can be substituted for DBPARTITIONNUMS.

**Related reference:**

- “GET MONITOR SWITCHES” on page 410
- “LIST APPLICATIONS” on page 480
- “RESET MONITOR” on page 643
- “UPDATE MONITOR SWITCHES” on page 740

# HELP

Permits the user to invoke help from the Information Center.

This command is not available on UNIX based systems.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►─HELP─┬──────────────────┬────────────────────────────────────────►◄
        └─character-string─┘
```

**Command parameters:**

**HELP character-string**
Any SQL or DB2 command, or any other item listed in the Information Center.

**Examples:**

Following are examples of the HELP command:

- `db2 help`

  This command opens the DB2 Information Center, which contains information about DB2 divided into categories, such as tasks, reference, books, and so on. This is equivalent to invoking the **db2ic** command with no parameters.
- `db2 help drop`

  This command opens the Web browser, and displays information about the SQL DROP statement. This is equivalent to invoking the following command: `db2ic -j drop`. The **db2ic** command searches first the SQL Reference and then the Command Reference, for a statement or a command called DROP, and then displays the first one found.
- `db2 help 'drop database'`

  This command initiates a more refined search, and causes information about the DROP DATABASE command to be displayed.

**Usage notes:**

The Information Center must be installed on the user's system. HTML books in the DB2 library must be located in the `\sqllib\doc\html` subdirectory.

The command line processor will not know if the command succeeds or fails, and cannot report error conditions.

# HISTORY

Displays the history of commands run within a CLP interactive mode session.

**Scope**

This command can only be run within CLP interactive mode. Specifically, it cannot be run from the CLP command mode or the CLP batch mode.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──┬─HISTORY─┬──┬─────────┬──┬─────┬──────────────────────────────►◄
    └─H───────┘  ├─REVERSE─┤  └─num─┘
                 └─R───────┘
```

**Command parameters:**

**REVERSE**
: Displays the command history in reverse order, with the most-recently run command listed first. If this parameter is not specified, the commands are listed in chronological order, with the most recently run command listed last.

*num*
: Displays only the most recent *num* commands. If this parameter is not specified, a maximum of 20 commands are displayed. However, the number of commands that are displayed is also restricted by the number of commands that are stored in the command history.

**Usage notes:**

1. The value of the DB2_CLP_HISTSIZE registry variable specifies the maximum number of commands to be stored in the command history. This registry variable can be set to any value between 1 and 500 inclusive. If this registry variable is not set or is set to a value outside the valid range, a maximum of 20 commands is stored in the command history.
2. Since the HISTORY command will always be listed in the command history, the maximum number of commands displayed will always be one greater than the user-specified maximum.
3. The command history is not persistent across CLP interactive mode sessions, which means that the command history is not saved at the end of an interactive mode session.
4. The command histories of multiple concurrently running CLP interactive mode sessions are independent of one another.

**Related reference:**

- “EDIT” on page 361
- “RUNCMD” on page 666

# IMPORT

Inserts data from an external file with a supported file format into a table, hierarchy, or view. LOAD is a faster alternative, but the load utility does not support loading data at the hierarchy level.

**Authorization:**

- IMPORT using the INSERT option requires one of the following:
  - *sysadm*
  - *dbadm*
  - CONTROL privilege on each participating table or view
  - INSERT and SELECT privilege on each participating table or view
- IMPORT to an existing table using the INSERT_UPDATE option, requires one of the following:
  - *sysadm*
  - *dbadm*
  - CONTROL privilege on the table or view
  - INSERT, SELECT, UPDATE and DELETE privilege on each participating table or view
- IMPORT to an existing table using the REPLACE or REPLACE_CREATE option, requires one of the following:
  - *sysadm*
  - *dbadm*
  - CONTROL privilege on the table or view
  - INSERT, SELECT, and DELETE privilege on the table or view
- IMPORT to a new table using the CREATE or REPLACE_CREATE option, requires one of the following:
  - *sysadm*
  - *dbadm*
  - CREATETAB authority on the database and USE privilege on the table space, as well as one of:
    - IMPLICIT_SCHEMA authority on the database, if the implicit or explicit schema name of the table does not exist
    - CREATIN privilege on the schema, if the schema name of the table refers to an existing schema
- IMPORT to a hierarchy that does not exist using the CREATE, or the REPLACE_CREATE option, requires one of the following:
  - *sysadm*
  - *dbadm*
  - CREATETAB authority on the database and USE privilege on the table space and one of:
    - IMPLICIT_SCHEMA authority on the database, if the schema name of the table does not exist
    - CREATEIN privilege on the schema, if the schema of the table exists
    - CONTROL privilege on every sub-table in the hierarchy, if the REPLACE_CREATE option on the entire hierarchy is used
- IMPORT to an existing hierarchy using the REPLACE option requires one of the following:

- *sysadm*
- *dbadm*
- CONTROL privilege on every sub-table in the hierarchy

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
>>--IMPORT FROM--filename--OF--filetype--+-------------------------+--+-------------------------------+-->
                                         |             .-,--------. |  |                .-------------. |
                                         |             V          | |  |                V             | |
                                         '-LOBS FROM----lob-path--+-'  '-MODIFIED BY----filetype-mod--+-'

>--+------------------------------------------------------------------------------------------+-->
   |              .-,-------------------------.                                                |
   |              V                           |                                                |
   '-METHOD--+-L--(----column-start--column-end--+--)--+-------------------------------------------+-+-'
             |                                          |                        .-,-------------------. | |
             |                                          |                        V                   | | |
             |                                          '-NULL INDICATORS--(----null-indicator-list--+--)-' |
             |       .-,------------.                                                                       |
             |       V              |                                                                       |
             +-N--(----column-name--+--)--------------------------------------------------------------------+
             |       .-,----------------.                                                                   |
             |       V                  |                                                                   |
             '-P--(----column-position--+--)--------------------------------------------------------------'

   .-ALLOW NO ACCESS----.
>--+--------------------+--+---------------------------+--+------------------------+--+-------------+-->
   '-ALLOW WRITE ACCESS-'  '-COMMITCOUNT--+-n---------+-'  '-+-RESTARTCOUNT-+--n--'  '-ROWCOUNT--n-'
                                          '-AUTOMATIC-'      '-SKIPCOUNT----'

>--+------------------+--+-----------+--+---------------------------+-->
   '-WARNINGCOUNT--n--'  '-NOTIMEOUT-'  '-MESSAGES--message-file-----'

>--+-+-INSERT---------+--INTO--+-table-name--+-------------------------+-+--------------------------------+-->
   | +-INSERT_UPDATE--+        |             |    .-,-------------.    | |                                |
   | +-REPLACE--------+        |             |    V               |    | |                                |
   | '-REPLACE_CREATE-'        |             '-(----insert-column--+--)-' |                                |
   |                           '-| hierarchy description |--------------'                                |
   '-CREATE--INTO--+-table-name--+-------------------------+-----------------------+--| tblspace-specs |--'
                   |             |    .-,-------------.    |                       |
                   |             |    V               |    |                       |
                   |             '-(----insert-column--+--)-'                       |
                   '-| hierarchy description |--+-AS ROOT TABLE-----------+-------'
                                                '-UNDER--sub-table-name---'

>--+------------------------------------------+--------------------------------------------------><
   '-DATALINK SPECIFICATION--| datalink-spec |-'
```

## hierarchy description:

```
   .-ALL TABLES---------------.
|--+--| sub-table-list |-----+--+----+--HIERARCHY--+-STARTING--sub-table-name-----+--------------|
                                '-IN-'             '-| traversal-order-list |------'
```

## sub-table-list:

```
      .-,---------------------------------------------------.
      V                                                     |
|--(----sub-table-name--+------------------------------+--+--)---------------------|
                        |    .-,-------------.          |
                        |    V               |          |
                        '-(----insert-column--+--)------'
```

### traversal-order-list:

```
      ┌─,──────────────┐
      ▼                │
├──(──────sub-table-name──┴──)──────────────────────────────────────────────────┤
```

### tblspace-specs:

```
├──┬─────────────────────────────────────────────────────────────────────────┬──────┤
   └─IN──tablespace-name──┬──────────────────────────┬──┬──────────────────────────┬─┘
                          └─INDEX IN──tablespace-name─┘  └─LONG IN──tablespace-name─┘
```

### datalink-spec:

```
     ┌─,──────────────────────────────────────────────────────────────────────────────┐
     ▼                                                                                 │
├────(──┬───────────────────┬──┬──────────────────────────────────┬──┬──────────────────────────┬──)─┴───┤
        └─DL_LINKTYPE URL───┘  ├─DL_URL_REPLACE_PREFIX──"prefix"──┤  └─DL_URL_SUFFIX──"suffix"──┘
                               └─DL_URL_DEFAULT_PREFIX──"prefix"──┘
```

**Command parameters:**

**ALL TABLES**
An implicit keyword for hierarchy only. When importing a hierarchy, the default is to import all tables specified in the traversal order.

**ALLOW NO ACCESS**
Runs import in the offline mode. An exclusive (X) lock on the target table is acquired before any rows are inserted. This prevents concurrent applications from accessing table data. This is the default import behavior.

**ALLOW WRITE ACCESS**
Runs import in the online mode. An intent exclusive (IX) lock on the target table is acquired when the first row is inserted. This allows concurrent readers and writers to access table data. Online mode is not compatible with the REPLACE, CREATE, or REPLACE_CREATE import options. Online mode is not supported in conjunction with buffered inserts. The import operation will periodically commit inserted data to prevent lock escalation to a table lock and to avoid running out of active log space. These commits will be performed even if the COMMITCOUNT option was not used. During each commit, import will lose its IX table lock, and will attempt to reacquire it after the commit.

**AS ROOT TABLE**
Creates one or more sub-tables as a stand-alone table hierarchy.

**COMMITCOUNT *n*/AUTOMATIC**
Performs a COMMIT after every *n* records are imported. When a number *n* is specified, import performs a COMMIT after every *n* records are imported. When compound inserts are used, a user-specified commit frequency of *n* is rounded up to the first integer multiple of the compound count value. When AUTOMATIC is specified, import internally determines when a commit needs to be performed. The utility will commit for either one of two reasons:

- to avoid running out of active log space
- to avoid lock escalation from row level to table level

If the ALLOW WRITE ACCESS option is specified, and the COMMITCOUNT option is not specified, the import utility will perform commits as if COMMITCOUNT AUTOMATIC had been specified.

**CREATE**

Creates the table definition and row contents in the code page of the database. If the data was exported from a DB2 table, sub-table, or hierarchy, indexes are created. If this option operates on a hierarchy, and data was exported from DB2, a type hierarchy will also be created. This option can only be used with IXF files.

**Note:** If the data was exported from an MVS host database, and it contains LONGVAR fields whose lengths, calculated on the page size, are less than 254, CREATE might fail because the rows are too long. See Using import to recreate an exported table for a list of restrictions. In this case, the table should be created manually, and IMPORT with INSERT should be invoked, or, alternatively, the LOAD command should be used.

**DATALINK SPECIFICATION**

For each DATALINK column, there can be one column specification enclosed by parentheses. Each column specification consists of one or more DL_LINKTYPE, prefix, and a DL_URL_SUFFIX specification. The prefix specification can be either DL_URL_REPLACE_PREFIX or DL_URL_DEFAULT_PREFIX.

There can be as many DATALINK column specifications as the number of DATALINK columns defined in the table. The order of specifications follows the order of DATALINK columns found within the *insert-column* list, or within the table definition (if an *insert-column* list is not specified).

**DL_LINKTYPE**

If specified, it should match the LINKTYPE of the column definition. Thus, DL_LINKTYPE URL is acceptable if the column definition specifies LINKTYPE URL.

**DL_URL_DEFAULT_PREFIX ″prefix″**

If specified, it should act as the default prefix for all DATALINK values within the same column. In this context, prefix refers to the ″scheme host port″ part of the URL specification.

Examples of prefix are:

```
"http://server"
"file://server"
"file:"
"http://server:80"
```

If no prefix is found in a column’s data, and a default prefix is specified with DL_URL_DEFAULT_PREFIX, the default prefix is prefixed to the column value (if not NULL).

For example, if DL_URL_DEFAULT_PREFIX specifies the default prefix `"http://toronto"`:

- The column input value ″/x/y/z″ is stored as ″http://toronto/x/y/z″.
- The column input value ″http://coyote/a/b/c″ is stored as ″http://coyote/a/b/c″.
- The column input value NULL is stored as NULL.

**DL_URL_REPLACE_PREFIX ″prefix″**
This clause is useful for loading or importing data previously generated by the export utility, when the user wants to globally replace the host name in the data with another host name. If specified, it becomes the prefix for *all* non-NULL column values. If a column value has a prefix, this will replace it. If a column value has no prefix, the prefix specified by DL_URL_REPLACE_PREFIX is prefixed to the column value.

For example, if DL_URL_REPLACE_PREFIX specifies the prefix `"http://toronto"`:

- The column input value ″/x/y/z″ is stored as ″http://toronto/x/y/z″.
- The column input value ″http://coyote/a/b/c″ is stored as ″http://toronto/a/b/c″. Note that ″toronto″ replaces ″coyote″.
- The column input value NULL is stored as NULL.

**DL_URL_SUFFIX ″suffix″**
If specified, it is appended to every non-NULL column value for the column. It is, in fact, appended to the ″path″ component of the URL part of the DATALINK value.

**FROM filename**
Specifies the file that contains the data to be imported. If the path is omitted, the current working directory is used.

**HIERARCHY**
Specifies that hierarchical data is to be imported.

**IN tablespace-name**
Identifies the table space in which the table will be created. The table space must exist, and must be a REGULAR table space. If no other table space is specified, all table parts are stored in this table space. If this clause is not specified, the table is created in a table space created by the authorization ID. If none is found, the table is placed into the default table space USERSPACE1. If USERSPACE1 has been dropped, table creation fails.

**INDEX IN tablespace-name**
Identifies the table space in which any indexes on the table will be created. This option is allowed only when the primary table space specified in the IN clause is a DMS table space. The specified table space must exist, and must be a REGULAR or LARGE DMS table space.

**Note:** Specifying which table space will contain an index can only be done when the table is created.

**insert-column**
Specifies the name of a column in the table or the view into which data is to be inserted.

**INSERT**
Adds the imported data to the table without changing the existing table data.

**INSERT_UPDATE**
Adds rows of imported data to the target table, or updates existing rows (of the target table) with matching primary keys.

**INTO table-name**
Specifies the database table into which the data is to be imported. This table cannot be a system table, a declared temporary table or a summary table.

One can use an alias for INSERT, INSERT_UPDATE, or REPLACE, except in the case of a down-level server, when the fully qualified or the unqualified table name should be used. A qualified table name is in the form: *schema.tablename*. The *schema* is the user name under which the table was created.

**LOBS FROM lob-path**
Specifies one or more paths that store LOB files. The names of the LOB data files are stored in the main data file (ASC, DEL, or IXF), in the column that will be loaded into the LOB column. This option is ignored if the `lobsinfile` modifier is not specified.

**LONG IN tablespace-name**
Identifies the table space in which the values of any long columns (LONG VARCHAR, LONG VARGRAPHIC, LOB data types, or distinct types with any of these as source types) will be stored. This option is allowed only if the primary table space specified in the IN clause is a DMS table space. The table space must exist, and must be a LARGE DMS table space.

**MESSAGES message-file**
Specifies the destination for warning and error messages that occur during an import operation. If the file already exists, the import utility appends the information. If the complete path to the file is not specified, the utility uses the current directory and the default drive as the destination. If *message-file* is omitted, the messages are written to standard output.

**METHOD**

**L** Specifies the start and end column numbers from which to import data. A column number is a byte offset from the beginning of a row of data. It is numbered starting from 1.

**Note:** This method can only be used with ASC files, and is the only valid option for that file type.

**N** Specifies the names of the columns to be imported.

**Note:** This method can only be used with IXF files.

**P** Specifies the field numbers of the input data fields to be imported.

**Note:** This method can only be used with IXF or DEL files, and is the only valid option for the DEL file type.

**MODIFIED BY filetype-mod**
Specifies file type modifier options. See File type modifiers for import.

**NOTIMEOUT**
Specifies that the import utility will not time out while waiting for locks. This option supersedes the *locktimeout* database configuration parameter. Other applications are not affected.

**NULL INDICATORS null-indicator-list**
This option can only be used when the METHOD L parameter is specified. That is, the input file is an ASC file. The null indicator list is a comma-separated list of positive integers specifying the column number of each null indicator field. The column number is the byte offset of the null indicator field from the beginning of a row of data. There must be one entry in the null indicator list for each data field defined in the METHOD L parameter. A column number of zero indicates that the corresponding data field always contains data.

A value of Y in the NULL indicator column specifies that the column data is NULL. Any character *other than* Y in the NULL indicator column specifies that the column data is not NULL, and that column data specified by the METHOD L option will be imported.

The NULL indicator character can be changed using the MODIFIED BY option, with the nullindchar file type modifier.

**OF filetype**

Specifies the format of the data in the input file:

- ASC (non-delimited ASCII format)
- DEL (delimited ASCII format), which is used by a variety of database manager and file manager programs
- WSF (work sheet format), which is used by programs such as:
  - Lotus 1-2-3
  - Lotus Symphony
- IXF (integrated exchange format, PC version), which means it was exported from the same or another DB2 table. An IXF file also contains the table definition and definitions of any existing indexes, except when columns are specified in the SELECT statement.

**REPLACE**

Deletes all existing data from the table by truncating the data object, and inserts the imported data. The table definition and the index definitions are not changed. This option can only be used if the table exists. It is not valid for tables with DATALINK columns. If this option is used when moving data between hierarchies, only the data for an entire hierarchy, not individual subtables, can be replaced.

**REPLACE_CREATE**

If the table exists, deletes all existing data from the table by truncating the data object, and inserts the imported data without changing the table definition or the index definitions.

If the table does not exist, creates the table and index definitions, as well as the row contents, in the code page of the database. See Using import to recreate an exported table for a list of restrictions.

This option can only be used with IXF files. It is not valid for tables with DATALINK columns. If this option is used when moving data between hierarchies, only the data for an entire hierarchy, not individual subtables, can be replaced.

**RESTARTCOUNT** *n*

Specifies that an import operation is to be started at record *n* + 1. The first *n* records are skipped. This option is functionally equivalent to SKIPCOUNT. RESTARTCOUNT and SKIPCOUNT are mutually exclusive.

**ROWCOUNT** *n*

Specifies the number *n* of physical records in the file to be imported (inserted or updated). Allows a user to import only *n* rows from a file, starting from the record determined by the SKIPCOUNT or RESTARTCOUNT options. If the SKIPCOUNT or RESTARTCOUNT options are not specified, the first *n* rows are imported. If SKIPCOUNT *m* or RESTARTCOUNT *m* is specified, rows *m*+1 to *m*+*n* are imported. When compound inserts are used, user specified rowcount n is rounded up to the first integer multiple of the compound count value.

**SKIPCOUNT** *n*
: Specifies that an import operation is to be started at record *n* + 1. The first *n* records are skipped. This option is functionally equivalent to RESTARTCOUNT. SKIPCOUNT and RESTARTCOUNT are mutually exclusive.

**STARTING sub-table-name**
: A keyword for hierarchy only, requesting the default order, starting from *sub-table-name*. For PC/IXF files, the default order is the order stored in the input file. The default order is the only valid order for the PC/IXF file format.

**sub-table-list**
: For typed tables with the INSERT or the INSERT_UPDATE option, a list of sub-table names is used to indicate the sub-tables into which data is to be imported.

**traversal-order-list**
: For typed tables with the INSERT, INSERT_UPDATE, or the REPLACE option, a list of sub-table names is used to indicate the traversal order of the importing sub-tables in the hierarchy.

**UNDER sub-table-name**
: Specifies a parent table for creating one or more sub-tables.

**WARNINGCOUNT** *n*
: Stops the import operation after *n* warnings. Set this parameter if no warnings are expected, but verification that the correct file and table are being used is desired. If the import file or the target table is specified incorrectly, the import utility will generate a warning for each row that it attempts to import, which will cause the import to fail. If *n* is zero, or this option is not specified, the import operation will continue regardless of the number of warnings issued.

**Examples:**

**Example 1**

The following example shows how to import information from `myfile.ixf` to the STAFF table:

```
   db2 import from myfile.ixf of ixf messages msg.txt insert into staff

SQL3150N  The H record in the PC/IXF file has product "DB2    01.00", date
"19970220", and time "140848".

SQL3153N  The T record in the PC/IXF file has name "myfile",
qualifier "        ", and source "            ".

SQL3109N  The utility is beginning to load data from file "myfile".

SQL3110N  The utility has completed processing.  "58" rows were read
from the input file.

SQL3221W  ...Begin COMMIT WORK. Input Record Count = "58".

SQL3222W  ...COMMIT of any database changes was successful.

SQL3149N  "58" rows were processed from the input file.  "58" rows were
successfully inserted into the table.  "0" rows were rejected.
```

**Example 2**

The following example shows how to import the table MOVIETABLE from the input file `delfile1`, which has data in the DEL format:

```
db2 import from delfile1 of del
    modified by dldel|
    insert into movietable (actorname, description, url_making_of,
    url_movie) datalink specification (dl_url_default_prefix
    "http://narang"), (dl_url_replace_prefix "http://bomdel"
    dl_url_suffix ".mpeg")
```

**Notes:**

1. The table has four columns:

   ```
   actorname             VARCHAR(n)
   description           VARCHAR(m)
   url_making_of         DATALINK (with LINKTYPE URL)
   url_movie             DATALINK (with LINKTYPE URL)
   ```
2. The DATALINK data in the input file has the vertical bar (|) character as the sub-field delimiter.
3. If any column value for url_making_of does not have the prefix character sequence, ″http://narang″ is used.
4. Each non-NULL column value for url_movie will get ″http://bomdel″ as its prefix. Existing values are replaced.
5. Each non-NULL column value for url_movie will get ″.mpeg″ appended to the path. For example, if a column value of url_movie is ″http://server1/x/y/z″, it will be stored as ″http://bomdel/x/y/z.mpeg″; if the value is ″/x/y/z″, it will be stored as ″http://bomdel/x/y/z.mpeg″.

**Example 3 (Importing into a Table with an Identity Column)**

TABLE1 has 4 columns:

- C1 VARCHAR(30)
- C2 INT GENERATED BY DEFAULT AS IDENTITY
- C3 DECIMAL(7,2)
- C4 CHAR(1)

TABLE2 is the same as TABLE1, except that C2 is a GENERATED ALWAYS identity column.

Data records in DATAFILE1 (DEL format):

```
"Liszt"
"Hummel",,187.43, H
"Grieg",100, 66.34, G
"Satie",101, 818.23, I
```

Data records in DATAFILE2 (DEL format):

```
"Liszt", 74.49, A
"Hummel", 0.01, H
"Grieg", 66.34, G
"Satie", 818.23, I
```

The following command generates identity values for rows 1 and 2, since no identity values are supplied in DATAFILE1 for those rows. Rows 3 and 4, however, are assigned the user-supplied identity values of 100 and 101, respectively.

```
db2 import from datafile1.del of del replace into table1
```

To import DATAFILE1 into TABLE1 so that identity values are generated for all rows, issue one of the following commands:

```
db2 import from datafile1.del of del method P(1, 3, 4)
   replace into table1 (c1, c3, c4)
db2 import from datafile1.del of del modified by identityignore
   replace into table1
```

To import DATAFILE2 into TABLE1 so that identity values are generated for each row, issue one of the following commands:

```
db2 import from datafile2.del of del replace into table1 (c1, c3, c4)
db2 import from datafile2.del of del modified by identitymissing
   replace into table1
```

If DATAFILE1 is imported into TABLE2 without using any of the identity-related file type modifiers, rows 1 and 2 will be inserted, but rows 3 and 4 will be rejected, because they supply their own non-NULL values, and the identity column is GENERATED ALWAYS.

**Usage notes:**

Be sure to complete all table operations and release all locks before starting an import operation. This can be done by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK.

The import utility adds rows to the target table using the SQL INSERT statement. The utility issues one INSERT statement for each row of data in the input file. If an INSERT statement fails, one of two actions result:

- If it is likely that subsequent INSERT statements can be successful, a warning message is written to the message file, and processing continues.
- If it is likely that subsequent INSERT statements will fail, and there is potential for database damage, an error message is written to the message file, and processing halts.

The utility performs an automatic COMMIT after the old rows are deleted during a REPLACE or a REPLACE_CREATE operation. Therefore, if the system fails, or the application interrupts the database manager after the table object is truncated, all of the old data is lost. Ensure that the old data is no longer needed before using these options.

If the log becomes full during a CREATE, REPLACE, or REPLACE_CREATE operation, the utility performs an automatic COMMIT on inserted records. If the system fails, or the application interrupts the database manager after an automatic COMMIT, a table with partial data remains in the database. Use the REPLACE or the REPLACE_CREATE option to rerun the whole import operation, or use INSERT with the RESTARTCOUNT parameter set to the number of rows successfully imported.

By default, automatic COMMITs are not performed for the INSERT or the INSERT_UPDATE option. They are, however, performed if the COMMITCOUNT parameter is not zero. If automatic COMMITs are not performed, a full log results in a ROLLBACK.

Offline import does not perform automatic COMMITs if any of the following conditions is true:

- the target is a view, not a table
- compound inserts are used
- buffered inserts are used

By default, online import performs automatic COMMITs to free both the active log space and the lock list. Automatic COMMITs are not performed only if a COMMITCOUNT value of zero is specified.

Whenever the import utility performs a COMMIT, two messages are written to the message file: one indicates the number of records to be committed, and the other is written after a successful COMMIT. When restarting the import operation after a failure, specify the number of records to skip, as determined from the last successful COMMIT.

The import utility accepts input data with minor incompatibility problems (for example, character data can be imported using padding or truncation, and numeric data can be imported with a different numeric data type), but data with major incompatibility problems is not accepted.

One cannot REPLACE or REPLACE_CREATE an object table if it has any dependents other than itself, or an object view if its base table has any dependents (including itself). To replace such a table or a view, do the following:

1. Drop all foreign keys in which the table is a parent.
2. Run the import utility.
3. Alter the table to recreate the foreign keys.

If an error occurs while recreating the foreign keys, modify the data to maintain referential integrity.

Referential constraints and foreign key definitions are not preserved when creating tables from PC/IXF files. (Primary key definitions *are* preserved if the data was previously exported using `SELECT *`.)

Importing to a remote database requires enough disk space on the server for a copy of the input data file, the output message file, and potential growth in the size of the database.

If an import operation is run against a remote database, and the output message file is very long (more than 60KB), the message file returned to the user on the client might be missing messages from the middle of the import operation. The first 30KB of message information and the last 30KB of message information are always retained.

Importing PC/IXF files to a remote database is much faster if the PC/IXF file is on a hard drive rather than on diskettes.

The database table or hierarchy must exist before data in the ASC, DEL, or WSF file formats can be imported; however, if the table does not already exist, IMPORT CREATE or IMPORT REPLACE_CREATE creates the table when it imports data from a PC/IXF file. For typed tables, IMPORT CREATE can create the type hierarchy and the table hierarchy as well.

PC/IXF import should be used to move data (including hierarchical data) between databases. If character data containing row separators is exported to a delimited ASCII (DEL) file and processed by a text transfer program, fields containing the row separators will shrink or expand. The file copying step is not necessary if the source and the target databases are both accessible from the same client.

The data in ASC and DEL files is assumed to be in the code page of the client application performing the import. PC/IXF files, which allow for different code pages, are recommended when importing data in different code pages. If the PC/IXF file and the import utility are in the same code page, processing occurs as for a regular application. If the two differ, and the FORCEIN option is specified, the import utility assumes that data in the PC/IXF file has the same code page as the application performing the import. This occurs even if there is a conversion table for the two code pages. If the two differ, the FORCEIN option is not specified, and there is a conversion table, all data in the PC/IXF file will be converted from the file code page to the application code page. If the two differ, the FORCEIN option is not specified, and there is no conversion table, the import operation will fail. This applies only to PC/IXF files on DB2 UDB clients on the AIX operating system.

For table objects on an 8 KB page that are close to the limit of 1012 columns, import of PC/IXF data files might cause DB2 to return an error, because the maximum size of an SQL statement was exceeded. This situation can occur only if the columns are of type CHAR, VARCHAR, or CLOB. The restriction does not apply to import of DEL or ASC files. If PC/IXF files are being used to create a new table, an alternative is use **db2look** to dump the DDL statement that created the table, and then to issue that statement through the CLP.

DB2 Connect can be used to import data to DRDA servers such as DB2 for OS/390, DB2 for VM and VSE, and DB2 for OS/400. Only PC/IXF import (INSERT option) is supported. The RESTARTCOUNT parameter, but not the COMMITCOUNT parameter, is also supported.

When using the CREATE option with typed tables, create every sub-table defined in the PC/IXF file; sub-table definitions cannot be altered. When using options other than CREATE with typed tables, the traversal order list enables one to specify the traverse order; therefore, the traversal order list must match the one used during the export operation. For the PC/IXF file format, one need only specify the target sub-table name, and use the traverse order stored in the file.

The import utility can be used to recover a table previously exported to a PC/IXF file. The table returns to the state it was in when exported.

Data cannot be imported to a system table, a declared temporary table, or a summary table.

Views cannot be created through the import utility.

On the Windows operating system:

- Importing logically split PC/IXF files is not supported.
- Importing bad format PC/IXF or WSF files is not supported.

**DB2 Data Links Manager considerations:**

Before running the DB2 import utility, do the following:

1. Copy the files that will be referenced to the appropriate Data Links servers. The **dlfm_import** utility can be used to extract files from an archive that is generated by the **dlfm_export** utility.
2. Register the required prefix names to the DB2 Data Links Managers. There might be other administrative tasks, such as registering the database, if required.

3. Update the Data Links server information in the URLs (of the DATALINK columns) from the exported data for the SQL table, if required. (If the original configuration's Data Links servers are the same at the target location, the Data Links server names need not be updated.)
4. Define the Data Links servers at the target configuration in the DB2 Data Links Manager configuration file.

When the import utility runs against the target database, files referred to by DATALINK column data are linked on the appropriate Data Links servers.

During the insert operation, DATALINK column processing links the files in the appropriate Data Links servers according to the column specifications at the target database.

**Related concepts:**

- "Import Overview" in the *Data Movement Utilities Guide and Reference*
- "Privileges, authorities, and authorization required to use import" in the *Data Movement Utilities Guide and Reference*

**Related tasks:**

- "Using import" in the *Data Movement Utilities Guide and Reference*

**Related reference:**

- "db2Import - Import" in the *Administrative API Reference*
- "db2look - DB2 Statistics and DDL Extraction Tool" on page 125
- "Import Sessions - CLP Examples" in the *Data Movement Utilities Guide and Reference*
- "LOAD" on page 520
- "File type modifiers for import" on page 461
- "Delimiter restrictions for moving data" on page 370

## File type modifiers for import

*Table 11. Valid file type modifiers for import: All file formats*

| Modifier | Description |
|---|---|
| compound=*x* | *x* is a number between 1 and 100 inclusive. Uses nonatomic compound SQL to insert the data, and *x* statements will be attempted each time.<br>If this modifier is specified, and the transaction log is not sufficiently large, the import operation will fail. The transaction log must be large enough to accommodate either the number of rows specified by COMMITCOUNT, or the number of rows in the data file if COMMITCOUNT is not specified. It is therefore recommended that the COMMITCOUNT option be specified to avoid transaction log overflow.<br>This modifier is incompatible with INSERT_UPDATE mode, hierarchical tables, and the following modifiers: `usedefaults`, `identitymissing`, `identityignore`, `generatedmissing`, and `generatedignore`. |
| generatedignore | This modifier informs the import utility that data for all generated columns is present in the data file but should be ignored. This results in all values for the generated columns being generated by the utility. This modifier cannot be used with the `generatedmissing` modifier. |

Table 11. Valid file type modifiers for import: All file formats (continued)

| Modifier | Description |
|---|---|
| generatedmissing | If this modifier is specified, the utility assumes that the input data file contains no data for the generated columns (not even NULLs), and will therefore generate a value for each row. This modifier cannot be used with the `generatedignore` modifier. |
| identityignore | This modifier informs the import utility that data for the identity column is present in the data file but should be ignored. This results in all identity values being generated by the utility. The behavior will be the same for both GENERATED ALWAYS and GENERATED BY DEFAULT identity columns. This means that for GENERATED ALWAYS columns, no rows will be rejected. This modifier cannot be used with the `identitymissing` modifier. |
| identitymissing | If this modifier is specified, the utility assumes that the input data file contains no data for the identity column (not even NULLs), and will therefore generate a value for each row. The behavior will be the same for both GENERATED ALWAYS and GENERATED BY DEFAULT identity columns. This modifier cannot be used with the `identityignore` modifier. |
| lobsinfile | *lob-path* specifies the path to the files containing LOB data.<br>Each path contains at least one file that contains at least one LOB pointed to by a Lob Location Specifier (LLS) in the data file. The LLS is a string representation of the location of a LOB in a file stored in the LOB file path. The format of an LLS is *filename.ext.nnn.mmm/*, where *filename.ext* is the name of the file that contains the LOB, *nnn* is the offset in bytes of the LOB within the file, and *mmm* is the length of the LOB in bytes. For example, if the string `db2exp.001.123.456/` is stored in the data file, the LOB is located at offset 123 in the file db2exp.001, and is 456 bytes long.<br>The LOBS FROM clause specifies where the LOB files are located when the “lobsinfile” modifier is used. The LOBS FROM clause means nothing outside the context of the `lobsinfile` modifier. The LOBS FROM clause conveys to the IMPORT utility the list of paths to search for the LOB files while importing the data.<br>To indicate a null LOB, enter the size as -1. If the size is specified as 0, it is treated as a 0 length LOB. For null LOBS with length of -1, the offset and the file name are ignored. For example, the LLS of a null LOB might be db2exp.001.7.-1/. |
| no_type_id | Valid only when importing into a single sub-table. Typical usage is to export data from a regular table, and then to invoke an import operation (using this modifier) to convert the data into a single sub-table. |
| nodefaults | If a source column for a target table column is not explicitly specified, and the table column is not nullable, default values are not loaded. Without this option, if a source column for one of the target table columns is not explicitly specified, one of the following occurs:<br>• If a default value can be specified for a column, the default value is loaded<br>• If the column is nullable, and a default value cannot be specified for that column, a NULL is loaded<br>• If the column is not nullable, and a default value cannot be specified, an error is returned, and the utility stops processing. |
| norowwarnings | Suppresses all warnings about rejected rows. |

*Table 11. Valid file type modifiers for import: All file formats (continued)*

| Modifier | Description |
|---|---|
| usedefaults | If a source column for a target table column has been specified, but it contains no data for one or more row instances, default values are loaded. Examples of missing data are:<br>• For DEL files: ",," is specified for the column<br>• For ASC files: The NULL indicator is set to yes for the column<br>• For DEL/ASC/WSF files: A row that does not have enough columns, or is not long enough for the original specification.<br>Without this option, if a source column contains no data for a row instance, one of the following occurs:<br>• If the column is nullable, a NULL is loaded<br>• If the column is not nullable, the utility rejects the row. |

*Table 12. Valid file type modifiers for import: ASCII file formats (ASC/DEL)*

| Modifier | Description |
|---|---|
| codepage=*x* | *x* is an ASCII character string. The value is interpreted as the code page of the data in the output data set. Converts character data to this code page from the application code page during the import operation.<br>The following rules apply:<br>• For pure DBCS (graphic) mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive.<br>• `nullindchar` must specify symbols included in the standard ASCII set between code points x20 ans x7F, inclusive. This refers to ASCII symbols and code points.<br>**Notes:**<br>1. The `codepage` modifier cannot be used with the `lobsinfile` modifier.<br>2. If data expansion occurs when the code page is converted from the application code page to the database code page, the data might be truncated and loss of data can occur. |
| dateformat="*x*" | *x* is the format of the date in the source file.[2] Valid date elements are:<br>`YYYY - Year (four digits ranging from 0000 - 9999)`<br>`M    - Month (one or two digits ranging from 1 - 12)`<br>`MM   - Month (two digits ranging from 1 - 12;`<br>`          mutually exclusive with M)`<br>`D    - Day (one or two digits ranging from 1 - 31)`<br>`DD   - Day (two digits ranging from 1 - 31;`<br>`          mutually exclusive with D)`<br>`DDD  - Day of the year (three digits ranging`<br>`          from 001 - 366; mutually exclusive`<br>`          with other day or month elements)`<br>A default value of 1 is assigned for each element that is not specified. Some examples of date formats are:<br>`"D-M-YYYY"`<br>`"MM.DD.YYYY"`<br>`"YYYYDDD"` |
| implieddecimal | The location of an implied decimal point is determined by the column definition; it is no longer assumed to be at the end of the value. For example, the value `12345` is loaded into a DECIMAL(8,2) column as `123.45`, *not* `12345.00`. |
| noeofchar | The optional end-of-file character `x'1A'` is not recognized as the end of file. Processing continues as if it were a normal character. |

*Table 12. Valid file type modifiers for import: ASCII file formats (ASC/DEL) (continued)*

| Modifier | Description |
|---|---|
| timeformat=*"x"* | *x* is the format of the time in the source file.[2] Valid time elements are:<br>`H     - Hour (one or two digits ranging from 0 - 12 for a 12 hour system, and 0 - 24 for a 24 hour system)`<br>`HH    - Hour (two digits ranging from 0 - 12 for a 12 hour system, and 0 - 24 for a 24 hour system; mutually exclusive with H)`<br>`M     - Minute (one or two digits ranging from 0 - 59)`<br>`MM    - Minute (two digits ranging from 0 - 59; mutually exclusive with M)`<br>`S     - Second (one or two digits ranging from 0 - 59)`<br>`SS    - Second (two digits ranging from 0 - 59; mutually exclusive with S)`<br>`SSSSS - Second of the day after midnight (5 digits ranging from 00000 - 86399; mutually exclusive with other time elements)`<br>`TT    - Meridian indicator (AM or PM)`<br>A default value of 0 is assigned for each element that is not specified. Some examples of time formats are:<br>`"HH:MM:SS"`<br>`"HH.MM TT"`<br>`"SSSSS"` |

*Table 12. Valid file type modifiers for import: ASCII file formats (ASC/DEL) (continued)*

| Modifier | Description |
|---|---|
| timestampformat="*x*" | *x* is the format of the time stamp in the source file.[2] Valid time stamp elements are:<br>`YYYY   - Year (four digits ranging from 0000 - 9999)`<br>`M      - Month (one or two digits ranging from 1 - 12)`<br>`MM     - Month (two digits ranging from 01 - 12;`<br>`            mutually exclusive with M and MMM)`<br>`MMM    - Month (three-letter case-insensitive abbreviation for`<br>`            the month name; mutually exclusive with M and MM)`<br>`D      - Day (one or two digits ranging from 1 - 31)`<br>`DD     - Day (two digits ranging from 1 - 31; mutually exclusive with D)`<br>`DDD    - Day of the year (three digits ranging from 001 - 366;`<br>`            mutually exclusive with other day or month elements)`<br>`H      - Hour (one or two digits ranging from 0 - 12`<br>`            for a 12 hour system, and 0 - 24 for a 24 hour system)`<br>`HH     - Hour (two digits ranging from 0 - 12`<br>`            for a 12 hour system, and 0 - 24 for a 24 hour system;`<br>`            mutually exclusive with H)`<br>`M      - Minute (one or two digits ranging from 0 - 59)`<br>`MM     - Minute (two digits ranging from 0 - 59;`<br>`            mutually exclusive with M, minute)`<br>`S      - Second (one or two digits ranging from 0 - 59)`<br>`SS     - Second (two digits ranging from 0 - 59;`<br>`            mutually exclusive with S)`<br>`SSSSS  - Second of the day after midnight (5 digits`<br>`            ranging from 00000 - 86399; mutually`<br>`            exclusive with other time elements)`<br>`UUUUUU - Microsecond (6 digits ranging from 000000 - 999999;`<br>`            mutually exclusive with all other microsecond elements)`<br>`UUUUU  - Microsecond (5 digits ranging  from 00000 - 99999,`<br>`            maps to range from 000000 - 999990;`<br>`            mutually exclusive with all other microseond elements)`<br>`UUUU   - Microsecond (4 digits ranging from 0000 - 9999,`<br>`            maps to range from 000000 - 999900;`<br>`            mutually exclusive with all other microseond elements)`<br>`UUU    - Microsecond (3 digits ranging from 000 - 999,`<br>`            maps to range from 000000 - 999000;`<br>`            mutually exclusive with all other microseond elements)`<br>`UU     - Microsecond (2 digits ranging from 00 - 99,`<br>`            maps to range from 000000 - 990000;`<br>`            mutually exclusive with all other microseond elements)`<br>`U      - Microsecond (1 digit ranging from 0 - 9,`<br>`            maps to range from 000000 - 900000;`<br>`            mutually exclusive with all other microseond elements)`<br>`TT     - Meridian indicator (AM or PM)`<br><br>A default value of 1 is assigned for unspecified YYYY, M, MM, D, DD, or DDD elements. A default value of 'Jan' is assigned to an unspecified MMM element. A default value of 0 is assigned for all other unspecified elements. Following is an example of a time stamp format:<br>`"YYYY/MM/DD HH:MM:SS.UUUUUU"`<br><br>The valid values for the MMM element include: 'jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov' and 'dec'. These values are case insensitive.<br><br>The following example illustrates how to import data containing user defined date and time formats into a table called schedule:<br>`db2 import from delfile2 of del`<br>`   modified by timestampformat="yyyy.mm.dd hh:mm tt"`<br>`   insert into schedule` |

*Table 12. Valid file type modifiers for import: ASCII file formats (ASC/DEL) (continued)*

| Modifier | Description |
|---|---|
| usegraphiccodepage | If usegraphiccodepage is given, the assumption is made that data being imported into graphic or double-byte character large object (DBCLOB) data fields is in the graphic code page. The rest of the data is assumed to be in the character code page. The graphic code page is associated with the character code page. IMPORT determines the character code page through either the codepage modifier, if it is specified, or through the code page of the application if the codepage modifier is not specified.<br>This modifier should be used in conjunction with the delimited data file generated by drop table recovery only if the table being recovered has graphic data.<br>**Restrictions**<br>The usegraphiccodepage modifier MUST NOT be specified with DEL or ASC files created by the EXPORT utility, as these files contain data encoded in only one code page. The usegraphiccodepage modifier is also ignored by the double-byte character large objects (DBCLOBs) in files. |

*Table 13. Valid file type modifiers for import: ASC (non-delimited ASCII) file format*

| Modifier | Description |
|---|---|
| nochecklengths | If nochecklengths is specified, an attempt is made to import each row, even if the source data has a column definition that exceeds the size of the target table column. Such rows can be successfully imported if code page conversion causes the source data to shrink; for example, 4-byte EUC data in the source could shrink to 2-byte DBCS data in the target, and require half the space. This option is particularly useful if it is known that the source data will fit in all cases despite mismatched column definitions. |
| nullindchar=*x* | *x* is a single character. Changes the character denoting a null value to *x*. The default value of *x* is Y.[3]<br>This modifier is case sensitive for EBCDIC data files, except when the character is an English letter. For example, if the null indicator character is specified to be the letter N, then n is also recognized as a null indicator. |
| reclen=*x* | *x* is an integer with a maximum value of 32 767. *x* characters are read for each row, and a new-line character is not used to indicate the end of the row. |
| striptblanks | Truncates any trailing blank spaces when loading data into a variable-length field. If this option is not specified, blank spaces are kept.<br>In the following example, striptblanks causes the import utility to truncate trailing blank spaces:<br>`db2 import from myfile.asc of asc`<br>`   modified by striptblanks`<br>`   method l (1 10, 12 15) messages msgs.txt`<br>`   insert into staff`<br>This option cannot be specified together with striptnulls. These are mutually exclusive options.<br>**Note:** This option replaces the obsolete t option, which is supported for back-level compatibility only. |

*Table 13. Valid file type modifiers for import: ASC (non-delimited ASCII) file format (continued)*

| **Modifier** | **Description** |
|---|---|
| striptnulls | Truncates any trailing NULLs (0x00 characters) when loading data into a variable-length field. If this option is not specified, NULLs are kept.<br>This option cannot be specified together with `striptblanks`. These are mutually exclusive options.<br>**Note:** This option replaces the obsolete `padwithzero` option, which is supported for back-level compatibility only. |

*Table 14. Valid file type modifiers for import: DEL (delimited ASCII) file format*

| **Modifier** | **Description** |
|---|---|
| chardel*x* | *x* is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.[34] If you want to explicitly specify the double quotation mark as the character string delimiter, it should be specified as follows:<br>`modified by chardel""`<br>The single quotation mark (') can also be specified as a character string delimiter. In the following example, `chardel''` causes the import utility to interpret any single quotation mark (') it encounters as a character string delimiter:<br>`db2 "import from myfile.del of del`<br>`   modified by chardel''`<br>`   method p (1, 4) insert into staff (id, years)"` |
| coldel*x* | *x* is a single character column delimiter. The default value is a comma (,). The specified character is used in place of a comma to signal the end of a column.[34]<br>In the following example, `coldel;` causes the import utility to interpret any semicolon (;) it encounters as a column delimiter:<br>`db2 import from myfile.del of del`<br>`   modified by coldel;`<br>`   messages msgs.txt insert into staff` |
| datesiso | Date format. Causes all date data values to be imported in ISO format. |
| decplusblank | Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign. |
| decpt*x* | *x* is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.[34]<br>In the following example, `decpt;` causes the import utility to interpret any semicolon (;) it encounters as a decimal point:<br>`db2 "import from myfile.del of del`<br>`   modified by chardel'`<br>`   decpt; messages msgs.txt insert into staff"` |

*Table 14. Valid file type modifiers for import: DEL (delimited ASCII) file format (continued)*

| **Modifier** | **Description** |
|---|---|
| delprioritychar | The current default priority for delimiters is: record delimiter, character delimiter, column delimiter. This modifier protects existing applications that depend on the older priority by reverting the delimiter priorities to: character delimiter, record delimiter, column delimiter. Syntax:<br>`db2 import ... modified by delprioritychar ...`<br>For example, given the following DEL data file:<br>`"Smith, Joshua",4000,34.98<row delimiter>`<br>`"Vincent,<row delimiter>, is a manager", ...`<br>`... 4005,44.37<row delimiter>`<br>With the `delprioritychar` modifier specified, there will be only two rows in this data file. The second <row delimiter> will be interpreted as part of the first data column of the second row, while the first and the third <row delimiter> are interpreted as actual record delimiters. If this modifier is *not* specified, there will be three rows in this data file, each delimited by a <row delimiter>. |
| dldel*x* | *x* is a single character DATALINK delimiter. The default value is a semicolon (;). The specified character is used in place of a semicolon as the inter-field separator for a DATALINK value. It is needed because a DATALINK value can have more than one sub-value. [34]<br>**Note:** *x* must not be the same character specified as the row, column, or character string delimiter. |
| keepblanks | Preserves the leading and trailing blanks in each field of type CHAR, VARCHAR, LONG VARCHAR, or CLOB. Without this option, all leading and trailing blanks that are not inside character delimiters are removed, and a NULL is inserted into the table for all blank fields. |
| nochardel | The import utility will assume all bytes found between the column delimiters to be part of the column's data. Character delimiters will be parsed as part of column data. This option should not be specified if the data was exported using DB2 (unless nochardel was specified at export time). It is provided to support vendor data files that do not have character delimiters. Improper usage might result in data loss or corruption.<br>This option cannot be specified with `chardelx`, `delprioritychar` or `nodoubledel`. These are mutually exclusive options. |
| nodoubledel | Suppresses recognition of double character delimiters. |

*Table 15. Valid file type modifiers for import: IXF file format*

| **Modifier** | **Description** |
|---|---|
| forcein | Directs the utility to accept data despite code page mismatches, and to suppress translation between code pages.<br>Fixed length target fields are checked to verify that they are large enough for the data. If `nochecklengths` is specified, no checking is done, and an attempt is made to import each row. |
| indexixf | Directs the utility to drop all indexes currently defined on the existing table, and to create new ones from the index definitions in the PC/IXF file. This option can only be used when the contents of a table are being replaced. It cannot be used with a view, or when a *insert-column* is specified. |
| indexschema=*schema* | Uses the specified *schema* for the index name during index creation. If *schema* is not specified (but the keyword `indexschema` *is* specified), uses the connection user ID. If the keyword is not specified, uses the schema in the IXF file. |

*Table 15. Valid file type modifiers for import: IXF file format (continued)*

| Modifier | Description |
|---|---|
| nochecklengths | If `nochecklengths` is specified, an attempt is made to import each row, even if the source data has a column definition that exceeds the size of the target table column. Such rows can be successfully imported if code page conversion causes the source data to shrink; for example, 4-byte EUC data in the source could shrink to 2-byte DBCS data in the target, and require half the space. This option is particularly useful if it is known that the source data will fit in all cases despite mismatched column definitions. |

**Notes:**

1. The import utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the import operation fails, and an error code is returned.
2. Double quotation marks around the date format string are mandatory. Field separators cannot contain any of the following: a-z, A-Z, and 0-9. The field separator should not be the same as the character delimiter or field delimiter in the DEL file format. A field separator is optional if the start and end positions of an element are unambiguous. Ambiguity can exist if (depending on the modifier) elements such as D, H, M, or S are used, because of the variable length of the entries.

   For time stamp formats, care must be taken to avoid ambiguity between the month and the minute descriptors, since they both use the letter M. A month field must be adjacent to other date fields. A minute field must be adjacent to other time fields. Following are some ambiguous time stamp formats:

   ```
   "M" (could be a month, or a minute)
   "M:M" (Which is which?)
   "M:YYYY:M" (Both are interpreted as month.)
   "S:M:YYYY" (adjacent to both a time value and a date value)
   ```

   In ambiguous cases, the utility will report an error message, and the operation will fail.

   Following are some unambiguous time stamp formats:

   ```
   "M:YYYY" (Month)
   "S:M" (Minute)
   "M:YYYY:S:M" (Month....Minute)
   "M:H:YYYY:M:D" (Minute....Month)
   ```

   Some characters, such as double quotation marks and back slashes, must be preceded by an escape character (for example, \).
3. The character must be specified in the code page of the source data.

   The character code point (instead of the character symbol), can be specified using the syntax xJJ or 0xJJ, where JJ is the hexadecimal representation of the code point. For example, to specify the # character as a column delimiter, use one of the following:

   ```
   ... modified by coldel# ...
   ... modified by coldel0x23 ...
   ... modified by coldelX23 ...
   ```
4. Delimiter restrictions for moving data lists restrictions that apply to the characters that can be used as delimiter overrides.

*Table 16. IMPORT behavior when using codepage and usegraphiccodepage*

| codepage=N | usegraphiccodepage | IMPORT behavior |
|---|---|---|
| Absent | Absent | All data in the file is assumed to be in the application code page. |

*Table 16. IMPORT behavior when using codepage and usegraphiccodepage (continued)*

| codepage=N | usegraphiccodepage | IMPORT behavior |
|---|---|---|
| Present | Absent | All data in the file is assumed to be in code page N.<br>**Warning:** Graphic data will be corrupted when imported into the database if N is a single-byte code page. |
| Absent | Present | Character data in the file is assumed to be in the application code page. Graphic data is assumed to be in the code page of the application graphic data.<br>If the application code page is single-byte, then all data is assumed to be in the application code page.<br>**Warning:** If the application code page is single-byte, graphic data will be corrupted when imported into the database, even if the database contains graphic columns. |
| Present | Present | Character data is assumed to be in code page N. Graphic data is assumed to be in the graphic code page of N.<br>If N is a single-byte or double-byte code page, then all data is assumed to be in code page N.<br>**Warning:** Graphic data will be corrupted when imported into the database if N is a single-byte code page. |

**Related reference:**

- "db2Import - Import" in the *Administrative API Reference*
- "IMPORT" on page 449
- "Delimiter restrictions for moving data" on page 370

## Delimiter restrictions for moving data

**Delimiter restrictions:**

It is the user's responsibility to ensure that the chosen delimiter character is not part of the data to be moved. If it is, unexpected errors might occur. The following restrictions apply to column, string, DATALINK, and decimal point delimiters when moving data:

- Delimiters are mutually exclusive.
- A delimiter cannot be binary zero, a line-feed character, a carriage-return, or a blank space.
- The default decimal point (.) cannot be a string delimiter.
- The following characters are specified differently by an ASCII-family code page and an EBCDIC-family code page:
  - The Shift-In (0x0F) and the Shift-Out (0x0E) character cannot be delimiters for an EBCDIC MBCS data file.
  - Delimiters for MBCS, EUC, or DBCS code pages cannot be greater than 0x40, except the default decimal point for EBCDIC MBCS data, which is 0x4b.
  - Default delimiters for data files in ASCII code pages or EBCDIC MBCS code pages are:

```
" (0x22, double quotation mark; string delimiter)
, (0x2c, comma; column delimiter)
```

- Default delimiters for data files in EBCDIC SBCS code pages are:

  ```
  " (0x7F, double quotation mark; string delimiter)
  , (0x6B, comma; column delimiter)
  ```

- The default decimal point for ASCII data files is 0x2e (period).
- The default decimal point for EBCDIC data files is 0x4B (period).
- If the code page of the server is different from the code page of the client, it is recommended that the hex representation of non-default delimiters be specified. For example,

  ```
  db2 load from ... modified by chardel0x0C coldelX1e ...
  ```

The following information about support for double character delimiter recognition in DEL files applies to the export, import, and load utilities:

- Character delimiters are permitted within the character-based fields of a DEL file. This applies to fields of type CHAR, VARCHAR, LONG VARCHAR, or CLOB (except when `lobsinfile` is specified). Any pair of character delimiters found between the enclosing character delimiters is imported or loaded into the database. For example,

  ```
  "What a ""nice"" day!"
  ```

  will be imported as:

  ```
  What a "nice" day!
  ```

  In the case of export, the rule applies in reverse. For example,

  ```
  I am 6" tall.
  ```

  will be exported to a DEL file as:

  ```
  "I am 6"" tall."
  ```

- In a DBCS environment, the pipe (|) character delimiter is not supported.

# INITIALIZE TAPE

When running on Windows NT-based operating systems, DB2 supports backup and restore operations to streaming tape devices. Use this command for tape initialization.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──INITIALIZE TAPE──┬─────────────┬──┬───────────────────┬─────────────────────────────►◄
                     └─ON──device──┘  └─USING──blksize──┘
```

**Command parameters:**

**ON device**
Specifies a valid tape device name. The default value is `\\.\TAPE0`.

**USING blksize**
Specifies the block size for the device, in bytes. The device is initialized to use the block size specified, if the value is within the supported range of block sizes for the device.

**Note:** The buffer size specified for the BACKUP DATABASE command and for RESTORE DATABASE must be divisible by the block size specified here.

If a value for this parameter is not specified, the device is initialized to use its default block size. If a value of zero is specified, the device is initialized to use a variable length block size; if the device does not support variable length block mode, an error is returned.

**Related reference:**

- "BACKUP DATABASE" on page 280
- "RESTORE DATABASE" on page 647
- "REWIND TAPE" on page 656
- "SET TAPE POSITION" on page 685

# INSPECT

Inspect database for architectural integrity, checking the pages of the database for page consistency. The inspection checks the structures of table objects and structures of table spaces are valid.

**Scope:**

In a single-partition system, the scope is that single partition only. In a partitioned database system, it is the collection of all logical partitions defined in db2nodes.cfg.

**Authorization:**

For INSPECT CHECK, one of the following:

- *sysadm*
- *dbadm*
- *sysctrl*
- *sysmaint*
- CONTROL privilege if single table.

**Required Connection:**

Database

**Command Syntax:**

```
>>-INSPECT--CHECK---------------------------------------------->

>--+-+-DATABASE--+-------------------------------------+-----------+-+------>
   | |           '-BEGIN TBSPACEID--n--+-------------+-'           | |
   | |                                 '-OBJECTID--n-'             | |
   | '-TABLESPACE--+-NAME--tablespace-name-+--+------------------+-' |
   |               '-TBSPACEID--n----------'  '-BEGIN OBJECTID--n-'   |
   '-TABLE--+-NAME--table-name--+---------------------+-+------------'
            |                   '-SCHEMA--schema-name-' |
            '-TBSPACEID--n--OBJECTID--n-----------------'

                             .-LIMIT ERROR TO DEFAULT--.
>--+---------------------+---+-------------------------+--+------------------+--->
   '-FOR ERROR STATE ALL-'   '-LIMIT ERROR TO--+-n---+-'  '-| Level Clause |-'
                                               '-ALL-'

>--RESULTS--+------+--filename--+----------------------------------+-><
            '-KEEP-'            '-| On Database Partition Clause |-'
```

### Level Clause:

```
   .-EXTENTMAP NORMAL-----.  .-DATA NORMAL-----.  .-BLOCKMAP NORMAL-----.
|--+----------------------+--+-----------------+--+---------------------+-->
   '-EXTENTMAP--+-NONE-+--'  '-DATA--+-NONE-+--'  '-BLOCKMAP--+-NONE-+--'
                '-LOW--'             '-LOW--'                 '-LOW--'
   ┌─INDEX NORMAL──────────┐  ┌─LONG NORMAL──────────┐  ┌─LOB NORMAL──────────┐
►──┤                       ├──┤                      ├──┤                     ├─────────────────────┤
   └─INDEX──────┬─NONE─┬───┘  └─LONG──────┬─NONE─┬───┘  └─LOB──────┬─NONE─┬───┘
                └─LOW──┘                  └─LOW──┘                └─LOW──┘
```

**On Database Partition Clause:**

```
├──ON──┬─┤ Database Partition List Clause ├──────────────────────────────────┬───┤
       └─ALL DBPARTITIONNUMS──────────────────────────────────────────────┬───┘
                                └─EXCEPT──┤ Database Partition List Clause ├─┘
```

**Database Partition List Clause:**

```
├──┬─DBPARTITIONNUM──┬──────────────────────────────────────────────────────────►
   └─DBPARTITIONNUMS─┘


      ┌─,──────────────────────────────────────────────┐
      ▼                                                │
►──(────db-partition-number1──┬──────────────────────────┬─┴──)──────────────────┤
                              └─TO──db-partition-number2─┘
```

**Command Parameters:**

**CHECK**
Specifies check processing.

**DATABASE**
Specifies whole database.

**BEGIN TBSPACEID n**
Specifies processing to begin from table space with given table space ID number.

**BEGIN TBSPACEID n OBJECTID n**
Specifies processing to begin from table with given table space ID number and object ID number.

**TABLESPACE**

**NAME tablespace-name**
Specifies single table space with given table space name.

**TBSPACEID n**
Specifies single table space with given table space ID number.

**BEGIN OBJECTID n**
Specifies processing to begin from table with given object ID number.

**TABLE**

**NAME table-name**
Specifies table with given table name.

**SCHEMA schema-name**
Specifies schema name for specified table name for single table operation.

**TBSPACEID n OBJECTID n**
Specifies table with given table space ID number and object ID number.

**FOR ERROR STATE ALL**
For table object with internal state already indicating error state, the check will just report this status and not scan through the object. Specifying this option will have the processing scan through the object even if internal state already lists error state.

**LIMIT ERROR TO** *n*
Number of pages in error for an object to limit reporting for. When this limit of the number of pages in error for an object is reached, the processing will discontinue the check on the rest of the object.

**LIMIT ERROR TO DEFAULT**
Default number of pages in error for an object to limit reporting for. This value is the extent size of the object. This parameter is the default.

**LIMIT ERROR TO ALL**
No limit on number of pages in error reported.

**EXTENTMAP**

**NORMAL**
Specifies processing level is normal for extent map. Default.

**NONE**
Specifies processing level is none for extent map.

**LOW** Specifies processing level is low for extent map.

**DATA**

**NORMAL**
Specifies processing level is normal for data object. Default.

**NONE**
Specifies processing level is none for data object.

**LOW** Specifies processing level is low for data object.

**BLOCKMAP**

**NORMAL**
Specifies processing level is normal for block map object. Default.

**NONE**
Specifies processing level is none for block map object.

**LOW** Specifies processing level is low for block map object.

**INDEX**

**NORMAL**
Specifies processing level is normal for index object. Default.

**NONE**
Specifies processing level is none for index object.

**LOW** Specifies processing level is low for index object.

**LONG**

**NORMAL**
Specifies processing level is normal for long object. Default.

**NONE**
Specifies processing level is none for long object.

**LOW** Specifies processing level is low for long object.

**LOB**

**NORMAL**
Specifies processing level is normal for LOB object. Default.

**NONE**
Specifies processing level is none for LOB object.

**LOW** Specifies processing level is low for LOB object.

**RESULTS**
Specifies the result output file. The file will be written out to the diagnostic data directory path. If there is no error found by the check processing, this result output file will be erased at the end of the INSPECT operation. If there are errors found by the check processing, this result output file will not be erased at the end of the INSPECT operation.

**KEEP** Specifies to always keep the result output file.

**file-name**
Specifies the name for the result output file.

**ALL DBPARTITIONNUMS**
Specifies that operation is to be done on all database partitions specified in the db2nodes.cfg file. This is the default if a node clause is not specified.

**EXCEPT**
Specifies that operation is to be done on all database partitions specified in the db2nodes.cfg file, except those specified in the node list.

**ON DBPARTITIONNUM / ON DBPARTITIONNUMS**
Perform operation on a set of database partitions.

**db-partition-number1**
Specifies a database partition number in the database partition list.

**db-partition-number2**
Specifies the second database partition number, so that all database partitions from db-partition-number1 up to and including db-partition-number2 are included in the database partition list.

**Usage Notes:**

1. For check operations on table objects, the level of processing can be specified for the objects. The default is NORMAL level, specifying NONE for an object excludes it. Specifying LOW will do subset of checks that are done for NORMAL.
2. The check database can be specified to start from a specific table space or from a specific table by specifying the ID value to identify the table space or the table.
3. The check table space can be specified to start from a specific table by specifying the ID value to identify the table.
4. The processing of table spaces will affect only the objects that reside in the table space.

5. The online inspect processing will access database objects using isolation level uncommitted read. COMMIT processing will be done during INSPECT processing. It is advisable to end the unit of work by issuing a COMMIT or ROLLBACK before invoking INSPECT.
6. The online inspect check processing will write out unformatted inspection data results to the results file specified. The file will be written out to the diagnostic data directory path. If there is no error found by the check processing, this result output file will be erased at the end of INSPECT operation. If there are errors found by the check processing, this result output file will not be erased at the end of INSPECT operation. After check processing completes, to see inspection details, the inspection result data will require to be formatted out with the utility db2inspf. The results file will have file extension of the database partition number. In a partitioned database environment, each database partition will generate its own results output file with extension corresponding to its database partition number The output location for the results output file will be the database manager diagnostic data directory path. If the name of a file that already exists is specified, the operation will not be processed, the file will have to be removed before that file name can be specified.

# LIST ACTIVE DATABASES

Displays a subset of the information listed by the GET SNAPSHOT FOR ALL DATABASES command. An active database is available for connection and use by any application. For each active database, this command displays the following:

- Database name
- Number of applications currently connected to the database
- Database path.

**Scope:**

This command can be issued from any database partition that is listed in `$HOME/sqllib/db2nodes.cfg`. It returns the same information from any of these database partitions.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Command syntax:**

```
>>──LIST ACTIVE DATABASES──┬─────────────────────────────────────────┬──────────><
                           ├─AT DBPARTITIONNUM──db-partition-number──┤
                           └─GLOBAL──────────────────────────────────┘
```

**Command parameters:**

**AT DBPARTITIONNUM db-partition-number**
Specifies the database partition for which the status of the monitor switches is to be displayed.

**GLOBAL**
Returns an aggregate result for all nodes in a partitioned database system.

**Examples:**

Following is sample output from the LIST ACTIVE DATABASES command:

```
                           Active Databases

Database name                           = TEST
Applications connected currently        = 0
Database path                           = /home/smith/smith/NODE0000/SQL00002/

Database name                           = SAMPLE
Applications connected currently        = 1
Database path                           = /home/smith/smith/NODE0000/SQL00001/
```

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "GET SNAPSHOT" on page 419
- "ACTIVATE DATABASE" on page 265
- "DEACTIVATE DATABASE" on page 342

# LIST APPLICATIONS

Displays to standard output the application program name, authorization ID (user name), application handle, application ID, and database name of all active database applications. This command can also optionally display an application's sequence number, status, status change time, and database path.

**Scope:**

This command only returns information for the database partition on which it is issued.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance. To list applications for a remote instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──LIST APPLICATIONS──┬──────────────────────────────────────┬──────────────────────►
                       └─FOR──┬─DATABASE─┬──database-alias─┘
                              └─DB───────┘

►──┬─────────────────────────────────────────┬──┬─────────────┬──────────────────────►◄
   ├─AT DBPARTITIONNUM──db-partition-number─┤  └─SHOW DETAIL─┘
   └─GLOBAL──────────────────────────────────┘
```

**Command parameters:**

**FOR DATABASE database-alias**

Information for each application that is connected to the specified database is to be displayed. Database name information is not displayed. If this option is not specified, the command displays the information for each application that is currently connected to any database at the database partition to which the user is currently attached.

The default application information is comprised of the following:

- Authorization ID
- Application program name
- Application handle
- Application ID
- Database name.

**AT DBPARTITIONNUM db-partition-number**

Specifies the database partition for which the status of the monitor switches is to be displayed.

**GLOBAL**
Returns an aggregate result for all database partitions in a partitioned database system.

**SHOW DETAIL**
Output will include the following additional information:

- Sequence #
- Application status
- Status change time
- Database path.

**Note:** If this option is specified, it is recommended that the output be redirected to a file, and that the report be viewed with the help of an editor. The output lines might wrap around when displayed on the screen.

**Examples:**

The following is sample output from LIST APPLICATIONS:

```
Auth Id  Application    Appl.      Application Id                 DB       # of
         Name           Handle                                    Name    Agents
-------- -------------- ---------- ------------------------------ -------- -----
smith    db2bp_32       12         *LOCAL.smith.970220191502      TEST     1
smith    db2bp_32       11         *LOCAL.smith.970220191453      SAMPLE   1
```

**Usage notes:**

The database administrator can use the output from this command as an aid to problem determination. In addition, this information is required if the database administrator wants to use the GET SNAPSHOT command or the FORCE APPLICATION command in an application.

To list applications at a remote instance (or a different local instance), it is necessary to first attach to that instance. If FOR DATABASE is specified when an attachment exists, and the database resides at an instance which differs from the current attachment, the command will fail.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- “GET SNAPSHOT” on page 419
- “FORCE APPLICATION” on page 372

# LIST COMMAND OPTIONS

Lists the current settings for the environment variables:

- **DB2BQTIME**
- **DB2DQTRY**
- **DB2RQTIME**
- **DB2IQTIME**
- **DB2OPTIONS**.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──LIST COMMAND OPTIONS──────────────────────────────────────────────►◄
```

**Command parameters:**

None

**Examples:**

The following is sample output from LIST COMMAND OPTIONS:

```
    Command Line Processor Option Settings

Backend process wait time (seconds)        (DB2BQTIME) = 1
No. of retries to connect to backend        (DB2BQTRY) = 60
Request queue wait time (seconds)          (DB2RQTIME) = 5
Input queue wait time (seconds)            (DB2IQTIME) = 5
Command options                           (DB2OPTIONS) =

Option  Description                               Current Setting
------  ----------------------------------------  ---------------
  -a    Display SQLCA                             OFF
  -c    Auto-Commit                               ON
  -e    Display SQLCODE/SQLSTATE                  OFF
  -f    Read from input file                      OFF
  -l    Log commands in history file              OFF
  -n    Remove new line character                 OFF
  -o    Display output                            ON
  -p    Display interactive input prompt          ON
  -r    Save output to report file                OFF
  -s    Stop execution on command error           OFF
  -t    Set statement termination character       OFF
  -v    Echo current command                      OFF
  -w    Display FETCH/SELECT warning messages     ON
  -z    Save all output to output file            OFF
```

**Related reference:**

- "UPDATE COMMAND OPTIONS" on page 726

# LIST DATABASE DIRECTORY

Lists the contents of the system database directory. If a path is specified, the contents of the local database directory are listed.

**Scope:**

If this command is issued without the ON *path* parameter, the system database directory is returned. This information is the same at all database partitions.

If the ON *path* parameter is specified, the local database directory on that path is returned. This information is not the same at all database partitions.

**Authorization:**

None

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──LIST──┬─DATABASE─┬──DIRECTORY──┬──────────────────┬──────────────────────────────►◄
          └─DB───────┘              └─ON──┬─path──┬──┘
                                          └─drive─┘
```

**Command parameters:**

**ON path/drive**
Specifies the local database directory from which to list information. If not specified, the contents of the system database directory are listed.

**Examples:**

The following shows sample output for a system database directory:

```
System Database Directory

Number of entries in the directory = 2

Database 1 entry:
  Database alias                       = SAMPLE
  Database name                        = SAMPLE
  Database drive                       = /home/smith
  Database release level               = 8.00
  Comment                              =
  Directory entry type                 = Indirect
  Catalog database partition number    = 0
  Alternate server hostname            = montero
  Alternate server port number         = 29384

Database 2 entry:
  Database alias                       = TC004000
  Database name                        = TC004000
  Node name                            = PRINODE
  Database release level               = a.00
  Comment                              =
  Directory entry type                 = LDAP
  Catalog database partition number    = -1
  Gateway node name                     = PRIGW
  Alternate server node name            =
  Alternate server gateway node name    = ALTGW
```

The following shows sample output for a local database directory:

```
Local Database Directory on /u/smith

Number of entries in the directory = 1

Database 1 entry:

  Database alias                       = SAMPLE
  Database name                        = SAMPLE
  Database directory                   = SQL00001
  Database release level               = 8.00
  Comment                              =
  Directory entry type                 = Home
  Catalog database partition number    = 0
  Database partition number            = 0
```

These fields are identified as follows:

**Database alias**
The value of the *alias* parameter when the database was created or cataloged. If an alias was not entered when the database was cataloged, the database manager uses the value of the *database-name* parameter when the database was cataloged.

**Database name**
The value of the *database-name* parameter when the database was cataloged. This name is usually the name under which the database was created.

**Local database directory**
The path on which the database resides. This field is filled in only if the system database directory has been scanned.

**Database directory/Database drive**
The name of the directory or drive where the database resides. This field is filled in only if the local database directory has been scanned.

**Node name**
The name of the remote node. This name corresponds to the value entered for the *nodename* parameter when the database and the node were cataloged.

**Database release level**
The release level of the database manager that can operate on the database.

**Comment**
Any comments associated with the database that were entered when it was cataloged.

**Directory entry type**
The location of the database:

- A *remote* entry describes a database that resides on another node.
- An *indirect* entry describes a database that is local. Databases that reside on the same node as the system database directory are thought to indirectly reference the home entry (to a local database directory), and are considered indirect entries.
- A *home* entry indicates that the database directory is on the same path as the local database directory.

- An LDAP entry indicates that the database location information is stored on an LDAP server.

All entries in the system database directory are either remote or indirect. All entries in local database directories are identified in the system database directory as indirect entries.

**Authentication**
The authentication type cataloged at the client.

**Principal name**
Specifies a fully qualified Kerberos principal name.

**Database partition number**
Specifies which node is the catalog database partition. This is the database partition on which the CREATE DATABASE command was issued.

**Database partition number**
Specifies the number that is assigned in `db2nodes.cfg` to the node where the command was issued.

**Alternate server hostname**
Specifies the host name or the IP address for the alternate server to be used when there is communication failure on the connection to the database. This field is displayed only for the system database directory.

**Alternate server port number**
Specifies the port number for the alternate server to be used when there is communication failure on the connection to the database. This field is displayed only for the system database directory.

**Alternate server node name**
If the directory entry type is LDAP, specifies the node name for the alternate server to be used when there is communication failure on the connection to the database.

**Alternate server gateway node name**
If the directory entry type is LDAP, specifies the gateway node name for the alternate gateway to be used when there is communication failure on the connection to the database.

**Usage notes:**

There can be a maximum of eight opened database directory scans per process. To overcome this restriction for a batch file that issues more than eight LIST DATABASE DIRECTORY commands within a single DB2 session, convert the batch file into a shell script. The ″db2″ prefix generates a new DB2 session for each command.

**Related reference:**

- “CHANGE DATABASE COMMENT” on page 327
- “CREATE DATABASE” on page 331
- “UPDATE ALTERNATE SERVER FOR DATABASE” on page 721

# LIST DATABASE PARTITION GROUPS

Lists all database partition groups associated with the current database.

**Scope:**

This command can be issued from any database partition that is listed in `$HOME/sqllib/db2nodes.cfg`. It returns the same information from any of these database partitions.

**Authorization:**

For the system catalogs `SYSCAT.DBPARTITIONGROUPS` and `SYSCAT.DBPARTITIONGROUPDEF`, one of the following is required:

- *sysadm* or *dbadm* authority
- CONTROL privilege
- SELECT privilege.

**Required connection:**

Database

**Command syntax:**

```
►►──LIST DATABASE PARTITION GROUPS──┬─────────────┬──────────────────────────►◄
                                    └─SHOW DETAIL─┘
```

**Command parameters:**

**SHOW DETAIL**
Specifies that the output should include the following information:

- Partitioning map ID
- Database partition number
- In-use flag

**Examples:**

Following is sample output from the LIST DATABASE PARTITION GROUPS command:

```
DATABASE PARTITION GROUP NAME
-----------------------------
IBMCATGROUP
IBMDEFAULTGROUP

  2 record(s) selected.
```

Following is sample output from the LIST DATABASE PARTITION GROUPS SHOW DETAIL command:

```
DATABASE PARTITION GROUP NAME  PMAP_ID DATABASE PARTITION NUMBER IN_USE
------------------------------ ------- ------------------------- ------
IBMCATGROUP                          0                         0 Y
IBMDEFAULTGROUP                      1                         0 Y

  2 record(s) selected.
```

The fields are identified as follows:

**DATABASE PARTITION GROUP NAME**
The name of the database partition group. The name is repeated for each database partition in the database partition group.

**PMAP_ID**
The ID of the partitioning map. The ID is repeated for each database partition in the database partition group.

**DATABASE PARTITION NUMBER**
The number of the database partition.

**IN_USE**
One of four values:

**Y** The database partition is being used by the database partition group.

**D** The database partition is going to be dropped from the database partition group as a result of a REDISTRIBUTE DATABASE PARTITION GROUP operation. When the operation completes, the database partition will not be included in reports from LIST DATABASE PARTITION GROUPS.

**A** The database partition has been added to the database partition group but is not yet added to the partitioning map. The containers for the table spaces in the database partition group have been added on this database partition. The value is changed to `Y` when the REDISTRIBUTE DATABASE PARTITION GROUP operation completes successfully.

**T** The database partition has been added to the database partition group, but is not yet added to the partitioning map. The containers for the table spaces in the database partition group have not been added on this database partition. Table space containers must be added on the new database partition for each table space in the database partition group. The value is changed to `A` when containers have successfully been added.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODEGROUPS can be substituted for DATABASE PARTITION GROUPS.

**Related reference:**

- "REDISTRIBUTE DATABASE PARTITION GROUP" on page 609

# LIST DATALINKS MANAGERS

Lists the DB2 Data Links Managers that are registered to a specified database.

**Authorization:**

None

**Command syntax:**

```
►►──LIST DATALINKS MANAGERS FOR──┬─DATABASE─┬──dbname────────────────────────►◄
                                 └─DB───────┘
```

**Command parameters:**

**DATABASE dbname**
Specifies a database name.

**Related reference:**

- "ADD DATALINKS MANAGER" on page 269
- "DROP DATALINKS MANAGER" on page 354

# LIST DBPARTITIONNUMS

Lists all database partitions associated with the current database.

**Scope:**

This command can be issued from any database partition that is listed in `$HOME/sqllib/db2nodes.cfg`. It returns the same information from any of these database partitions.

**Authorization:**

None

**Required connection:**

Database

**Command syntax:**

```
►►──LIST DBPARTITIONNUMS──────────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Examples:**
Following is sample output from the LIST DBPARTITIONNUMS command:

```
DATABASE PARTITION NUMBER
-------------------------
                        0
                        2
                        5
                        7
                        9

     5 record(s) selected.
```

**Compatibilities:**

For compatibility with versions earlier than Version 8:
- The keyword NODES can be substituted for DBPARTITIONNUMS.

**Related reference:**
- "REDISTRIBUTE DATABASE PARTITION GROUP" on page 609

# LIST DCS APPLICATIONS

Displays to standard output information about applications that are connected to host databases via DB2 Connect Enterprise Edition.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance. To list the DCS applications at a remote instance, it is necessary to first attach to that instance.

**Command syntax:**

```
►►──LIST DCS APPLICATIONS──┬─────────────┬────────────────────────────────►◄
                           ├─SHOW DETAIL─┤
                           └─EXTENDED────┘
```

**Command parameters:**

**LIST DCS APPLICATIONS**
The default application information includes:

- Host authorization ID (*username*)
- Application program name
- Application handle
- Outbound application ID (*luwid*).

**SHOW DETAIL**
Specifies that output include the following additional information:

- Client application ID
- Client sequence number
- Client database alias
- Client node name (*nname*)
- Client release level
- Client code page
- Outbound sequence number
- Host database name
- Host release level.

**EXTENDED**
Generates an extended report. This report includes all of the fields that are listed when the SHOW DETAIL option is specified, plus the following additional fields:

- DCS application status
- Status change time
- Client platform

- Client protocol
- Client code page
- Process ID of the client application
- Host coded character set ID (CCSID).

**Examples:**

The following is sample output from LIST DCS APPLICATIONS:

```
   Auth Id  Application Name     Appl.      Outbound Application Id
                                 Handle
   -------- -------------------- ---------- --------------------------------
   DDCSUS1  db2bp_s              2          0915155C.139D.971205184245
```

The following is sample output from LIST DCS APPLICATIONS EXTENDED:

```
               List of DCS Applications - Extended Report

Client application ID                     = 09151251.0AD1.980529194106
  Sequence number                         = 0001
  Authorization ID                        = SMITH
  Application name                        = db2bp
  Application handle                      = 0
  Application status                      = waiting for reply
  Status change time                      = Not Collected
  Client DB alias                         = MVSDB
  Client node                             = antman
  Client release level                    = SQL05020
  Client platform                         = AIX
  Client protocol                         = TCP/IP
  Client codepage                         = 819
  Process ID of client application        = 38340
  Client login ID                         = user1
  Host application ID                     = G9151251.GAD2.980529194108
  Sequence number                         = 0000
  Host DB name                            = GILROY
  Host release level                      = DSN05011
  Host CCSID                              = 500
```

**Notes:**

1. The application status field contains one of the following values:

   **connect pending - outbound**
   Denotes that the request to connect to a host database has been issued, and that DB2 Connect is waiting for the connection to be established.

   **waiting for request**
   Denotes that the connection to the host database has been established, and that DB2 Connect is waiting for an SQL statement from the client application.

   **waiting for reply**
   Denotes that the SQL statement has been sent to the host database.

2. The status change time is shown only if the System Monitor UOW switch was turned on during processing. Otherwise, `Not Collected` is shown.

**Usage notes:**

The database administrator can use this command to match client application connections *to* the gateway with corresponding host connections *from* the gateway.

The database administrator can also use agent ID information to force specified applications off a DB2 Connect server.

**Related reference:**

- "FORCE APPLICATION" on page 372

# LIST DCS DIRECTORY

Lists the contents of the Database Connection Services (DCS) directory.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──LIST DCS DIRECTORY─────────────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Examples:**
The following is sample output from LIST DCS DIRECTORY:

```
 Database Connection Services (DCS) Directory

 Number of entries in the directory = 1

DCS 1 entry:

 Local database name                = DB2
 Target database name               = DSN_DB_1
 Application requestor name         =
 DCS parameters                     =
 Comment                            = DB2/MVS Location name DSN_DB_1
 DCS directory release level        = 0x0100
```

These fields are identified as follows:

**Local database name**
Specifies the local alias of the target host database. This corresponds to the *database-name* parameter entered when the host database was cataloged in the DCS directory.

**Target database name**
Specifies the name of the host database that can be accessed. This corresponds to the *target-database-name* parameter entered when the host database was cataloged in the DCS directory.

**Application requester name**
Specifies the name of the program residing on the application requester or server.

**DCS parameters**
String that contains the connection and operating environment parameters to use with the application requester. Corresponds to the parameter string entered when the host database was cataloged. The string must be enclosed by double quotation marks, and the parameters must be separated by commas.

**Comment**
Describes the database entry.

**DCS directory release level**
Specifies the version number of the Distributed Database Connection Services program under which the database was created.

**Usage notes:**

The DCS directory is created the first time that the CATALOG DCS DATABASE command is invoked. It is maintained on the path/drive where DB2 was installed, and provides information about host databases that the workstation can access if the DB2 Connect program has been installed. The host databases can be:

- DB2 UDB databases on OS/390 and z/OS host
- DB2 UDB databases on iSeries hosts
- DB2 databases on VSE & VM hosts

**Related reference:**

- "CATALOG DCS DATABASE" on page 311

# LIST DRDA INDOUBT TRANSACTIONS

Provides a list of transactions that are indoubt between DRDA requesters and DRDA servers. If APPC commit protocols are being used, lists indoubt transactions between partner LUs. If DRDA commit protocols are being used, lists indoubt transactions between DRDA sync point managers.

**Authorization:**

*sysadm*

**Required connection:**

Instance

**Command syntax:**

```
►►──LIST DRDA INDOUBT TRANSACTIONS──┬────────────────┬──────────────────────►◄
                                    └─WITH PROMPTING─┘
```

**Command parameters:**

**WITH PROMPTING**

Indicates that indoubt transactions are to be processed. If this parameter is specified, an interactive dialog mode is initiated, permitting the user to commit or roll back indoubt transactions. If this parameter is not specified, indoubt transactions are written to the standard output device, and the interactive dialog mode is not initiated.

**Note:** A forget option is not supported. Once the indoubt transaction is committed or rolled back, the transaction is automatically forgotten.

Interactive dialog mode permits the user to:

- List all indoubt transactions (enter `l`)
- List indoubt transaction number *x* (enter `l`, followed by a valid transaction number)
- Quit (enter `q`)
- Commit transaction number *x* (enter `c`, followed by a valid transaction number)
- Roll back transaction number *x* (enter `r`, followed by a valid transaction number).

**Note:** A blank space must separate the command letter from its argument.

Before a transaction is committed or rolled back, the transaction data is displayed, and the user is asked to confirm the action.

**Usage notes:**

DRDA indoubt transactions occur when communication is lost between coordinators and participants in distributed units of work. A distributed unit of work lets a user or application read and update data at multiple locations within a single unit of work. Such work requires a two-phase commit.

The first phase requests all the participants to prepare for a commit. The second phase commits or rolls back the transactions. If a coordinator or participant becomes unavailable after the first phase, the distributed transactions are indoubt.

Before issuing the LIST DRDA INDOUBT TRANSACTIONS command, the application process must be connected to the DB2 sync point manager (SPM) instance. Use the *spm_name* database manager configuration parameter as the *dbalias* on the CONNECT statement.

TCP/IP connections, using the SPM to coordinate commits, use DRDA two-phase commit protocols. APPC connections use LU6.2 two-phase commit protocols.

# LIST HISTORY

Lists entries in the history file. The history file contains a record of recovery and administrative events. Recovery events include full database and table space level backup, incremental backup, restore, and rollforward operations. Additional logged events include create, alter, drop, or rename table space, reorganize table, drop table, and load.

**Authorization:**

None

**Required connection:**

Instance. You must attach to any remote database in order to run this command against it. For a local database, an explicit attachment is not required.

**Command syntax:**

```
►►──LIST HISTORY──┬───────────────────┬──────────────────────────────────────►
                  ├─BACKUP────────────┤
                  ├─ROLLFORWARD───────┤
                  ├─DROPPED TABLE─────┤
                  ├─LOAD──────────────┤
                  ├─CREATE TABLESPACE─┤
                  ├─ALTER TABLESPACE──┤
                  ├─RENAME TABLESPACE─┤
                  ├─REORG─────────────┤
                  └─ARCHIVE LOG───────┘

►──┬─ALL────────────────────────────────┬──FOR──┬──────────┬──database-alias──►◄
   ├─SINCE──timestamp───────────────────┤       ├─DATABASE─┤
   └─CONTAINING──┬─schema.object_name─┬─┘       └─DB───────┘
                 └─object_name────────┘
```

**Command parameters:**

**HISTORY**
: Lists all events that are currently logged in the history file.

**BACKUP**
: Lists backup and restore operations.

**ROLLFORWARD**
: Lists rollforward operations.

**DROPPED TABLE**
: Lists dropped table records. A dropped table record is created only when the table is dropped and the table space containing it has the DROPPED TABLE RECOVERY option enabled.

**LOAD**
: Lists load operations.

**CREATE TABLESPACE**
: Lists table space create and drop operations.

**RENAME TABLESPACE**
: Lists table space renaming operations.

**REORG**
Lists reorganization operations.

**ALTER TABLESPACE**
Lists alter table space operations.

**ARCHIVE LOG**
Lists archive log operations and the archived logs.

**ALL** Lists all entries of the specified type in the history file.

**SINCE timestamp**
A complete time stamp (format *yyyymmddhhmmss*), or an initial prefix (minimum *yyyy*) can be specified. All entries with time stamps equal to or greater than the time stamp provided are listed.

**CONTAINING schema.object_name**
This qualified name uniquely identifies a table.

**CONTAINING object_name**
This unqualified name uniquely identifies a table space.

**FOR DATABASE database-alias**
Used to identify the database whose recovery history file is to be listed.

**Examples:**

```
db2 list history since 19980201 for sample
db2 list history backup containing userspace1 for sample
db2 list history dropped table all for db sample
```

**Usage notes:**

The report generated by this command contains the following symbols:

```
Operation

   A - Create table space
   B - Backup
   C - Load copy
   D - Dropped table
   F - Roll forward
   G - Reorganize table
   L - Load
   N - Rename table space
   O - Drop table space
   Q - Quiesce
   R - Restore
   T - Alter table space
   U - Unload
   X - Archive log

Type

Archive Log types:
   P - Primary log path
   M - Secondary (mirror) log path
   F - Failover archive path
   1 - Primary log archive method
   2 - Secondary log archive method

Backup types:

   F - Offline
   N - Online
   I - Incremental offline
   O - Incremental online
   D - Delta offline
   E - Delta online

Rollforward types:

   E - End of logs
   P - Point in time

Load types:

   I - Insert
   R - Replace

Alter table space types:

   C - Add containers
   R - Rebalance

Quiesce types:

   S - Quiesce share
   U - Quiesce update
   X - Quiesce exclusive
   Z - Quiesce reset
```

# LIST INDOUBT TRANSACTIONS

Provides a list of transactions that are indoubt. The user can interactively commit, roll back, or forget the indoubt transactions.

The two-phase commit protocol comprises:

1. The PREPARE phase, in which the resource manager writes the log pages to disk, so that it can respond to either a COMMIT or a ROLLBACK primitive
2. The COMMIT (or ROLLBACK) phase, in which the transaction is actually committed or rolled back.

An indoubt transaction is one which has been prepared, but not yet committed or rolled back.

**Scope:**

This command returns a list of indoubt transactions on the executed node.

**Authorization:**

*dbadm*

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
►►──LIST INDOUBT TRANSACTIONS──┬────────────────┬──────────────────────────────►◄
                               └─WITH PROMPTING─┘
```

**Command parameters:**

**WITH PROMPTING**

Indicates that indoubt transactions are to be processed. If this parameter is specified, an interactive dialog mode is initiated, permitting the user to commit, roll back, or forget indoubt transactions. If this parameter is not specified, indoubt transactions are written to the standard output device, and the interactive dialog mode is not initiated.

Interactive dialog mode permits the user to:

- List all indoubt transactions (enter `l`)
- List indoubt transaction number *x* (enter `l`, followed by a valid transaction number)
- Quit (enter `q`)
- Commit transaction number *x* (enter `c`, followed by a valid transaction number)
- Roll back transaction number *x* (enter `r`, followed by a valid transaction number)
- Forget transaction number *x* (enter `f`, followed by a valid transaction number).

**Note:** A blank space must separate the command letter from its argument.

Before a transaction is committed, rolled back, or forgotten, the transaction data is displayed, and the user is asked to confirm the action.

**Examples:**

The following is sample dialog generated by LIST INDOUBT TRANSACTIONS:

```
In-doubt Transactions for Database SAMPLE

1.   originator: XA
     appl_id: *LOCAL.DB2.95051815165159       sequence_no: 0001  status: i
timestamp: 05-18-1997 16:51:59 auth_id: SMITH  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD

2.   originator: XA
     appl_id: *LOCAL.DATABASE.950407161043  sequence_no: 0002  status: i
timestamp: 04-07-1997 16:10:43  auth_id: JONES  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1
   .
   .
   .

Enter in-doubt transaction command or 'q' to quit.
e.g. 'c 1' heuristically commits transaction 1.
c/r/f/l/q: c 1

1.   originator: XA
     appl_id: *LOCAL.DB2.95051815165159       sequence_no: 0001  status: i
timestamp: 05-18-1997 16:51:59 auth_id: SMITH  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD

Do you want to heuristically commit this in-doubt transaction ? (y/n) y

DB20000I "COMMIT INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: c 5

DB20030E "5" is not a valid in-doubt transaction number.

c/r/f/l/q: l

In-doubt Transactions for Database SAMPLE

1.   originator: XA
     appl_id: *LOCAL.DB2.95051815165159       sequence_no: 0001  status: c
timestamp: 05-18-1997 16:51:59 auth_id: SMITH  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F93DD A92F8C4FF3000000
0000BD

2.   originator: XA
     appl_id: *LOCAL.DATABASE.950407161043  sequence_no: 0002  status: i
timestamp: 04-07-1997 16:10:43  auth_id: JONES  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1
   .
   .
   .
c/r/f/l/q: r 2

2.   originator: XA
     appl_id: *LOCAL.DATABASE.950407161043  sequence_no: 0002  status: i
timestamp: 04-07-1997 16:10:43  auth_id: JONES  log_full: n type: RM
```

```
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

Do you want to heuristically rollback this in-doubt transaction ? (y/n) y

DB20000I "ROLLBACK INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: l 2

2.   originator: XA
     appl_id: *LOCAL.DATABASE.950407161043  sequence_no: 0002  status: r
timestamp: 04-07-1997 16:10:43  auth_id: JONES  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

c/r/f/l/q: f 2

2.   originator: XA
     appl_id: *LOCAL.DATABASE.950407161043  sequence_no: 0002  status: r
timestamp: 04-07-1997 16:10:43  auth_id: JONES  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

Do you want to forget this in-doubt transaction ? (y/n) y

DB20000I "FORGET INDOUBT TRANSACTION" completed successfully

c/r/f/l/q: l 2

2.   originator: XA
     appl_id: *LOCAL.DATABASE.950407161043  sequence_no: 0002  status: f
timestamp: 04-07-1997 16:10:43  auth_id: JONES  log_full: n type: RM
xid: 53514C2000000017 00000000544D4442 00000000002F95FE B62F8C4FF3000000
0000C1

c/r/f/l/q: q
```

**Note:** The LIST INDOUBT TRANSACTIONS command returns *type* information to show the role of the database in each indoubt transaction:

**TM** Indicates the indoubt transaction is using the database as a transaction manager database.

**RM** Indicates the indoubt transaction is using the database as a resource manager, meaning that it is one of the databases participating in the transaction, but is not the transaction manager database.

**Usage notes:**

An indoubt transaction is a global transaction that was left in an indoubt state. This occurs when either the Transaction Manager (TM) or at least one Resource Manager (RM) becomes unavailable after successfully completing the first phase (that is, the PREPARE phase) of the two-phase commit protocol. The RMs do not know whether to commit or to roll back their branch of the transaction until the TM can consolidate its own log with the indoubt status information from the RMs when they again become available. An indoubt transaction can also exist in an MPP environment.

If LIST INDOUBT TRANSACTIONS is issued against the currently connected database, the command returns the information on indoubt transactions in that database.

Only transactions whose status is indoubt (i) or missing commit acknowledgment (m) can be committed.

Only transactions whose status is indoubt (i), missing rollback acknowledgment (b), or ended (e) can be rolled back.

Only transactions whose status is committed (c) or rolled back (r) can be forgotten.

**Note:** In the commit phase of a two-phase commit, the coordinator node waits for commit acknowledgments. If one or more nodes do not reply (for example, because of node failure), the transaction is placed in missing commit acknowledgment state.

Indoubt transaction information is valid only at the time that the command is issued. Once in interactive dialog mode, transaction status might change because of external activities. If this happens, and an attempt is made to process an indoubt transaction which is no longer in an appropriate state, an error message is displayed.

After this type of error occurs, the user should quit (q) the interactive dialog and reissue the LIST INDOUBT TRANSACTIONS WITH PROMPTING command to refresh the information shown.

**Related concepts:**

- "Configuration considerations for XA transaction managers" in the *Administration Guide: Planning*

# LIST NODE DIRECTORY

Lists the contents of the node directory.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──LIST──┬───────┬──NODE DIRECTORY──┬─────────────┬──────────────────────►◄
          └─ADMIN─┘                  └─SHOW DETAIL─┘
```

**Command parameters:**

**ADMIN**
Specifies administration server nodes.

**SHOW DETAIL**
Specifies that the output should include the following information:

- Remote instance name
- System
- Operating system type

**Examples:**

The following is sample output from LIST NODE DIRECTORY:

```
 Node Directory

 Number of entries in the directory = 2

Node 1 entry:

 Node name                      = LANNODE
 Comment                        =
 Directory entry type           = LDAP
 Protocol                       = TCPIP
 Hostname                       = LAN.db2ntd3.torolab.ibm.com
 Service name                   = 50000

Node 2 entry:

 Node name                      = TLBA10ME
 Comment                        =
 Directory entry type           = LOCAL
 Protocol                       = TCPIP
 Hostname                       = tlba10me
 Service name                   = 447
```

The following is sample output from LIST ADMIN NODE DIRECTORY:

```
 Node Directory

 Number of entries in the directory = 2
```

```
Node 1 entry:

 Node name                      = LOCALADM
 Comment                        =
 Directory entry type           = LOCAL
 Protocol                       = TCPIP
 Hostname                       = jaguar
 Service name                   = 523

Node 2 entry:

 Node name                      = MYDB2DAS
 Comment                        =
 Directory entry type           = LDAP
 Protocol                       = TCPIP
 Hostname                       = peng.torolab.ibm.com
 Service name                   = 523
```

The common fields are identified as follows:

**Node name**
The name of the remote node. This corresponds to the name entered for the *nodename* parameter when the node was cataloged.

**Comment**
A comment associated with the node, entered when the node was cataloged. To change a comment in the node directory, uncatalog the node, and then catalog it again with the new comment.

**Directory entry type**
LOCAL means the entry is found in the local node directory file. LDAP means the entry is found in LDAP server or LDAP cache.

**Protocol**
The communications protocol cataloged for the node.

**Note:** For information about fields associated with a specific node type, see the applicable CATALOG...NODE command.

**Usage notes:**

A node directory is created and maintained on each database client. It contains an entry for each remote workstation having databases that the client can access. The DB2 client uses the communication end point information in the node directory whenever a database connection or instance attachment is requested.

The database manager creates a node entry and adds it to the node directory each time it processes a CATALOG...NODE command. The entries can vary, depending on the communications protocol being used by the node.

The node directory can contain entries for the following types of nodes:

- APPC
- APPCLU
- APPN
- LDAP
- Local
- Named pipe
- NetBIOS

- TCP/IP.

**Related reference:**

- "CATALOG APPC NODE" on page 303
- "CATALOG TCPIP NODE" on page 324
- "CATALOG NETBIOS NODE" on page 321
- "CATALOG LOCAL NODE" on page 317
- "CATALOG APPN NODE" on page 305
- "CATALOG NAMED PIPE NODE" on page 319
- "CATALOG LDAP NODE" on page 316

# LIST ODBC DATA SOURCES

Lists all available user or system ODBC data sources.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database. That name is used to access the database or file system through ODBC APIs. On Windows, either user or system data sources can be cataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows only.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
             ┌─USER───┐
►►──LIST─────┼────────┼──ODBC DATA SOURCES─────────────────────────────────►◄
             └─SYSTEM─┘
```

**Command parameters:**

**USER** List only user ODBC data sources. This is the default if no keyword is specified.

**SYSTEM**
List only system ODBC data sources.

**Examples:**

The following is sample output from the LIST ODBC DATA SOURCES command:

```
               User ODBC Data Sources

Data source name                 Description
-------------------------------- ----------------------------------------
SAMPLE                           IBM DB2 ODBC DRIVER
```

**Related reference:**

- "CATALOG ODBC DATA SOURCE" on page 323
- "UNCATALOG ODBC DATA SOURCE" on page 712

# LIST PACKAGES/TABLES

Lists packages or tables associated with the current database.

**Authorization:**

For the system catalog SYSCAT.PACKAGES (LIST PACKAGES) and SYSCAT.TABLES (LIST TABLES), one of the following is required:

- *sysadm* or *dbadm* authority
- CONTROL privilege
- SELECT privilege.

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
►►──LIST──┬─PACKAGES─┬──┬──────────────────────────────┬──┬─────────────┬─────►◄
          └─TABLES───┘  │      ┌─USER────────────────┐ │  └─SHOW DETAIL─┘
                        └─FOR──┼─ALL─────────────────┼─┘
                               ├─SCHEMA──schema-name─┤
                               └─SYSTEM──────────────┘
```

**Command parameters:**

**FOR** If the FOR clause is not specified, the packages or tables for USER are listed.

**ALL** Lists all packages or tables in the database.

**SCHEMA**
Lists all packages or tables in the database for the specified schema only.

**SYSTEM**
Lists all system packages or tables in the database.

**USER** Lists all user packages or tables in the database for the current user.

**SHOW DETAIL**
If this option is chosen with the LIST TABLES command, the full table name and schema name are displayed. If this option is not specified, the table name is truncated to 30 characters, and the ″>″ symbol in the 31st column represents the truncated portion of the table name; the schema name is truncated to 14 characters and the ″>″ symbol in the 15th column represents the truncated portion of the schema name. If this option is chosen with the LIST PACKAGES command, the full package schema (creator), version and boundby authid are displayed, and the package unique_id (consistency token shown in hexadecimal form). If this option is not specified, the schema name and bound by ID are truncated to 8 characters and the ″>″ symbol in the 9th column represents the truncated portion of the schema or bound by ID; the version is truncated to 10 characters and the ″>″ symbol in the 11th column represents the truncated portion of the version.

**Examples:**

The following is sample output from LIST PACKAGES:

```
                                Bound     Total                      Isolation
Package    Schema    Version    by        sections     Valid  Format level     Blocking
---------- --------- ---------- --------- ------------ ------ ------- --------- --------
F4INS      USERA     VER1       SNOWBELL           221 Y      0       CS        U
F4INS      USERA     VER2.0     SNOWBELL           201 Y      0       RS        U
F4INS      USERA     VER2.3     SNOWBELL           201 N      3       CS        U
F4INS      USERA     VER2.5     SNOWBELL           201 Y      0       CS        U
PKG12      USERA                USERA               12 Y      3       RR        B
PKG15      USERA                USERA               42 Y      3       RR        B
SALARY     USERT     YEAR2000   USERT               15 Y      3       CS        N
```

The following is sample output from LIST TABLES:

```
Table/View         Schema           Type       Creation time
------------------ ---------------- ---------- ----------------------------
DEPARTMENT         SMITH            T          1997-02-19-13.32.25.971890
EMP_ACT            SMITH            T          1997-02-19-13.32.27.851115
EMP_PHOTO          SMITH            T          1997-02-19-13.32.29.953624
EMP_RESUME         SMITH            T          1997-02-19-13.32.37.837433
EMPLOYEE           SMITH            T          1997-02-19-13.32.26.348245
ORG                SMITH            T          1997-02-19-13.32.24.478021
PROJECT            SMITH            T          1997-02-19-13.32.29.300304
SALES              SMITH            T          1997-02-19-13.32.42.973739
STAFF              SMITH            T          1997-02-19-13.32.25.156337

  9 record(s) selected.
```

**Usage notes:**

LIST PACKAGES and LIST TABLES commands are available to provide a quick interface to the system tables.

The following SELECT statements return information found in the system tables. They can be expanded to select the additional information that the system tables provide.

```
   select tabname, tabschema, type, create_time
   from syscat.tables
   order by tabschema, tabname;

   select pkgname, pkgschema, pkgversion, unique_id, boundby, total_sect,
      valid, format, isolation, blocking
   from syscat.packages
   order by pkgschema, pkgname, pkgversion;

   select tabname, tabschema, type, create_time
   from syscat.tables
   where tabschema = 'SYSCAT'
   order by tabschema, tabname;

   select pkgname, pkgschema, pkgversion, unique_id, boundby, total_sect,
      valid, format, isolation, blocking
   from syscat.packages
   where pkgschema = 'NULLID'
   order by pkgschema, pkgname, pkgversion;

   select tabname, tabschema, type, create_time
   from syscat.tables
   where tabschema = USER
   order by tabschema, tabname;
```

```
select pkgname, pkgschema, pkgversion, unique_id, boundby, total_sect,
   valid, format, isolation, blocking
from syscat.packages
where pkgschema = USER
order by pkgschema, pkgname, pkgversion;
```

**Related concepts:**

- "Catalog views" in the *SQL Reference, Volume 1*
- "Efficient SELECT statements" in the *Administration Guide: Performance*

# LIST TABLESPACE CONTAINERS

Lists containers for the specified table space.

**Note:** The table space snapshot contains all of the information displayed by the LIST TABLESPACE CONTAINERS command.

**Scope:**

This command returns information only for the node on which it is executed.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

Database

**Command syntax:**

```
►►──LIST TABLESPACE CONTAINERS FOR──tablespace-id──┬─────────────┬──────────────►◄
                                                   └─SHOW DETAIL─┘
```

**Command parameters:**

**FOR tablespace-id**
An integer that uniquely represents a table space used by the current database. To get a list of all the table spaces used by the current database, use the LIST TABLESPACES command.

**SHOW DETAIL**
If this option is not specified, only the following basic information about each container is provided:

- Container ID
- Name
- Type (file, disk, or path).

If this option is specified, the following additional information about each container is provided:

- Total number of pages
- Number of useable pages
- Accessible (yes or no).

**Examples:**

The following is sample output from LIST TABLESPACE CONTAINERS:

```
          Tablespace Containers for Tablespace 0

 Container ID                         = 0
```

```
Name                                 = /home/smith/smith/NODE0000/
                                       SQL00001/SQLT0000.0
Type                                 = Path
```

The following is sample output from LIST TABLESPACE CONTAINERS with SHOW DETAIL specified:

```
           Tablespace Containers for Tablespace 0

Container ID                         = 0
Name                                 = /home/smith/smith/NODE0000/
                                       SQL00001/SQLT0000.0
Type                                 = Path
Total pages                          = 895
Useable pages                        = 895
Accessible                           = Yes
```

**Related concepts:**

- "Snapshot monitor" in the *System Monitor Guide and Reference*

**Related reference:**

- "LIST TABLESPACES" on page 513

# LIST TABLESPACES

Lists table spaces for the current database.

**Note:** Information displayed by this command is also available in the table space snapshot.

**Scope:**

This command returns information only for the node on which it is executed.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*
- *load*

**Required connection:**

Database

**Command syntax:**

```
►►──LIST TABLESPACES──┬─────────────┬────────────────────────────────────►◄
                      └─SHOW DETAIL─┘
```

**Command parameters:**

**SHOW DETAIL**

If this option is not specified, only the following basic information about each table space is provided:

- Table space ID
- Name
- Type (system managed space or database managed space)
- Contents (any data, long or index data, or temporary data)
- State, a hexadecimal value indicating the current table space state. The externally visible state of a table space is composed of the hexadecimal sum of certain state values. For example, if the state is ″quiesced: EXCLUSIVE″ and ″Load pending″, the value is 0x0004 + 0x0008, which is 0x000c. The db2tbst (Get Tablespace State) command can be used to obtain the table space state associated with a given hexadecimal value. Following are the bit definitions listed in sqlutil.h:

```
0x0          Normal
0x1          Quiesced: SHARE
0x2          Quiesced: UPDATE
0x4          Quiesced: EXCLUSIVE
0x8          Load pending
0x10         Delete pending
0x20         Backup pending
0x40         Roll forward in progress
0x80         Roll forward pending
0x100        Restore pending
```

```
0x100       Recovery pending (not used)
0x200       Disable pending
0x400       Reorg in progress
0x800       Backup in progress
0x1000      Storage must be defined
0x2000      Restore in progress
0x4000      Offline and not accessible
0x8000      Drop pending
0x2000000   Storage may be defined
0x4000000   StorDef is in 'final' state
0x8000000   StorDef was changed prior to rollforward
0x10000000  DMS rebalancer is active
0x20000000  TBS deletion in progress
0x40000000  TBS creation in progress
0x8         For service use only
```

If this option is specified, the following additional information about each table space is provided:

- Total number of pages
- Number of usable pages
- Number of used pages
- Number of free pages
- High water mark (in pages)
- Page size (in bytes)
- Extent size (in pages)
- Prefetch size (in pages)
- Number of containers
- Minimum recovery time (displayed only if not zero)
- State change table space ID (displayed only if the table space state is ″load pending″ or ″delete pending″)
- State change object ID (displayed only if the table space state is ″load pending″ or ″delete pending″)
- Number of quiescers (displayed only if the table space state is ″quiesced: SHARE″, ″quiesced: UPDATE″, or ″quiesced: EXCLUSIVE″)
- Table space ID and object ID for each quiescer (displayed only if the number of quiescers is greater than zero).

**Examples:**

The following are two sample outputs from LIST TABLESPACES SHOW DETAIL.

```
          Tablespaces for Current Database
 Tablespace ID                        = 0
 Name                                 = SYSCATSPACE
 Type                                 = System managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 895
 Useable pages                        = 895
 Used pages                           = 895
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 4096
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
 Tablespace ID                        = 1
 Name                                 = TEMPSPACE1
 Type                                 = System managed space
 Contents                             = Temporary data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 1
 Useable pages                        = 1
 Used pages                           = 1
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 4096
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
 Tablespace ID                        = 2
 Name                                 = USERSPACE1
 Type                                 = System managed space
 Contents                             = Any data
 State                                = 0x000c
   Detailed explanation:
     Quiesced: EXCLUSIVE
     Load pending
 Total pages                          = 337
 Useable pages                        = 337
 Used pages                           = 337
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 4096
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
 State change tablespace ID           = 2
 State change object ID               = 3
 Number of quiescers                  = 1
   Quiescer 1:
     Tablespace ID                    = 2
     Object ID                        = 3
DB21011I  In a partitioned database server environment, only the table spaces
on the current node are listed.


           Tablespaces for Current Database
 Tablespace ID                        = 0
 Name                                 = SYSCATSPACE
 Type                                 = System managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 1200
 Useable pages                        = 1200
 Used pages                           = 1200
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 4096
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
 Tablespace ID                        = 1
 Name                                 = TEMPSPACE1
 Type                                 = System managed space
 Contents                             = Temporary data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 1
 Useable pages                        = 1
 Used pages                           = 1
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 4096
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
Tablespace ID                         = 2
 Name                                 = USERSPACE1
 Type                                 = System managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 1
 Useable pages                        = 1
 Used pages                           = 1
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 4096
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
Tablespace ID                         = 3
 Name                                 = DMS8K
 Type                                 = Database managed space
 Contents                             = Any data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 2000
 Useable pages                        = 1952
 Used pages                           = 96
 Free pages                           = 1856
 High water mark (pages)              = 96
 Page size (bytes)                    = 8192
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 2
Tablespace ID                         = 4
 Name                                 = TEMP8K
 Type                                 = System managed space
 Contents                             = Temporary data
 State                                = 0x0000
   Detailed explanation:
     Normal
 Total pages                          = 1
 Useable pages                        = 1
 Used pages                           = 1
 Free pages                           = Not applicable
 High water mark (pages)              = Not applicable
 Page size (bytes)                    = 8192
 Extent size (pages)                  = 32
 Prefetch size (pages)                = 32
 Number of containers                 = 1
DB21011I  In a partitioned database server environment, only the table spaces
on the current node are listed.
```

**Usage notes:**

In a partitioned database environment, this command does not return all the table spaces in the database. To obtain a list of all the table spaces, query SYSCAT.SYSTABLESPACES.

During a table space rebalance, the number of usable pages includes pages for the newly added container, but these new pages are not reflected in the number of free pages until the rebalance is complete. When a table space rebalance is not in progress, the number of used pages plus the number of free pages equals the number of usable pages.

**Related reference:**

- "LIST TABLESPACE CONTAINERS" on page 511
- "db2tbst - Get Tablespace State" on page 229

# LIST UTILITIES

Displays to standard output the list of active utilities on the instance. The description of each utility can include attributes such as start time, description, throttling priority (if applicable), as well as progress monitoring information (if applicable).

**Scope:**

This command only returns information for the database partition on which it is issued.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance.

**Command syntax:**

```
►►──LIST UTILITIES──┬─────────────┬────────────────────────────────────►◄
                    └─SHOW DETAIL─┘
```

**Command parameters:**

**SHOW DETAIL**
Displays detailed progress information for utilities that support progress monitoring.

**Examples:**

A RUNSTATS invocation on table some_table:

```
LIST UTILITIES

ID                           = 1
Type                         = RUNSTATS
Database Name                = PROD
Description                  = krrose.some_table
Start Time                   = 12/19/2003 11:54:45.773215
Priority                     = 10
```

Monitoring the performance of an offline database backup:

```
LIST UTILITIES SHOW DETAIL

ID                           = 2
Type                         = BACKUP
Database Name                = SAMPLE
Description                  = offline db
Start Time                   = 10/30/2003 12:55:31.786115
Priority                     = 0
Progress Monitoring:
   Phase Number [CURRENT]    = 1
      Description            =
      Work Metric          = BYTES
      Total Work Units     = 20232453
      Completed Work Units = 230637
      Start Time           = 10/30/2003 12:55:31.786115
```

**Usage notes:**

Use this command to monitor the status of running utilities. For example, you might use this utility to monitor the progress of an online backup. In another example, you might investigate a performance problem by using this command to determine which utilities are running. If the utility is suspected to be responsible for degrading performance then you might elect to throttle the utility (if the utility supports throttling). Note that the ID from the LIST UTILITIES command is the same ID used in the SET UTIL_IMPACT_PRIORITY command.

**Related reference:**

- "SET UTIL_IMPACT_PRIORITY" on page 686

# LOAD

Loads data into a DB2 table. Data residing on the server can be in the form of a file, tape, or named pipe. Data residing on a remotely connected client can be in the form of a fully qualified file or named pipe. Data can also be loaded from a user-defined cursor.

**Restrictions:**

The load utility does not support loading data at the hierarchy level. The load utility is not compatible with range-clustered tables.

**Scope:**

This command can be issued against multiple database partitions in a single request.

**Authorization:**

One of the following:

- *sysadm*
- *dbadm*
- load authority on the database and
  - INSERT privilege on the table when the load utility is invoked in INSERT mode, TERMINATE mode (to terminate a previous load insert operation), or RESTART mode (to restart a previous load insert operation)
  - INSERT and DELETE privilege on the table when the load utility is invoked in REPLACE mode, TERMINATE mode (to terminate a previous load replace operation), or RESTART mode (to restart a previous load replace operation)
  - INSERT privilege on the exception table, if such a table is used as part of the load operation.

Since all load processes (and all DB2 server processes, in general) are owned by the instance owner, and all of these processes use the identification of the instance owner to access needed files, the instance owner must have read access to input data files. These input data files must be readable by the instance owner, regardless of who invokes the command.

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

Instance. An explicit attachment is not required. If a connection to the database has been established, an implicit attachment to the local instance is attempted.

**Command syntax:**

```
                          ┌─,──────────┐
►►──LOAD──┬────────┬──FROM──▼─┬─filename───┬─┴──OF──filetype──┬──────────────────────────┬──────────►
          └─CLIENT─┘          ├─pipename───┤                  │            ┌─,────────┐  │
                              ├─device─────┤                  │            ▼          │  │
                              └─cursorname─┘                  └─LOBS FROM────lob-path─┴──┘
```

```
>--+------------------------------+------------------------------------------------------->
   |                .----------.  |
   |                V          |  |
   '-MODIFIED BY----filetype-mod-+'

>--+----------------------------------------------------------------------------------+--->
   |               .-,-----------------------.                                        |
   |               V                         |                                        |
   '-METHOD--+-L--(----column-start--column-end-+--)--+---------------------------------------------+-+-'
             |                                        |                      .-,-------------------. | |
             |                                        |                      V                     | | |
             |                                        '-NULL INDICATORS--(----null-indicator-list-+--)-' |
             |       .-,-----------.                                                                    |
             |       V             |                                                                    |
             +-N--(----column-name-+--)-----------------------------------------------------------------+
             |       .-,---------------.                                                                |
             |       V                 |                                                                |
             '-P--(----column-position-+--)-------------------------------------------------------------'

>--+-------------------+--+------------------+--+----------------------+--+-------------------------+-->
   '-SAVECOUNT--n------'  '-ROWCOUNT--n------'  '-WARNINGCOUNT--n------'  '-MESSAGES--message-file--'

>--+---------------------------------+--+-INSERT----+--INTO--table-name--+--------------------------+-->
   '-TEMPFILES PATH--temp-pathname--'   +-REPLACE---+                    |      .-,-------------.   |
                                        +-RESTART---+                    |      V               |   |
                                        '-TERMINATE-'                    '-(----insert-column---+-)-'

>--+------------------------------------------+--+-------------------------------+--+--------------------------------+-->
   '-DATALINK SPECIFICATION--| datalink-spec |-'  '-FOR EXCEPTION--table-name---'  '-STATISTICS--+-USE PROFILE-+--'
                                                                                                '-NO----------'

>--+-----------------------------------------------------------------------+--+---------------------+-->
   |        .-NO----------------------------------------------------.      |  '-WITHOUT PROMPTING---'
   +-COPY---+-YES--+-USE TSM--+------------------------------+------+-+----+
   |               |          '-OPEN--num-sess--SESSIONS-----'        |    |
   |               |      .-,----------------.                        |    |
   |               |      V                  |                        |    |
   |               +-TO------device/directory-+-----------------------+    |
   |               '-LOAD--lib-name--+------------------------------+-'    |
   |                                 '-OPEN--num-sess--SESSIONS-----'      |
   '-NONRECOVERABLE--------------------------------------------------------'

>--+--------------------------+--+--------------------------+--+----------------------+--+-----------------------+-->
   '-DATA BUFFER--buffer-size-'  '-SORT BUFFER--buffer-size-'  '-CPU_PARALLELISM--n---'  '-DISK_PARALLELISM--n---'

                                           .-ALLOW NO ACCESS--------------------------------------.
>--+--------------------------------+------+------------------------------------------------------+-->
   '-INDEXING MODE--+-AUTOSELECT--+-'      '-ALLOW READ ACCESS--+-----------------------------+----'
                    +-REBUILD-----+                             '-USE--tablespace-name--------'
                    +-INCREMENTAL-+
                    '-DEFERRED----'

>--+----------------------------------------+--+-------------------+-->
   '-CHECK PENDING CASCADE--+-IMMEDIATE-+---'  '-LOCK WITH FORCE---'
                            '-DEFERRED--'

>--+------------------------------------------------------------+----------------------------><
   |                           .-------------------------.       |
   |                           V                         |       |
   '-+-----------------------+----partitioned-db-option--+-------'
     '-PARTITIONED DB CONFIG-'
```

### datalink-spec:

```
    .-,----------------------------------------------------------------------------------------------.
    V                                                                                                 |
|------(--+------------------+--+-----------------------------------+--+----------------------------+--)-+-------|
          '-DL_LINKTYPE URL--'  +-DL_URL_REPLACE_PREFIX--"prefix"---+  '-DL_URL_SUFFIX--"suffix"----'
                                '-DL_URL_DEFAULT_PREFIX--"prefix"---'
```

## Command parameters:

**ALLOW NO ACCESS**

Load will lock the target table for exclusive access during the load. The table state will be set to LOAD IN PROGRESS during the load. ALLOW NO ACCESS is the default behavior. It is the only valid option for LOAD REPLACE.

When there are constraints on the table, the table state will be set to CHECK PENDING as well as LOAD IN PROGRESS. The SET INTEGRITY statement must be used to take the table out of CHECK PENDING.

**ALLOW READ ACCESS**

Load will lock the target table in a share mode. The table state will be set to both LOAD IN PROGRESS and READ ACCESS. Readers can access the non-delta portion of the data while the table is being load. In other words, data that existed before the start of the load will be accessible by readers to the table, data that is being loaded is not available until the load is complete. LOAD TERMINATE or LOAD RESTART of an ALLOW READ ACCESS load can use this option; LOAD TERMINATE or LOAD RESTART of an ALLOW NO ACCESS load cannot use this option. Furthermore, this option is not valid if the indexes on the target table are marked as requiring a rebuild.

When there are constraints on the table, the table state will be set to CHECK PENDING as well as LOAD IN PROGRESS, and READ ACCESS. At the end of the load the table state LOAD IN PROGRESS state will be removed but the table states CHECK PENDING and READ ACCESS will remain. The SET INTEGRITY statement must be used to take the table out of CHECK PENDING. While the table is in CHECK PENDING and READ ACCESS, the non-delta portion of the data is still accessible to readers, the new (delta) portion of the data will remain inaccessible until the SET INTEGRITY statement has completed. A user can perform multiple loads on the same table without issuing a SET INTEGRITY statement. Only the original (checked) data will remain visible, however, until the SET INTEGRITY statement is issued.

ALLOW READ ACCESS also supports the following modifiers:

**USE tablespace-name**

If the indexes are being rebuilt, a shadow copy of the index is built in table space *tablespace-name* and copied over to the original table space at the end of the load during an INDEX COPY PHASE. Only system temporary table spaces can be used with this option. If not specified then the shadow index will be created in the same table space as the index object. If the shadow copy is created in the same table space as the index object, the copy of the shadow index object over the old index object is instantaneous. If the shadow copy is in a different table space from the index object a physical copy is performed. This could involve considerable I/O and time. The copy happens while the table is offline at the end of a load during the INDEX COPY PHASE.

Without this option the shadow index is built in the same table space as the original. Since both the original index and shadow index by default reside in the same table space simultaneously, there might be insufficient space to hold both indexes within one table space. Using this option ensures that you retain enough table space for the indexes.

This option is ignored if the user does not specify INDEXING MODE REBUILD or INDEXING MODE AUTOSELECT. This option will also be ignored if INDEXING MODE AUTOSELECT is chosen and load chooses to incrementally update the index.

**CHECK PENDING CASCADE**

If LOAD puts the table into a check pending state, the CHECK PENDING CASCADE option allows the user to specify whether or not the check pending state of the loaded table is immediately cascaded to all descendents (including descendent foreign key tables, descendent immediate materialized query tables and descendent immediate staging tables).

**IMMEDIATE**

Indicates that the check pending state (read or no access mode) for foreign key constraints is immediately extended to all descendent foreign key tables. If the table has descendent immediate materialized query tables or descendent immediate staging tables, the check pending state is extended immediately to the materialized query tables and the staging tables. Note that for a LOAD INSERT operation, the check pending state is not extended to descendent foreign key tables even if the IMMEDIATE option is specified.

When the loaded table is later checked for constraint violations (using the IMMEDIATE CHECKED option of the SET INTEGRITY statement), descendent foreign key tables that were placed in check pending read state will be put into check pending no access state.

**DEFERRED**

Indicates that only the loaded table will be placed in the check pending state (read or no access mode). The states of the descendent foreign key tables, descendent immediate materialized query tables and descendent immediate staging tables will remain unchanged.

Descendent foreign key tables might later be implicitly placed in the check pending no access state when their parent tables are checked for constraint violations (using the IMMEDIATE CHECKED option of the SET INTEGRITY statement). Descendent immediate materialized query tables and descendent immediate staging tables will be implicitly placed in the check pending no access state when one of its underlying tables is checked for integrity violations. A warning (SQLSTATE 01586) will be issued to indicate that dependent tables have been placed in the check pending state. See the Notes section of the SET INTEGRITY statement in the SQL Reference for when these descendent tables will be put into the check pending state.

If the CHECK PENDING CASCADE option is not specified:

- Only the loaded table will be placed in the check pending state. The state of descendent foreign key tables, descendent immediate materialized query tables and descendent immediate staging tables will remain unchanged, and can later be implicitly put into the check pending state when the loaded table is checked for constraint violations.

If LOAD does not put the target table into check pending state, the CHECK PENDING CASCADE option is ignored.

**CLIENT**

Specifies that the data to be loaded resides on a remotely connected client. This option is ignored if the load operation is not being invoked from a remote client. This option is not supported in conjunction with the CURSOR filetype.

**Notes:**

1. The `dumpfile` and `lobsinfile` modifiers refer to files on the server even when the CLIENT keyword is specified.
2. Code page conversion is not performed during a remote load operation. If the code page of the data is different from that of the server, the data code page should be specified using the `codepage` modifier.

In the following example, a data file (`/u/user/data.del`) residing on a remotely connected client is to be loaded into MYTABLE on the server database:

```
db2 load client from /u/user/data.del of del
   modified by codepage=850 insert into mytable
```

**COPY NO**

Specifies that the table space in which the table resides will be placed in backup pending state if forward recovery is enabled (that is, *logretain* or *userexit* is on). The COPY NO option will also put the table space state into the Load in Progress table space state. This is a transient state that will disappear when the load completes or aborts. The data in any table in the table space cannot be updated or deleted until a table space backup or a full database backup is made. However, it is possible to access the data in any table by using the SELECT statement.

LOAD with COPY NO on a recoverable database leaves the table spaces in a backup pending state. For example, performing a LOAD with COPY NO and INDEXING MODE DEFERRED will leave indexes needing a refresh. Certain queries on the table might require an index scan and will not succeed until the indexes are refreshed. The index cannot be refreshed if it resides in a table space which is in the backup pending state. In that case, access to the table will not be allowed until a backup is taken.

**Note:** Index refresh is done automatically by the database when the index is accessed by a query.

**COPY YES**

Specifies that a copy of the loaded data will be saved. This option is invalid if forward recovery is disabled (both *logretain* and *userexit* are off). The option is not supported for tables with DATALINK columns.

**USE TSM**

Specifies that the copy will be stored using Tivoli Storage Manager (TSM).

**OPEN num-sess SESSIONS**

The number of I/O sessions to be used with TSM or the vendor product. The default value is 1.

**TO device/directory**

Specifies the device or directory on which the copy image will be created.

**LOAD lib-name**

The name of the shared library (DLL on Windows operating systems) containing the vendor backup and restore I/O functions to be used. It can contain the full path. If the full path is not given, it will default to the path where the user exit programs reside.

**CPU_PARALLELISM n**

Specifies the number of processes or threads that the load utility will spawn for parsing, converting, and formatting records when building table objects. This parameter is designed to exploit intra-partition parallelism. It is particularly useful when loading presorted data, because record order in the source data is preserved. If the value of this parameter is zero, or has not been specified, the load utility uses an intelligent default value (usually based on the number of CPUs available) at run time.

**Notes:**

1. If this parameter is used with tables containing either LOB or LONG VARCHAR fields, its value becomes one, regardless of the number of system CPUs or the value specified by the user.
2. Specifying a small value for the SAVECOUNT parameter causes the loader to perform many more I/O operations to flush both data and table metadata. When CPU_PARALLELISM is greater than one, the flushing operations are asynchronous, permitting the loader to exploit the CPU. When CPU_PARALLELISM is set to one, the loader waits on I/O during consistency points. A load operation with CPU_PARALLELISM set to two, and SAVECOUNT set to 10 000, completes faster than the same operation with CPU_PARALLELISM set to one, even though there is only one CPU.

**DATA BUFFER buffer-size**

Specifies the number of 4KB pages (regardless of the degree of parallelism) to use as buffered space for transferring data within the utility. If the value specified is less than the algorithmic minimum, the minimum required resource is used, and no warning is returned.

This memory is allocated directly from the utility heap, whose size can be modified through the *util_heap_sz* database configuration parameter.

If a value is not specified, an intelligent default is calculated by the utility at run time. The default is based on a percentage of the free space available in the utility heap at the instantiation time of the loader, as well as some characteristics of the table.

**DATALINK SPECIFICATION**

For each DATALINK column, there can be one column specification enclosed by parentheses. Each column specification consists of one or more DL_LINKTYPE, prefix, and a DL_URL_SUFFIX specification. The prefix specification can be either DL_URL_REPLACE_PREFIX or DL_URL_DEFAULT_PREFIX.

There can be as many DATALINK column specifications as the number of DATALINK columns defined in the table. The order of specifications follows the order of DATALINK columns found within the *insert-column* list, or within the table definition (if an *insert-column* list is not specified).

**DISK_PARALLELISM n**

Specifies the number of processes or threads that the load utility will spawn for writing data to the table space containers. If a value is not specified, the utility selects an intelligent default based on the number of table space containers and the characteristics of the table.

**DL_LINKTYPE**

If specified, it should match the LINKTYPE of the column definition. Thus, DL_LINKTYPE URL is acceptable if the column definition specifies LINKTYPE URL.

**DL_URL_DEFAULT_PREFIX ″prefix″**

If specified, it should act as the default prefix for all DATALINK values within the same column. In this context, prefix refers to the ″scheme host port″ part of the URL specification.

Examples of prefix are:

```
"http://server"
"file://server"
"file:"
"http://server:80"
```

If no prefix is found in the column data, and a default prefix is specified with DL_URL_DEFAULT_PREFIX, the default prefix is prefixed to the column value (if not NULL).

For example, if DL_URL_DEFAULT_PREFIX specifies the default prefix `"http://toronto"`:

- The column input value ″/x/y/z″ is stored as ″http://toronto/x/y/z″.
- The column input value ″http://coyote/a/b/c″ is stored as ″http://coyote/a/b/c″.
- The column input value NULL is stored as NULL.

**DL_URL_REPLACE_PREFIX ″prefix″**

This clause is useful when loading or importing data previously generated by the export utility, if the user wants to globally replace the host name in the data with another host name. If specified, it becomes the prefix for *all* non-NULL column values. If a column value has a prefix, this will replace it. If a column value has no prefix, the prefix specified by DL_URL_REPLACE_PREFIX is prefixed to the column value.

For example, if DL_URL_REPLACE_PREFIX specifies the prefix `"http://toronto"`:

- The column input value ″/x/y/z″ is stored as ″http://toronto/x/y/z″.
- The column input value ″http://coyote/a/b/c″ is stored as ″http://toronto/a/b/c″. Note that ″toronto″ replaces ″coyote″.
- The column input value NULL is stored as NULL.

**DL_URL_SUFFIX ″suffix″**

If specified, it is appended to every non-NULL column value for the column. It is, in fact, appended to the ″path″ component of the data location part of the DATALINK value.

**FOR EXCEPTION table-name**

Specifies the exception table into which rows in error will be copied. Any row that is in violation of a unique index or a primary key index is copied. DATALINK exceptions are also captured in the exception table. If an unqualified table name is specified, the table will be qualified with the CURRENT SCHEMA.

Information that is written to the exception table is *not* written to the dump file. In a partitioned database environment, an exception table must be defined for those partitions on which the loading table is defined. The dump file, on the other hand, contains rows that cannot be loaded because they are invalid or have syntax errors.

**FROM filename/pipename/device/cursorname**

Specifies the file, pipe, device, or cursor referring to an SQL statement that contains the data being loaded. If the input source is a file, pipe, or device, it must reside on the database partition where the database resides, unless the CLIENT option is specified. If several names are specified, they will be processed in sequence. If the last item specified is a tape device, the user is prompted for another tape. Valid response options are:

**c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted).

**d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes).

**t** Terminate. Terminate all devices.

**Notes:**

1. It is recommended that the fully qualified file name be used. If the server is remote, the fully qualified file name must be used. If the database resides on the same database partition as the caller, relative paths can be used.
2. Loading data from multiple IXF files is supported if the files are physically separate, but logically one file. It is *not* supported if the files are both logically and physically separate. (Multiple physical files would be considered logically one if they were all created with one invocation of the EXPORT command.)
3. If loading data that resides on a client machine, the data must be in the form of either a fully qualified file or a named pipe.

**INDEXING MODE**

Specifies whether the load utility is to rebuild indexes or to extend them incrementally. Valid values are:

**AUTOSELECT**

The load utility will automatically decide between REBUILD or INCREMENTAL mode.

**REBUILD**

All indexes will be rebuilt. The utility must have sufficient resources to sort all index key parts for both old and appended table data.

**INCREMENTAL**

Indexes will be extended with new data. This approach consumes index free space. It only requires enough sort space to append index keys for the inserted records. This method is only supported in cases where the index object is valid and accessible at the start of a load operation (it is, for example, not valid immediately following a load operation in which the DEFERRED mode was specified). If this mode is specified, but not supported due to the state of the index, a warning is returned, and the load operation continues in REBUILD mode. Similarly, if a load restart operation is begun in the load build phase, INCREMENTAL mode is not supported.

Incremental indexing is not supported when all of the following conditions are true:

- The LOAD COPY option is specified (*logretain* or *userexit* is enabled).
- The table resides in a DMS table space.
- The index object resides in a table space that is shared by other table objects belonging to the table being loaded.

To bypass this restriction, it is recommended that indexes be placed in a separate table space.

**DEFERRED**

The load utility will not attempt index creation if this mode is specified. Indexes will be marked as needing a refresh. The first access to such indexes that is unrelated to a load operation might force a rebuild, or indexes might be rebuilt when the database is restarted. This approach requires enough sort space for all key parts for the largest index. The total time subsequently taken for index construction is longer than that required in REBUILD mode. Therefore, when performing multiple load operations with deferred indexing, it is advisable (from a performance viewpoint) to let the last load operation in the sequence perform an index rebuild, rather than allow indexes to be rebuilt at first non-load access.

Deferred indexing is only supported for tables with non-unique indexes, so that duplicate keys inserted during the load phase are not persistent after the load operation.

**Note:** Deferred indexing is not supported for tables that have DATALINK columns.

**INSERT**

One of four modes under which the load utility can execute. Adds the loaded data to the table without changing the existing table data.

**insert-column**

Specifies the table column into which the data is to be inserted.

The load utility cannot parse columns whose names contain one or more spaces. For example,

```
db2 load from delfile1 of del modified by noeofchar noheader
   method P (1, 2, 3, 4, 5, 6, 7, 8, 9)
   insert into table1 (BLOB1, S2, I3, Int 4, I5, I6, DT7, I8, TM9)
```

will fail because of the `Int 4` column. The solution is to enclose such column names with double quotation marks:

```
db2 load from delfile1 of del modified by noeofchar noheader
   method P (1, 2, 3, 4, 5, 6, 7, 8, 9)
   insert into table1 (BLOB1, S2, I3, "Int 4", I5, I6, DT7, I8, TM9)
```

**INTO table-name**

Specifies the database table into which the data is to be loaded. This table cannot be a system table or a declared temporary table. An alias, or the fully qualified or unqualified table name can be specified. A qualified table name is in the form *schema.tablename*. If an unqualified table name is specified, the table will be qualified with the CURRENT SCHEMA.

**LOBS FROM lob-path**

The path to the data files containing LOB values to be loaded. The path must end with a slash (/). If the CLIENT option is specified, the path must be fully qualified. The names of the LOB data files are stored in the main data file (ASC, DEL, or IXF), in the column that will be loaded into the LOB column. This option is ignored if `lobsinfile` is not specified within the *filetype-mod* string.

This option is not supported in conjunction with the CURSOR filetype.

**LOCK WITH FORCE**

The utility acquires various locks including table locks in the process of loading. Rather than wait, and possibly timeout, when acquiring a lock, this option allows load to force off other applications that hold conflicting locks on the target table. Applications holding conflicting locks on the system catalog tables will not be forced off by the load utility. Forced applications will roll back and release the locks the load utility needs. The load utility can then proceed. This option requires the same authority as the FORCE APPLICATIONS command (SYSADM or SYSCTRL).

ALLOW NO ACCESS loads might force applications holding conflicting locks at the start of the load operation. At the start of the load the utility can force applications that are attempting to either query or modify the table.

ALLOW READ ACCESS loads can force applications holding conflicting locks at the start or end of the load operation. At the start of the load the load utility can force applications that are attempting to modify the table. At the end of the load operation, the load utility can force applications that are attempting to either query or modify the table.

**MESSAGES message-file**

Specifies the destination for warning and error messages that occur during the load operation. If a message file is not specified, messages are written to standard output. If the complete path to the file is not specified, the load utility uses the current directory and the default drive as the destination. If the name of a file that already exists is specified, the utility appends the information.

The message file is usually populated with messages at the end of the load operation and, as such, is not suitable for monitoring the progress of the operation.

**METHOD**

**L** Specifies the start and end column numbers from which to load data. A column number is a byte offset from the beginning of a row of data. It is numbered starting from 1.

**Note:** This method can only be used with ASC files, and is the only valid method for that file type.

**N** Specifies the names of the columns in the data file to be loaded. The case of these column names must match the case of the corresponding names in the system catalogs. Each table column that is not nullable should have a corresponding entry in the METHOD N list. For example, given data fields F1, F2, F3, F4, F5, and F6, and table columns C1 INT, C2 INT NOT NULL, C3 INT NOT NULL, and C4 INT, `method N (F2, F1, F4, F3)` is a valid request, while `method N (F2, F1)` is not valid.

**Note:** This method can only be used with file types IXF or CURSOR.

**P** Specifies the field numbers (numbered from 1) of the input data fields to be loaded. Each table column that is not nullable should have a corresponding entry in the METHOD P list. For example, given data fields F1, F2, F3, F4, F5, and F6, and table columns C1 INT, C2 INT NOT NULL, C3 INT NOT NULL, and C4 INT, `method P (2, 1, 4, 3)` is a valid request, while `method P (2, 1)` is not valid.

**Note:** This method can only be used with file types IXF, DEL, or CURSOR, and is the only valid method for the DEL file type.

**MODIFIED BY filetype-mod**

Specifies file type modifier options. See File type modifiers for load.

**NONRECOVERABLE**

Specifies that the load transaction is to be marked as non-recoverable and that it will not be possible to recover it by a subsequent roll forward action. The roll forward utility will skip the transaction and will mark the table into which data was being loaded as "invalid". The utility will also ignore any subsequent transactions against that table. After the roll forward operation is completed, such a table can only be dropped or restored from a backup (full or table space) taken after a commit point following the completion of the non-recoverable load operation.

With this option, table spaces are not put in backup pending state following the load operation, and a copy of the loaded data does not have to be made during the load operation.

This option should not be used when DATALINK columns with the FILE LINK CONTROL attribute are present in, or being added to, the table.

**NULL INDICATORS null-indicator-list**

This option can only be used when the METHOD L parameter is specified; that is, the input file is an ASC file). The null indicator list is a comma-separated list of positive integers specifying the column number of each null indicator field. The column number is the byte offset of the null indicator field from the beginning of a row of data. There must be one entry in the null indicator list for each data field defined in the METHOD L parameter. A column number of zero indicates that the corresponding data field always contains data.

A value of `Y` in the NULL indicator column specifies that the column data is NULL. Any character *other than* `Y` in the NULL indicator column specifies that the column data is not NULL, and that column data specified by the METHOD L option will be loaded.

The NULL indicator character can be changed using the MODIFIED BY option.

**OF filetype**

Specifies the format of the data:

- ASC (non-delimited ASCII format)
- DEL (delimited ASCII format)
- IXF (integrated exchange format, PC version), exported from the same or from another DB2 table
- CURSOR (a cursor declared against a SELECT or VALUES statement).

**PARTITIONED DB CONFIG**

Allows you to execute a load into a partitioned table. The PARTITIONED DB CONFIG parameter allows you to specify partitioned database-specific configuration options. The partitioned-db-option values can be any of the following:

```
HOSTNAME x
FILE_TRANSFER_CMD x
PART_FILE_LOCATION x
OUTPUT_DBPARTNUMS x
PARTITIONING_DBPARTNUMS x
MODE x
MAX_NUM_PART_AGENTS x
ISOLATE_PART_ERRS x
STATUS_INTERVAL x
PORT_RANGE x
CHECK_TRUNCATION
MAP_FILE_INPUT x
MAP_FILE_OUTPUT x
TRACE x
NEWLINE
DISTFILE x
OMIT_HEADER
RUN_STAT_DBPARTNUM x
```

Detailed descriptions of these options are provided in Partitioned database load configuration options.

**REPLACE**

One of four modes under which the load utility can execute. Deletes all existing data from the table, and inserts the loaded data. The table definition and index definitions are not changed. If this option is used when moving data between hierarchies, only the data for an entire hierarchy, not individual subtables, can be replaced.

This option is not supported for tables with DATALINK columns.

**RESTART**

One of four modes under which the load utility can execute. Restarts a previously interrupted load operation. The load operation will automatically continue from the last consistency point in the load, build, or delete phase.

**RESTARTCOUNT**

Reserved.

**ROWCOUNT n**

Specifies the number of *n* physical records in the file to be loaded. Allows a user to load only the first *n* rows in a file.

**SAVECOUNT n**

Specifies that the load utility is to establish consistency points after every *n* rows. This value is converted to a page count, and rounded up to intervals of the extent size. Since a message is issued at each consistency point, this option should be selected if the load operation will be monitored using LOAD QUERY. If the value of *n* is not sufficiently high, the synchronization of activities performed at each consistency point will impact performance.

The default value is zero, meaning that no consistency points will be established, unless necessary.

This option is not supported in conjunction with the CURSOR filetype.

**SORT BUFFER buffer-size**

This option specifies a value that overrides the SORTHEAP database configuration parameter during a load operation. It is relevant only when loading tables with indexes and only when the INDEXING MODE parameter is not specified as DEFERRED. The value that is specified cannot exceed the value of SORTHEAP. This parameter is useful for throttling the sort memory that is used when loading tables with many indexes without changing the value of SORTHEAP, which would also affect general query processing.

**STATISTICS USE PROFILE**

Instructs load to collect statistics during the load according to the profile defined for this table. This profile must be created before load is executed. The profile is created by the RUNSTATS command. If the profile does not exist and load is instructed to collect statistics according to the profile, a warning is returned and no statistics are collected.

**STATISTICS NO**

Specifies that no statistics are to be collected, and that the statistics in the catalogs are not to be altered. This is the default.

**TEMPFILES PATH temp-pathname**

Specifies the name of the path to be used when creating temporary files during a load operation, and should be fully qualified according to the server database partition.

Temporary files take up file system space. Sometimes, this space requirement is quite substantial. Following is an estimate of how much file system space should be allocated for all temporary files:

- 4 bytes for each duplicate or rejected row containing DATALINK values
- 136 bytes for each message that the load utility generates
- 15KB overhead if the data file contains long field data or LOBs. This quantity can grow significantly if the INSERT option is specified, and there is a large amount of long field or LOB data already in the table.

**TERMINATE**

One of four modes under which the load utility can execute. Terminates a previously interrupted load operation, and rolls back the operation to the point in time at which it started, even if consistency points were passed. The states of any table spaces involved in the operation return to normal, and all table objects are made consistent (index objects might be marked as invalid, in which case index rebuild will automatically take place at next access). If the load operation being terminated is a load REPLACE, the table will be truncated to an empty table after the load TERMINATE operation. If the load operation being terminated is a load INSERT, the table will retain all of its original records after the load TERMINATE operation.

The load terminate option will not remove a backup pending state from table spaces.

**Note:** This option is not supported for tables with DATALINK columns.

**USING directory**

Reserved.

**WARNINGCOUNT n**

Stops the load operation after *n* warnings. Set this parameter if no warnings are expected, but verification that the correct file and table are being used is desired. If the load file or the target table is specified incorrectly, the load utility will generate a warning for each row that it attempts to load, which will cause the load to fail. If *n* is zero, or this option is not specified, the load operation will continue regardless of the number of warnings issued. If the load operation is stopped because the threshold of warnings was encountered, another load operation can be started in RESTART mode. The load operation will automatically continue from the last consistency point. Alternatively, another load operation can be initiated in REPLACE mode, starting at the beginning of the input file.

**WITHOUT PROMPTING**

Specifies that the list of data files contains all the files that are to be loaded, and that the devices or directories listed are sufficient for the entire load operation. If a continuation input file is not found, or the copy targets are filled before the load operation finishes, the load operation will fail, and the table will remain in load pending state.

If this option is not specified, and the tape device encounters an end of tape for the copy image, or the last item listed is a tape device, the user is prompted for a new tape on that device.

**Examples:**

**Example 1**

TABLE1 has 5 columns:

- COL1 VARCHAR 20 NOT NULL WITH DEFAULT
- COL2 SMALLINT
- COL3 CHAR 4
- COL4 CHAR 2 NOT NULL WITH DEFAULT
- COL5 CHAR 2 NOT NULL

ASCFILE1 has 6 elements:

- ELE1 positions 01 to 20
- ELE2 positions 21 to 22
- ELE5 positions 23 to 23
- ELE3 positions 24 to 27
- ELE4 positions 28 to 31
- ELE6 positions 32 to 32
- ELE6 positions 33 to 40

Data Records:

```
1...5....10...15...20...25...30...35...40
Test data 1         XXN 123abcdN
Test data 2 and 3   QQY    wxyzN
Test data 4,5 and 6 WWN6789    Y
```

The following command loads the table from the file:

```
db2 load from ascfile1 of asc modified by striptblanks reclen=40
   method L (1 20, 21 22, 24 27, 28 31)
   null indicators (0,0,23,32)
   insert into table1 (col1, col5, col2, col3)
```

**Notes:**

1. The specification of `striptblanks` in the MODIFIED BY parameter forces the truncation of blanks in VARCHAR columns (COL1, for example, which is 11, 17 and 19 bytes long, in rows 1, 2 and 3, respectively).
2. The specification of `reclen=40` in the MODIFIED BY parameter indicates that there is no new-line character at the end of each input record, and that each record is 40 bytes long. The last 8 bytes are not used to load the table.
3. Since COL4 is not provided in the input file, it will be inserted into TABLE1 with its default value (it is defined NOT NULL WITH DEFAULT).
4. Positions 23 and 32 are used to indicate whether COL2 and COL3 of TABLE1 will be loaded NULL for a given row. If there is a `Y` in the column's null indicator position for a given record, the column will be NULL. If there is an `N`, the data values in the column's data positions of the input record (as defined in L(........)) are used as the source of column data for the row. In this example, neither column in row 1 is NULL; COL2 in row 2 is NULL; and COL3 in row 3 is NULL.
5. In this example, the NULL INDICATORS for COL1 and COL5 are specified as `0` (zero), indicating that the data is not nullable.
6. The NULL INDICATOR for a given column can be anywhere in the input record, but the position must be specified, and the `Y` or `N` values must be supplied.

**Example 2 (Loading LOBs from Files)**

TABLE1 has 3 columns:

- COL1 CHAR 4 NOT NULL WITH DEFAULT
- LOB1 LOB
- LOB2 LOB

ASCFILE1 has 3 elements:

- ELE1 positions 01 to 04
- ELE2 positions 06 to 13
- ELE3 positions 15 to 22

The following files reside in either `/u/user1` or `/u/user1/bin`:

- ASCFILE2 has LOB data
- ASCFILE3 has LOB data
- ASCFILE4 has LOB data
- ASCFILE5 has LOB data
- ASCFILE6 has LOB data
- ASCFILE7 has LOB data

Data Records in ASCFILE1:

```
1...5....10...15...20...25...30.
REC1 ASCFILE2 ASCFILE3
REC2 ASCFILE4 ASCFILE5
REC3 ASCFILE6 ASCFILE7
```

The following command loads the table from the file:

```
db2 load from ascfile1 of asc
   lobs from /u/user1, /u/user1/bin
   modified by lobsinfile reclen=22
   method L (1 4, 6 13, 15 22)
   insert into table1
```

**Notes:**

1. The specification of `lobsinfile` in the MODIFIED BY parameter tells the loader that all LOB data is to be loaded from files.
2. The specification of `reclen=22` in the MODIFIED BY parameter indicates that there is no new-line character at the end of each input record, and that each record is 22 bytes long.
3. LOB data is contained in 6 files, ASCFILE2 through ASCFILE7. Each file contains the data that will be used to load a LOB column for a specific row. The relationship between LOBs and other data is specified in ASCFILE1. The first record of this file tells the loader to place REC1 in COL1 of row 1. The contents of ASCFILE2 will be used to load LOB1 of row 1, and the contents of ASCFILE3 will be used to load LOB2 of row 1. Similarly, ASCFILE4 and ASCFILE5 will be used to load LOB1 and LOB2 of row 2, and ASCFILE6 and ASCFILE7 will be used to load the LOBs of row 3.
4. The LOBS FROM parameter contains 2 paths that will be searched for the named LOB files when those files are required by the loader.
5. To load LOBs directly from ASCFILE1 (a non-delimited ASCII file), without the `lobsinfile` modifier, the following rules must be observed:
   - The total length of any record, including LOBs, cannot exceed 32KB.
   - LOB fields in the input records must be of fixed length, and LOB data padded with blanks as necessary.
   - The `striptblanks` modifier must be specified, so that the trailing blanks used to pad LOBs can be removed as the LOBs are inserted into the database.

**Example 3 (Using Dump Files)**

Table FRIENDS is defined as:

```
table friends "( c1 INT NOT NULL, c2 INT, c3 CHAR(8) )"
```

If an attempt is made to load the following data records into this table,

```
23, 24, bobby
, 45, john
4,, mary
```

the second row is rejected because the first INT is NULL, and the column definition specifies NOT NULL. Columns which contain initial characters that are not consistent with the DEL format will generate an error, and the record will be rejected. Such records can be written to a dump file.

DEL data appearing in a column outside of character delimiters is ignored, but does generate a warning. For example:

```
22,34,"bob"
24,55,"sam" sdf
```

The utility will load ″sam″ in the third column of the table, and the characters ″sdf″ will be flagged in a warning. The record is not rejected. Another example:

```
22 3, 34,"bob"
```

The utility will load `22,34,"bob"`, and generate a warning that some data in column one following the `22` was ignored. The record is not rejected.

**Example 4 (Loading DATALINK Data)**

The following command loads the table MOVIETABLE from the input file `delfile1`, which has data in the DEL format:

```
db2 load from delfile1 of del
   modified by dldel|
   insert into movietable (actorname, description, url_making_of,
   url_movie) datalink specification (dl_url_default_prefix
   "http://narang"), (dl_url_replace_prefix "http://bomdel"
   dl_url_suffix ".mpeg") for exception excptab
```

**Notes:**

1. The table has four columns:
   ```
   actorname              VARCHAR(n)
   description            VARCHAR(m)
   url_making_of          DATALINK (with LINKTYPE URL)
   url_movie              DATALINK (with LINKTYPE URL)
   ```
2. The DATALINK data in the input file has the vertical bar (`|`) character as the sub-field delimiter.
3. If any column value for url_making_of does not have the prefix character sequence, ″http://narang″ is used.
4. Each non-NULL column value for url_movie will get ″http://bomdel″ as its prefix. Existing values are replaced.
5. Each non-NULL column value for url_movie will get ″.mpeg″ appended to the path. For example, if a column value of url_movie is ″http://server1/x/y/z″, it will be stored as ″http://bomdel/x/y/z.mpeg″; if the value is ″/x/y/z″, it will be stored as ″http://bomdel/x/y/z.mpeg″.
6. If any unique index or DATALINK exception occurs while loading the table, the affected records are deleted from the table and put into the exception table `excptab`.

**Example 5 (Loading a Table with an Identity Column)**

TABLE1 has 4 columns:

- C1 VARCHAR(30)
- C2 INT GENERATED BY DEFAULT AS IDENTITY
- C3 DECIMAL(7,2)
- C4 CHAR(1)

TABLE2 is the same as TABLE1, except that C2 is a GENERATED ALWAYS identity column.

Data records in DATAFILE1 (DEL format):

```
"Liszt"
"Hummel",,187.43, H
"Grieg",100, 66.34, G
"Satie",101, 818.23, I
```

Data records in DATAFILE2 (DEL format):

```
"Liszt", 74.49, A
"Hummel", 0.01, H
"Grieg", 66.34, G
"Satie", 818.23, I
```

**Notes:**

1. The following command generates identity values for rows 1 and 2, since no identity values are supplied in DATAFILE1 for those rows. Rows 3 and 4, however, are assigned the user-supplied identity values of 100 and 101, respectively.

   ```
   db2 load from datafile1.del of del replace into table1
   ```

2. To load DATAFILE1 into TABLE1 so that identity values are generated for all rows, issue one of the following commands:

   ```
   db2 load from datafile1.del of del method P(1, 3, 4)
      replace into table1 (c1, c3, c4)
   db2load from datafile1.del of del modified by identityignore
      replace into table1
   ```

3. To load DATAFILE2 into TABLE1 so that identity values are generated for each row, issue one of the following commands:

   ```
   db2 load from datafile2.del of del replace into table1 (c1, c3, c4)
   db2 load from datafile2.del of del modified by identitymissing
      replace into table1
   ```

4. To load DATAFILE1 into TABLE2 so that the identity values of 100 and 101 are assigned to rows 3 and 4, issue the following command:

   ```
   db2 load from datafile1.del of del modified by identityoverride
      replace into table2
   ```

   In this case, rows 1 and 2 will be rejected, because the utility has been instructed to override system-generated identity values in favor of user-supplied values. If user-supplied values are not present, however, the row must be rejected, because identity columns are implicitly not NULL.

5. If DATAFILE1 is loaded into TABLE2 without using any of the identity-related file type modifiers, rows 1 and 2 will be loaded, but rows 3 and 4 will be rejected, because they supply their own non-NULL values, and the identity column is GENERATED ALWAYS.

**Example 6 (Loading using the CURSOR filetype)**

Table ABC.TABLE1 has 3 columns:

```
ONE INT
TWO CHAR(10)
THREE DATE
```

Table ABC.TABLE2 has 3 columns:

```
ONE VARCHAR
TWO INT
THREE DATE
```

Executing the following commands will load all the data from ABC.TABLE1 into ABC.TABLE2:

```
db2 declare mycurs cursor for select two,one,three from abc.table1
db2 load from mycurs of cursor insert into abc.table2
```

**Usage notes:**

Data is loaded in the sequence that appears in the input file. If a particular sequence is desired, the data should be sorted before a load is attempted.

The load utility builds indexes based on existing definitions. The exception tables are used to handle duplicates on unique keys. The utility does not enforce referential integrity, perform constraints checking, or update summary tables that are dependent on the tables being loaded. Tables that include referential or check constraints are placed in check pending state. Summary tables that are defined with REFRESH IMMEDIATE, and that are dependent on tables being loaded, are also placed in check pending state. Issue the SET INTEGRITY statement to take the tables out of check pending state. Load operations cannot be carried out on replicated summary tables.

If a clustering index exists on the table, the data should be sorted on the clustering index prior to loading. Data does not need to be sorted prior to loading into a multidimensional clustering (MDC) table, however.

**DB2 Data Links Manager considerations:**

For each DATALINK column, there can be one column specification within parentheses. Each column specification consists of one or more of DL_LINKTYPE, *prefix* and a DL_URL_SUFFIX specification. The *prefix* information can be either DL_URL_REPLACE_PREFIX, or the DL_URL_DEFAULT_PREFIX specification.

There can be as many DATALINK column specifications as the number of DATALINK columns defined in the table. The order of specifications follows the order of DATALINK columns as found within the insert-column list (if specified by INSERT INTO (insert-column, ...)), or within the table definition (if insert-column is not specified).

For example, if a table has columns C1, C2, C3, C4, and C5, and among them only columns C2 and C5 are of type DATALINK, and the insert-column list is (C1, C5, C3, C2), there should be two DATALINK column specifications. The first column specification will be for C5, and the second column specification will be for C2. If an insert-column list is not specified, the first column specification will be for C2, and the second column specification will be for C5.

If there are multiple DATALINK columns, and some columns do not need any particular specification, the column specification should have at least the parentheses to unambiguously identify the order of specifications. If there are no specifications for any of the columns, the entire list of empty parentheses can be dropped. Thus, in cases where the defaults are satisfactory, there need not be any DATALINK specification.

If data is being loaded into a table with a DATALINK column that is defined with FILE LINK CONTROL, perform the following steps before invoking the load utility. (If all the DATALINK columns are defined with NO LINK CONTROL, these steps are not necessary).

1. Ensure that the DB2 Data Links Manager is installed on the Data Links servers that will be referred to by the DATALINK column values.
2. Ensure that the database is registered with the DB2 Data Links Manager.
3. Copy to the appropriate Data Links servers, all files that will be inserted as DATALINK values.
4. Define the prefix name (or names) to the DB2 Data Links Managers on the Data Links servers.
5. Register the Data Links servers referred to by DATALINK data (to be loaded) in the DB2 Data Links Manager configuration file.

The connection between DB2 and the Data Links server might fail while running the load utility, causing the load operation to fail. If this occurs:

1. Start the Data Links server and the DB2 Data Links Manager.
2. Invoke a load restart operation.

Links that fail during the load operation are considered to be data integrity violations, and are handled in much the same way as unique index violations. Consequently, a special exception has been defined for loading tables that have one or more DATALINK columns.

**Representation of DATALINK information in an input file**

The LINKTYPE (currently only URL is supported) is not specified as part of DATALINK information. The LINKTYPE is specified in the LOAD or the IMPORT command, and for input files of type PC/IXF, in the appropriate column descriptor records.

The syntax of DATALINK information for a URL LINKTYPE is as follows:

```
>>─┬─────────┬──┬──────────────────────┬───────────────────────────────><
   └─urlname─┘  └─dl_delimiter──comment─┘
```

Note that both *urlname* and *comment* are optional. If neither is provided, the NULL value is assigned.

**urlname**
The URL name must conform to valid URL syntax.

**Notes:**

1. Currently "http", "file", and "unc" are permitted as a schema name.
2. The prefix (schema, host, and port) of the URL name is optional. If a prefix is not present, it is taken from the DL_URL_DEFAULT_PREFIX or the DL_URL_REPLACE_PREFIX specification of the load or the import utility. If none of these is specified, the prefix defaults to "file://localhost". Thus, in the case of local files, the file name with full path name can be entered as the URL name, without the need for a DATALINK column specification within the LOAD or the IMPORT command.
3. Prefixes, even if present in URL names, are overridden by a different prefix name on the DL_URL_REPLACE_PREFIX specification during a load or import operation.
4. The "path" (after appending DL_URL_SUFFIX, if specified) is the full path name of the remote file in the remote server. Relative path names are not allowed. The http server default path-prefix is not taken into account.

**dl_delimiter**
For the delimited ASCII (DEL) file format, a character specified via the `dldel` modifier, or defaulted to on the LOAD or the IMPORT command. For the non-delimited ASCII (ASC) file format, this should correspond to the character sequence `\;` (a backslash followed by a semicolon). Whitespace characters (blanks, tabs, and so on) are permitted before and after the value specified for this parameter.

**comment**
The comment portion of a DATALINK value. If specified for the delimited ASCII (DEL) file format, the *comment* text must be enclosed by the character string delimiter, which is double quotation marks (") by default.

This character string delimiter can be overridden by the MODIFIED BY *filetype-mod* specification of the LOAD or the IMPORT command.

If no comment is specified, the comment defaults to a string of length zero.

Following are DATALINK data examples for the delimited ASCII (DEL) file format:

- `http://www.almaden.ibm.com:80/mrep/intro.mpeg; "Intro Movie"`

  This is stored with the following parts:
  - scheme = http
  - server = www.almaden.ibm.com
  - path = /mrep/intro.mpeg
  - comment = ″Intro Movie″
- `file://narang/u/narang; "InderPal's Home Page"`

  This is stored with the following parts:
  - scheme = file
  - server = narang
  - path = /u/narang
  - comment = ″InderPal's Home Page″

Following are DATALINK data examples for the non-delimited ASCII (ASC) file format:

- `http://www.almaden.ibm.com:80/mrep/intro.mpeg\;Intro Movie`

  This is stored with the following parts:
  - scheme = http
  - server = www.almaden.ibm.com
  - path = /mrep/intro.mpeg
  - comment = ″Intro Movie″
- `file://narang/u/narang\; InderPal's Home Page`

  This is stored with the following parts:
  - scheme = file
  - server = narang
  - path = /u/narang
  - comment = ″InderPal's Home Page″

Following are DATALINK data examples in which the load or import specification for the column is assumed to be DL_URL_REPLACE_PREFIX (″http://qso″):

- `http://www.almaden.ibm.com/mrep/intro.mpeg`

  This is stored with the following parts:
  - schema = http
  - server = qso
  - path = /mrep/intro.mpeg
  - comment = NULL string
- `/u/me/myfile.ps`

  This is stored with the following parts:
  - schema = http
  - server = qso
  - path = /u/me/myfile.ps

– comment = NULL string

**Related concepts:**

- "Load Overview" in the *Data Movement Utilities Guide and Reference*
- "Privileges, authorities, and authorizations required to use Load" in the *Data Movement Utilities Guide and Reference*

**Related tasks:**

- "Using Load" in the *Data Movement Utilities Guide and Reference*

**Related reference:**

- "QUIESCE TABLESPACES FOR TABLE" on page 591
- "db2atld - Autoloader" on page 22
- "Load - CLP Examples" in the *Data Movement Utilities Guide and Reference*
- "Partitioned database load configuration options" in the *Data Movement Utilities Guide and Reference*
- "db2Load - Load" in the *Administrative API Reference*
- "File type modifiers for load" on page 541

## File type modifiers for load

*Table 17. Valid file type modifiers for load: All file formats*

| Modifier | Description |
|---|---|
| anyorder | This modifier is used in conjunction with the *cpu_parallelism* parameter. Specifies that the preservation of source data order is not required, yielding significant additional performance benefit on SMP systems. If the value of *cpu_parallelism* is 1, this option is ignored. This option is not supported if SAVECOUNT > 0, since crash recovery after a consistency point requires that data be loaded in sequence. |
| generatedignore | This modifier informs the load utility that data for all generated columns is present in the data file but should be ignored. This results in all generated column values being generated by the utility. This modifier cannot be used with either the `generatedmissing` or the `generatedoverride` modifier. |
| generatedmissing | If this modifier is specified, the utility assumes that the input data file contains no data for the generated column (not even NULLs). This results in all generated column values being generated by the utility. This modifier cannot be used with either the `generatedignore` or the `generatedoverride` modifier. |

*Table 17. Valid file type modifiers for load: All file formats (continued)*

| Modifier | Description |
|---|---|
| generatedoverride | This modifier instructs the load utility to accept user-supplied data for all generated columns in the table (contrary to the normal rules for these types of columns). This is useful when migrating data from another database system, or when loading a table from data that was recovered using the RECOVER DROPPED TABLE option on the ROLLFORWARD DATABASE command. When this modifier is used, any rows with no data or NULL data for a non-nullable generated column will be rejected (SQL3116W).<br>**Note:** When this modifier is used, the table will be placed in CHECK PENDING state. To take the table out of CHECK PENDING state without verifying the user-supplied values, issue the following command after the load operation:<br>`SET INTEGRITY FOR < table-name > GENERATED COLUMN`<br>`  IMMEDIATED UNCHECKED`<br>To take the table out of CHECK PENDING state and force verification of the user-supplied values, issue the following command after the load operation:<br>`SET INTEGRITY FOR < table-name > IMMEDIATE CHECKED.`<br>This modifier cannot be used with either the `generatedmissing` or the `generatedignore` modifier. |
| identityignore | This modifier informs the load utility that data for the identity column is present in the data file but should be ignored. This results in all identity values being generated by the utility. The behavior will be the same for both GENERATED ALWAYS and GENERATED BY DEFAULT identity columns. This means that for GENERATED ALWAYS columns, no rows will be rejected. This modifier cannot be used with either the `identitymissing` or the `identityoverride` modifier. |
| identitymissing | If this modifier is specified, the utility assumes that the input data file contains no data for the identity column (not even NULLs), and will therefore generate a value for each row. The behavior will be the same for both GENERATED ALWAYS and GENERATED BY DEFAULT identity columns. This modifier cannot be used with either the `identityignore` or the `identityoverride` modifier. |
| identityoverride | This modifier should be used only when an identity column defined as GENERATED ALWAYS is present in the table to be loaded. It instructs the utility to accept explicit, non-NULL data for such a column (contrary to the normal rules for these types of identity columns). This is useful when migrating data from another database system when the table must be defined as GENERATED ALWAYS, or when loading a table from data that was recovered using the DROPPED TABLE RECOVERY option on the ROLLFORWARD DATABASE command. When this modifier is used, any rows with no data or NULL data for the identity column will be rejected (SQL3116W). This modifier cannot be used with either the `identitymissing` or the `identityignore` modifier.<br>**Note:** The load utility will not attempt to maintain or verify the uniqueness of values in the table's identity column when this option is used. |
| indexfreespace=*x* | *x* is an integer between 0 and 99 inclusive. The value is interpreted as the percentage of each index page that is to be left as free space when load rebuilds the index. Load with INDEXING MODE INCREMENTAL ignores this option. The first entry in a page is added without restriction; subsequent entries are added the percent free space threshold can be maintained. The default value is the one used at CREATE INDEX time.<br>This value takes precedence over the PCTFREE value specified in the CREATE INDEX statement; the registry variable DB2 INDEX FREE takes precedence over indexfreespace. The indexfreespace option affects index leaf pages only. |

Table 17. Valid file type modifiers for load: All file formats (continued)

| Modifier | Description |
|---|---|
| lobsinfile | *lob-path* specifies the path to the files containing LOB data. The ASC, DEL, or IXF load input files contain the names of the files having LOB data in the LOB column.<br>This option is not supported in conjunction with the CURSOR filetype.<br>The LOBS FROM clause specifies where the LOB files are located when the "lobsinfile" modifier is used. The LOBS FROM clause means nothing outside the context of the "lobsinfile" modifier. The LOBS FROM clause conveys to the LOAD utility the list of paths to search for the LOB files while loading the data.<br>Each path contains at least one file that contains at least one LOB pointed to by a Lob Location Specifier (LLS) in the data file. The LLS is a string representation of the location of a LOB in a file stored in the LOB file path. The format of an LLS is *filename.ext.nnn.mmm/*, where *filename.ext* is the name of the file that contains the LOB, *nnn* is the offset in bytes of the LOB within the file, and *mmm* is the length of the LOB in bytes. For example, if the string `db2exp.001.123.456/` is stored in the data file, the LOB is located at offset 123 in the file db2exp.001, and is 456 bytes long.<br>To indicate a null LOB , enter the size as -1. If the size is specified as 0, it is treated as a 0 length LOB. For null LOBS with length of -1, the offset and the file name are ignored. For example, the LLS of a null LOB might be db2exp.001.7.-1/. |
| noheader | Skips the header verification code (applicable only to load operations into tables that reside in a single-partition database partition group).<br>The AutoLoader utility writes a header to each file contributing data to a table in a multiple-partition database partition group. If the default MPP load (mode PARTITION_AND_LOAD) is used against a table residing in a single-partition database partition group, the file is not expected to have a header. Thus the noheader modifier is not needed. If the LOAD_ONLY mode is used, the file is expected to have a header. The only circumstance in which you should need to use the noheader modifier is if you wanted to perform LOAD_ONLY operation using a file that does not have a header. |
| norowwarnings | Suppresses all warnings about rejected rows. |
| pagefreespace=*x* | *x* is an integer between 0 and 100 inclusive. The value is interpreted as the percentage of each data page that is to be left as free space. If the specified value is invalid because of the minimum row size, (for example, a row that is at least 3 000 bytes long, and an *x* value of 50), the row will be placed on a new page. If a value of 100 is specified, each row will reside on a new page.<br>**Note:** The PCTFREE value of a table determines the amount of free space designated per page. If a pagefreespace value on the load operation or a PCTFREE value on a table have not been set, the utility will fill up as much space as possible on each page. The value set by pagefreespace overrides the PCTFREE value specified for the table. |
| subtableconvert | Valid only when loading into a single sub-table. Typical usage is to export data from a regular table, and then to invoke a load operation (using this modifier) to convert the data into a single sub-table. |

*Table 17. Valid file type modifiers for load: All file formats (continued)*

| Modifier | Description |
|---|---|
| totalfreespace=*x* | *x* is an integer greater than or equal to 0 . The value is interpreted as the percentage of the total pages in the table that is to be appended to the end of the table as free space. For example, if *x* is 20, and the table has 100 data pages after the data has been loaded, 20 additional empty pages will be appended. The total number of data pages for the table will be 120. The data pages total does not factor in the number of index pages in the table. This option does not affect the index object.<br>**Note:** If two loads are done with this option specified, the second load will not reuse the extra space appended to the end by the first load. |
| usedefaults | If a source column for a target table column has been specified, but it contains no data for one or more row instances, default values are loaded. Examples of missing data are:<br>• For DEL files: ",," is specified for the column<br>• For DEL/ASC/WSF files: A row that does not have enough columns, or is not long enough for the original specification.<br>Without this option, if a source column contains no data for a row instance, one of the following occurs:<br>• If the column is nullable, a NULL is loaded<br>• If the column is not nullable, the utility rejects the row. |

*Table 18. Valid file type modifiers for load: ASCII file formats (ASC/DEL)*

| Modifier | Description |
|---|---|
| codepage=*x* | *x* is an ASCII character string. The value is interpreted as the code page of the data in the input data set. Converts character data (and numeric data specified in characters) from this code page to the database code page during the load operation.<br>The following rules apply:<br>• For pure DBCS (graphic), mixed DBCS, and EUC, delimiters are restricted to the range of x00 to x3F, inclusive.<br>• For DEL data specified in an EBCDIC code page, the delimiters might not coincide with the shift-in and shift-out DBCS characters.<br>• `nullindchar` must specify symbols included in the standard ASCII set between code points x20 and x7F, inclusive. This refers to ASCII symbols and code points. EBCDIC data can use the corresponding symbols, even though the code points will be different.<br>This option is not supported in conjunction with the CURSOR filetype. |

*Table 18. Valid file type modifiers for load: ASCII file formats (ASC/DEL) (continued)*

| Modifier | Description |
|---|---|
| dateformat="*x*" | *x* is the format of the date in the source file.[1] Valid date elements are:<br>`YYYY - Year (four digits ranging from 0000 - 9999)`<br>`M    - Month (one or two digits ranging from 1 - 12)`<br>`MM   - Month (two digits ranging from 1 - 12;`<br>`          mutually exclusive with M)`<br>`D    - Day (one or two digits ranging from 1 - 31)`<br>`DD   - Day (two digits ranging from 1 - 31;`<br>`          mutually exclusive with D)`<br>`DDD  - Day of the year (three digits ranging`<br>`          from 001 - 366; mutually exclusive`<br>`          with other day or month elements)`<br>A default value of 1 is assigned for each element that is not specified. Some examples of date formats are:<br>`"D-M-YYYY"`<br>`"MM.DD.YYYY"`<br>`"YYYYDDD"` |
| dumpfile = *x* | *x* is the fully qualified (according to the server database partition) name of an exception file to which rejected rows are written. A maximum of 32 KB of data is written per record. Following is an example that shows how to specify a dump file:<br>`db2 load from data of del`<br>`   modified by dumpfile = /u/user/filename`<br>`   insert into table_name`<br>The file will be created and owned by the instance owner. To override the default file permissions, use the `dumpfileaccessall` file type modifier.<br>**Notes:**<br>1. In a partitioned database environment, the path should be local to the loading database partition, so that concurrently running load operations do not attempt to write to the same file.<br>2. The contents of the file are written to disk in an asynchronous buffered mode. In the event of a failed or an interrupted load operation, the number of records committed to disk cannot be known with certainty, and consistency cannot be guaranteed after a LOAD RESTART. The file can only be assumed to be complete for a load operation that starts and completes in a single pass.<br>3. This modifier does not support file names with multiple file extensions. For example,<br>`dumpfile = /home/svtdbm6/DUMP.FILE`<br>is acceptable to the load utility, but<br>`dumpfile = /home/svtdbm6/DUMP.LOAD.FILE`<br>is not. |
| dumpfileaccessall = *x* | Grants read access to 'OTHERS' when a dump file is created.<br>This file type modifier is only valid when:<br>1. it is used in conjunction with `dumpfile` file type modifier<br>2. the user has SELECT privilege on the load target table<br>3. it is issued on a DB2 server database partition that resides on a UNIX-based operating system |

*Table 18. Valid file type modifiers for load: ASCII file formats (ASC/DEL) (continued)*

| **Modifier** | **Description** |
|---|---|
| fastparse | Reduced syntax checking is done on user-supplied column values, and performance is enhanced. Tables loaded under this option are guaranteed to be architecturally correct, and the utility is guaranteed to perform sufficient data checking to prevent a segmentation violation or trap. Data that is in correct form will be loaded correctly.<br>For example, if a value of `123qwr4` were to be encountered as a field entry for an integer column in an ASC file, the load utility would ordinarily flag a syntax error, since the value does not represent a valid number. With `fastparse`, a syntax error is not detected, and an arbitrary number is loaded into the integer field. Care must be taken to use this modifier with clean data only. Performance improvements using this option with ASCII data can be quite substantial.<br>This option is not supported in conjunction with the CURSOR or IXF file types. |
| implieddecimal | The location of an implied decimal point is determined by the column definition; it is no longer assumed to be at the end of the value. For example, the value `12345` is loaded into a DECIMAL(8,2) column as `123.45`, *not* `12345.00`.<br>This modifier cannot be used with the `packeddecimal` modifier. |
| timeformat="*x*" | *x* is the format of the time in the source file.[1] Valid time elements are:<br>`H     - Hour (one or two digits ranging from 0 - 12 for a 12 hour system, and 0 - 24 for a 24 hour system)`<br>`HH    - Hour (two digits ranging from 0 - 12 for a 12 hour system, and 0 - 24 for a 24 hour system; mutually exclusive with H)`<br>`M     - Minute (one or two digits ranging from 0 - 59)`<br>`MM    - Minute (two digits ranging from 0 - 59; mutually exclusive with M)`<br>`S     - Second (one or two digits ranging from 0 - 59)`<br>`SS    - Second (two digits ranging from 0 - 59; mutually exclusive with S)`<br>`SSSSS - Second of the day after midnight (5 digits ranging from 00000 - 86399; mutually exclusive with other time elements)`<br>`TT    - Meridian indicator (AM or PM)`<br>A default value of 0 is assigned for each element that is not specified. Some examples of time formats are:<br>`"HH:MM:SS"`<br>`"HH.MM TT"`<br>`"SSSSS"` |

*Table 18. Valid file type modifiers for load: ASCII file formats (ASC/DEL) (continued)*

| Modifier | Description |
|---|---|
| timestampformat="*x*" | *x* is the format of the time stamp in the source file.[1] Valid time stamp elements are:<br>`YYYY   - Year (four digits ranging from 0000 - 9999)`<br>`M      - Month (one or two digits ranging from 1 - 12)`<br>`MM     - Month (two digits ranging from 01 - 12; mutually exclusive with M and MMM)`<br>`MMM    - Month (three-letter case-insensitive abbreviation for the month name; mutually exclusive with M and MM)`<br>`D      - Day (one or two digits ranging from 1 - 31)`<br>`DD     - Day (two digits ranging from 1 - 31; mutually exclusive with D)`<br>`DDD    - Day of the year (three digits ranging from 001 - 366; mutually exclusive with other day or month elements)`<br>`H      - Hour (one or two digits ranging from 0 - 12 for a 12 hour system, and 0 - 24 for a 24 hour system)`<br>`HH     - Hour (two digits ranging from 0 - 12 for a 12 hour system, and 0 - 24 for a 24 hour system; mutually exclusive with H)`<br>`M      - Minute (one or two digits ranging from 0 - 59)`<br>`MM     - Minute (two digits ranging from 0 - 59; mutually exclusive with M, minute)`<br>`S      - Second (one or two digits ranging from 0 - 59)`<br>`SS     - Second (two digits ranging from 0 - 59; mutually exclusive with S)`<br>`SSSSS  - Second of the day after midnight (5 digits ranging from 00000 - 86399; mutually exclusive with other time elements)`<br>`UUUUUU - Microsecond (6 digits ranging from 000000 - 999999; mutually exclusive with all other microsecond elements)`<br>`UUUUU  - Microsecond (5 digits ranging  from 00000 - 99999, maps to range from 000000 - 999990; mutually exclusive with all other microseond elements)`<br>`UUUU   - Microsecond (4 digits ranging from 0000 - 9999, maps to range from 000000 - 999900; mutually exclusive with all other microseond elements)`<br>`UUU    - Microsecond (3 digits ranging from 000 - 999, maps to range from 000000 - 999000; mutually exclusive with all other microseond elements)`<br>`UU     - Microsecond (2 digits ranging from 00 - 99, maps to range from 000000 - 990000; mutually exclusive with all other microseond elements)`<br>`U      - Microsecond (1 digit ranging from 0 - 9, maps to range from 000000 - 900000; mutually exclusive with all other microseond elements)`<br>`TT     - Meridian indicator (AM or PM)`<br>A default value of 1 is assigned for unspecified YYYY, M, MM, D, DD, or DDD elements. A default value of 'Jan' is assigned to an unspecified MMM element. A default value of 0 is assigned for all other unspecified elements. Following is an example of a time stamp format:<br>`"YYYY/MM/DD HH:MM:SS.UUUUUU"`<br>The valid values for the MMM element include: 'jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov' and 'dec'. These values are case insensitive.<br>The following example illustrates how to import data containing user defined date and time formats into a table called schedule:<br>`db2 import from delfile2 of del`<br>`   modified by timestampformat="yyyy.mm.dd hh:mm tt"`<br>`   insert into schedule` |
| noeofchar | The optional end-of-file character `x'1A'` is not recognized as the end of file. Processing continues as if it were a normal character. |

Table 18. Valid file type modifiers for load: ASCII file formats (ASC/DEL) (continued)

| Modifier | Description |
|---|---|
| usegraphiccodepage | If usegraphiccodepage is given, the assumption is made that data being loaded into graphic or double-byte character large object (DBCLOB) data field(s) is in the graphic code page. The rest of the data is assumed to be in the character code page. The graphic codepage is associated with the character code page. LOAD determines the character code page through either the codepage modifier, if it is specified, or through the code page of the database if the codepage modifier is not specified.<br>This modifier should be used in conjunction with the delimited data file generated by drop table recovery only if the table being recovered has graphic data.<br>**Restrictions**<br>The usegraphiccodepage modifier MUST NOT be specified with DEL or ASC files created by the EXPORT utility, as these files contain data encoded in only one code page. The usegraphiccodepage modifier is also ignored by the double-byte character large objects (DBCLOBs) in files. |

Table 19. Valid file type modifiers for load: ASC file formats (Non-delimited ASCII)

| Modifier | Description |
|---|---|
| binarynumerics | Numeric (but not DECIMAL) data must be in binary form, not the character representation. This avoids costly conversions.<br>This option is supported only with positional ASC, using fixed length records specified by the reclen option. The noeofchar option is assumed.<br>The following rules apply:<br>• No conversion between data types is performed, with the exception of BIGINT, INTEGER, and SMALLINT.<br>• Data lengths must match their target column definitions.<br>• FLOATs must be in IEEE Floating Point format.<br>• Binary data in the load source file is assumed to be big-endian, regardless of the platform on which the load operation is running.<br>**Note:** NULLs cannot be present in the data for columns affected by this modifier. Blanks (normally interpreted as NULL) are interpreted as a binary value when this modifier is used. |
| nochecklengths | If nochecklengths is specified, an attempt is made to load each row, even if the source data has a column definition that exceeds the size of the target table column. Such rows can be successfully loaded if code page conversion causes the source data to shrink; for example, 4-byte EUC data in the source could shrink to 2-byte DBCS data in the target, and require half the space. This option is particularly useful if it is known that the source data will fit in all cases despite mismatched column definitions. |
| nullindchar=*x* | *x* is a single character. Changes the character denoting a NULL value to *x*. The default value of *x* is Y.[2]<br>This modifier is case sensitive for EBCDIC data files, except when the character is an English letter. For example, if the NULL indicator character is specified to be the letter N, then n is also recognized as a NULL indicator. |

*Table 19. Valid file type modifiers for load: ASC file formats (Non-delimited ASCII) (continued)*

| Modifier | Description |
|---|---|
| packeddecimal | Loads packed-decimal data directly, since the `binarynumerics` modifier does not include the DECIMAL field type.<br>This option is supported only with positional ASC, using fixed length records specified by the `reclen` option. The `noeofchar` option is assumed.<br>Supported values for the sign nibble are:<br>`+ = 0xC 0xA 0xE 0xF`<br>`- = 0xD 0xB`<br>NULLs cannot be present in the data for columns affected by this modifier. Blanks (normally interpreted as NULL) are interpreted as a binary value when this modifier is used.<br>Regardless of the server platform, the byte order of binary data in the load source file is assumed to be big-endian; that is, when using this modifier on Windows operating systems, the byte order must not be reversed.<br>This modifier cannot be used with the `implieddecimal` modifier. |
| reclen=*x* | *x* is an integer with a maximum value of 32 767. *x* characters are read for each row, and a new-line character is not used to indicate the end of the row. |
| striptblanks | Truncates any trailing blank spaces when loading data into a variable-length field. If this option is not specified, blank spaces are kept.<br>This option cannot be specified together with `striptnulls`. These are mutually exclusive options.<br>**Note:** This option replaces the obsolete `t` option, which is supported for back-level compatibility only. |
| striptnulls | Truncates any trailing NULLs (0x00 characters) when loading data into a variable-length field. If this option is not specified, NULLs are kept.<br>This option cannot be specified together with `striptblanks`. These are mutually exclusive options.<br>**Note:** This option replaces the obsolete `padwithzero` option, which is supported for back-level compatibility only. |
| zoneddecimal | Loads zoned decimal data, since the BINARYNUMERICS modifier does not include the DECIMAL field type. This option is supported only with positional ASC, using fixed length records specified by the RECLEN option. The NOEOFCHAR option is assumed.<br>Half-byte sign values can be one of the following:<br>`+ = 0xC 0xA 0xE 0xF`<br>`- = 0xD 0xB`<br>Supported values for digits are `0x0` to `0x9.`<br>Supported values for zones are `0x3` and `0xF.` |

*Table 20. Valid file type modifiers for load: DEL file formats (Delimited ASCII)*

| Modifier | Description |
|---|---|
| chardel*x* | *x* is a single character string delimiter. The default value is a double quotation mark ("). The specified character is used in place of double quotation marks to enclose a character string.[23] If you wish to explicitly specify the double quotation mark(") as the character string delimiter, you should specify it as follows:<br>`modified by chardel""`<br>The single quotation mark (') can also be specified as a character string delimiter as follows:<br>`modified by chardel''` |
| coldel*x* | *x* is a single character column delimiter. The default value is a comma (,). The specified character is used in place of a comma to signal the end of a column.[23] |
| datesiso | Date format. Causes all date data values to be loaded in ISO format. |
| decplusblank | Plus sign character. Causes positive decimal values to be prefixed with a blank space instead of a plus sign (+). The default action is to prefix positive decimal values with a plus sign. |
| decpt*x* | *x* is a single character substitute for the period as a decimal point character. The default value is a period (.). The specified character is used in place of a period as a decimal point character.[23] |
| delprioritychar | The current default priority for delimiters is: record delimiter, character delimiter, column delimiter. This modifier protects existing applications that depend on the older priority by reverting the delimiter priorities to: character delimiter, record delimiter, column delimiter. Syntax:<br>`db2 load ... modified by delprioritychar ...`<br>For example, given the following DEL data file:<br>`"Smith, Joshua",4000,34.98<row delimiter>`<br>`"Vincent,<row delimiter>, is a manager", ...`<br>`... 4005,44.37<row delimiter>`<br>With the `delprioritychar` modifier specified, there will be only two rows in this data file. The second <row delimiter> will be interpreted as part of the first data column of the second row, while the first and the third <row delimiter> are interpreted as actual record delimiters. If this modifier is *not* specified, there will be three rows in this data file, each delimited by a <row delimiter>. |
| dldel*x* | *x* is a single character DATALINK delimiter. The default value is a semicolon (;). The specified character is used in place of a semicolon as the inter-field separator for a DATALINK value. It is needed because a DATALINK value can have more than one sub-value. [234]<br>**Note:** *x* must not be the same character specified as the row, column, or character string delimiter. |
| keepblanks | Preserves the leading and trailing blanks in each field of type CHAR, VARCHAR, LONG VARCHAR, or CLOB. Without this option, all leading and tailing blanks that are not inside character delimiters are removed, and a NULL is inserted into the table for all blank fields.<br>The following example illustrates how to load data into a table called TABLE1, while preserving all leading and trailing spaces in the data file:<br>`db2 load from delfile3 of del`<br>`modified by keepblanks`<br>`insert into table1` |

*Table 20. Valid file type modifiers for load: DEL file formats (Delimited ASCII) (continued)*

| Modifier | Description |
|---|---|
| nochardel | The load utility will assume all bytes found between the column delimiters to be part of the column's data. Character delimiters will be parsed as part of column data. This option should not be specified if the data was exported using DB2 (unless nochardel was specified at export time). It is provided to support vendor data files that do not have character delimiters. Improper usage might result in data loss or corruption.<br>This option cannot be specified with `chardelx`, `delprioritychar` or `nodoubledel`. These are mutually exclusive options. |
| nodoubledel | Suppresses recognition of double character delimiters. |

*Table 21. Valid file type modifiers for load: IXF file format*

| Modifier | Description |
|---|---|
| forcein | Directs the utility to accept data despite code page mismatches, and to suppress translation between code pages.<br>Fixed length target fields are checked to verify that they are large enough for the data. If `nochecklengths` is specified, no checking is done, and an attempt is made to load each row. |
| nochecklengths | If `nochecklengths` is specified, an attempt is made to load each row, even if the source data has a column definition that exceeds the size of the target table column. Such rows can be successfully loaded if code page conversion causes the source data to shrink; for example, 4-byte EUC data in the source could shrink to 2-byte DBCS data in the target, and require half the space. This option is particularly useful if it is known that the source data will fit in all cases despite mismatched column definitions. |

**Notes:**

1. Double quotation marks around the date format string are mandatory. Field separators cannot contain any of the following: a-z, A-Z, and 0-9. The field separator should not be the same as the character delimiter or field delimiter in the DEL file format. A field separator is optional if the start and end positions of an element are unambiguous. Ambiguity can exist if (depending on the modifier) elements such as D, H, M, or S are used, because of the variable length of the entries.

   For time stamp formats, care must be taken to avoid ambiguity between the month and the minute descriptors, since they both use the letter M. A month field must be adjacent to other date fields. A minute field must be adjacent to other time fields. Following are some ambiguous time stamp formats:

   ```
   "M" (could be a month, or a minute)
   "M:M" (Which is which?)
   "M:YYYY:M" (Both are interpreted as month.)
   "S:M:YYYY" (adjacent to both a time value and a date value)
   ```

   In ambiguous cases, the utility will report an error message, and the operation will fail.

   Following are some unambiguous time stamp formats:

   ```
   "M:YYYY" (Month)
   "S:M" (Minute)
   "M:YYYY:S:M" (Month....Minute)
   "M:H:YYYY:M:D" (Minute....Month)
   ```

   Some characters, such as double quotation marks and back slashes, must be preceded by an escape character (for example, \).

2. The character must be specified in the code page of the source data.

   The character code point (instead of the character symbol), can be specified using the syntax xJJ or 0xJJ, where JJ is the hexadecimal representation of the code point. For example, to specify the # character as a column delimiter, use one of the following:

   ```
   ... modified by coldel# ...
   ... modified by coldel0x23 ...
   ... modified by coldelX23 ...
   ```
3. Delimiter restrictions for moving data lists restrictions that apply to the characters that can be used as delimiter overrides.
4. Even if the DATALINK delimiter character is a valid character within the URL syntax, it will lose its special meaning within the scope of the load operation.
5. The load utility does not issue a warning if an attempt is made to use unsupported file types with the MODIFIED BY option. If this is attempted, the load operation fails, and an error code is returned.

*Table 22. LOAD behavior when using codepage and usegraphiccodepage*

| **codepage=N** | **usegraphiccodepage** | **LOAD behavior** |
|---|---|---|
| Absent | Absent | All data in the file is assumed to be in the database code page, not the application code page, even if the CLIENT option is specified. |
| Present | Absent | All data in the file is assumed to be in code page `N`.<br>**Warning:** Graphic data will be corrupted when loaded into the database if `N` is a single-byte code page. |
| Absent | Present | Character data in the file is assumed to be in the database code page, even if the CLIENT option is specified. Graphic data is assumed to be in the code page of the database graphic data, even if the CLIENT option is specified.<br>If the database code page is single-byte, then all data is assumed to be in the database code page.<br>**Warning:** Graphic data will be corrupted when loaded into a single-byte database. |
| Present | Present | Character data is assumed to be in code page `N`. Graphic data is assumed to be in the graphic code page of `N`.<br>If `N` is a single-byte or double-byte code page, then all data is assumed to be in code page `N`.<br>**Warning:** Graphic data will be corrupted when loaded into the database if `N` is a single-byte code page. |

**Related reference:**

- "LOAD" on page 520
- "db2Load - Load" in the *Administrative API Reference*
- "Delimiter restrictions for moving data" on page 370

## Delimiter restrictions for moving data

**Delimiter restrictions:**

It is the user's responsibility to ensure that the chosen delimiter character is not part of the data to be moved. If it is, unexpected errors might occur. The following restrictions apply to column, string, DATALINK, and decimal point delimiters when moving data:

- Delimiters are mutually exclusive.
- A delimiter cannot be binary zero, a line-feed character, a carriage-return, or a blank space.
- The default decimal point (.) cannot be a string delimiter.
- The following characters are specified differently by an ASCII-family code page and an EBCDIC-family code page:
  - The Shift-In (0x0F) and the Shift-Out (0x0E) character cannot be delimiters for an EBCDIC MBCS data file.
  - Delimiters for MBCS, EUC, or DBCS code pages cannot be greater than 0x40, except the default decimal point for EBCDIC MBCS data, which is 0x4b.
  - Default delimiters for data files in ASCII code pages or EBCDIC MBCS code pages are:

    ```
    " (0x22, double quotation mark; string delimiter)
    , (0x2c, comma; column delimiter)
    ```
  - Default delimiters for data files in EBCDIC SBCS code pages are:

    ```
    " (0x7F, double quotation mark; string delimiter)
    , (0x6B, comma; column delimiter)
    ```
  - The default decimal point for ASCII data files is 0x2e (period).
  - The default decimal point for EBCDIC data files is 0x4B (period).
  - If the code page of the server is different from the code page of the client, it is recommended that the hex representation of non-default delimiters be specified. For example,

    ```
    db2 load from ... modified by chardel0x0C coldelX1e ...
    ```

The following information about support for double character delimiter recognition in DEL files applies to the export, import, and load utilities:

- Character delimiters are permitted within the character-based fields of a DEL file. This applies to fields of type CHAR, VARCHAR, LONG VARCHAR, or CLOB (except when `lobsinfile` is specified). Any pair of character delimiters found between the enclosing character delimiters is imported or loaded into the database. For example,

  ```
  "What a ""nice"" day!"
  ```

  will be imported as:

  ```
  What a "nice" day!
  ```

  In the case of export, the rule applies in reverse. For example,

  ```
  I am 6" tall.
  ```

  will be exported to a DEL file as:

  ```
  "I am 6"" tall."
  ```
- In a DBCS environment, the pipe (|) character delimiter is not supported.

# LOAD QUERY

Checks the status of a load operation during processing and returns the table state. If a load is not processing, then the table state alone is returned. A connection to the same database, and a separate CLP session are also required to successfully invoke this command. It can be used either by local or remote users.

**Authorization:**

None

**Required connection:**

Database

**Command syntax:**

```
►►──LOAD QUERY──TABLE──table-name──┬───────────────────────┬──┬──────────────┬──────►
                                   └─TO──local-message-file─┘  ├─NOSUMMARY────┤
                                                               └─SUMMARYONLY──┘

►──┬───────────┬─────────────────────────────────────────────────────────────────►◄
   └─SHOWDELTA─┘
```

**Command parameters:**

**NOSUMMARY**
Specifies that no load summary information (rows read, rows skipped, rows loaded, rows rejected, rows deleted, rows committed, and number of warnings) is to be reported.

**SHOWDELTA**
Specifies that only new information (pertaining to load events that have occurred since the last invocation of the LOAD QUERY command) is to be reported.

**SUMMARYONLY**
Specifies that only load summary information is to be reported.

**TABLE table-name**
Specifies the name of the table into which data is currently being loaded. If an unqualified table name is specified, the table will be qualified with the CURRENT SCHEMA.

**TO local-message-file**
Specifies the destination for warning and error messages that occur during the load operation. This file cannot be the *message-file* specified for the LOAD command. If the file already exists, all messages that the load utility has generated are appended to it.

**Examples:**

A user loading a large amount of data into the STAFF table wants to check the status of the load operation. The user can specify:

```
db2 connect to <database>
db2 load query table staff to /u/mydir/staff.tempmsg
```

The output file `/u/mydir/staff.tempmsg` might look like the following:

```
SQL3501W  The table space(s) in which the table resides will not be placed in
backup pending state since forward recovery is disabled for the database.

SQL3109N  The utility is beginning to load data from file
"/u/mydir/data/staffbig.del"

SQL3500W  The utility is beginning the "LOAD" phase at time "03-21-2002
11:31:16.597045".

SQL3519W  Begin Load Consistency Point. Input record count = "0".

SQL3520W  Load Consistency Point was successful.

SQL3519W  Begin Load Consistency Point. Input record count = "104416".

SQL3520W  Load Consistency Point was successful.

SQL3519W  Begin Load Consistency Point. Input record count = "205757".

SQL3520W  Load Consistency Point was successful.

SQL3519W  Begin Load Consistency Point. Input record count = "307098".

SQL3520W  Load Consistency Point was successful.

SQL3519W  Begin Load Consistency Point. Input record count = "408439".

SQL3520W  Load Consistency Point was successful.

SQL3532I  The Load utility is currently in the "LOAD" phase.


Number of rows read         = 453376
Number of rows skipped      = 0
Number of rows loaded       = 453376
Number of rows rejected     = 0
Number of rows deleted      = 0
Number of rows committed    = 408439
Number of warnings          = 0

Tablestate:
  Load in Progress
```

**Usage Notes:**

In addition to locks, the load utility uses table states to control access to the table. The LOAD QUERY command can be used to determine the table state; LOAD QUERY can be used on tables that are not currently being loaded. The table states described by LOAD QUERY are described in Table locking, table states and table space states.

The progress of a load operation can also be monitored with the LIST UTILITIES command.

**Related concepts:**

- "Load Overview" in the *Data Movement Utilities Guide and Reference*
- "Table locking, table states and table space states" in the *Data Movement Utilities Guide and Reference*

**Related reference:**

- "LOAD" on page 520
- "LIST UTILITIES" on page 518

# MIGRATE DATABASE

Converts previous versions of DB2 databases to current formats.

**Attention:** The database pre-migration tool must be run prior to DB2 Version 8 installation (on Windows operating systems), or before instance migration (on UNIX based systems), because it cannot be executed on DB2 Version 8. On Windows the pre-migration tool is db2ckmig. On UNIX systems, db2imigr performs similar tasks. Backup all databases prior to migration, and prior to DB2 Version 8 installation on Windows operating systems.

**Authorization:**

*sysadm*

**Required connection:**

This command establishes a database connection.

**Command syntax:**

```
►►──MIGRATE──┬─DATABASE─┬──database-alias────────────────────────────────►
             └─DB───────┘

►──┬────────────────────────────────────┬─────────────────────────────►◄
   └─USER──username──┬──────────────────┤
                     └─USING──password──┘
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database to be migrated to the currently installed version of the database manager.

**USER username**
Identifies the user name under which the database is to be migrated.

**USING password**
The password used to authenticate the user name. If the password is omitted, but a user name was specified, the user is prompted to enter it.

**Examples:**

The following example migrates the database cataloged under the database alias `sales`:

```
db2 migrate database sales
```

**Usage notes:**

This command will only migrate a database to a newer version, and cannot be used to convert a migrated database to its previous version.

The database must be cataloged before migration.

If an error occurs during migration, it might be necessary to issue the TERMINATE command before attempting the suggested user response. For example, if a log full error occurs during migration (SQL1704: Database migration failed. Reason code

″3″.), it will be necessary to issue the TERMINATE command before increasing the values of the database configuration parameters LOGPRIMARY and LOGFILSIZ. The CLP must refresh its database directory cache if the migration failure occurs after the database has already been relocated (which is likely to be the case when a ″log full″ error returns).

When a database is migrated to Version 8, a detailed deadlocks event monitor is created. As with any monitor, there is some overhead associated with this event monitor. You can drop the deadlocks event monitor by issuing the DROP EVENT MONITOR command.

**Related reference:**

- "TERMINATE" on page 705

# PING

Tests the network response time of the underlying connectivity between a client and a connected database server.

**Authorization:**

None

**Required connection:**

Database

**Command syntax:**

```
                   ┌─REQUEST─packet_size─┐  ┌─RESPONSE─packet_size─┐
►►──PING──db_alias─┴─────────────────────┴──┴──────────────────────┴──────►

        ┌─TIME─┐
   ┌─1──┴──────┴─────────┐
►──┼─────────────────────┼─────────────────────────────────────────────►◄
   └─number_of_times─┬───────┤
                     ├─TIMES─┤
                     └─TIME──┘
```

**Command parameters:**

**db_alias**
Specifies the database alias for the database on a DRDA server that the ping is being sent to.

**Note:** This parameter, although mandatory, is not currently used. It is reserved for future use. Any valid database alias name can be specified.

**REQUEST** *packet_size*
Specifies the size, in bytes, of the packet to be sent to the server. The size must be between 0 and 32767 inclusive. The default is 10 bytes. This option is only valid on servers running DB2 Universal Database (UDB) for Linux, UNIX, and Windows Version 8 or later, or DB2 UDB for z/OS Version 8 or later.

**RESPONSE** *packet_size*
Specifies the size, in bytes, of the packet to be returned back to client. The size must be between 0 and 32767 inclusive. The default is 10 bytes. This option is only valid on servers running DB2 Universal Database (UDB) for Linux, UNIX, and Windows Version 8 or later, or DB2 UDB for z/OS Version 8 or later.

*number_of_times*
Specifies the number of iterations for this test. The value must be between 1 and 32767 inclusive. The default is 1. One timing will be returned for each iteration.

**Examples:**

Example 1

To test the network response time for the connection to the host database *hostdb* once:

```
   db2 ping hostdb 1
or
   db2 ping hostdb
```

The command will display output that looks like this:

```
   Elapsed time: 7221 microseconds
```

Example 2

To test the network response time for the connection to the host database *hostdb* 5 times:

```
   db2 ping hostdb 5
or
   db2 ping hostdb 5 times
```

The command will display output that looks like this:

```
   Elapsed time: 8412 microseconds
   Elapsed time: 11876 microseconds
   Elapsed time: 7789 microseconds
   Elapsed time: 10124 microseconds
   Elapsed time: 10988 microseconds
```

Example 3

To test the network response time for a connection to the host database *hostdb*, with a 100-byte request packet and a 200-byte response packet:

```
   db2 ping hostdb request 100 response 200
or
   db2 ping hostdb request 100 response 200 1 time
```

**Usage notes:**

A database connection must exist before invoking this command, otherwise an error will result.

The elapsed time returned is for the connection between the DB2 client and the DB2 server.

# PRECOMPILE

Processes an application program source file containing embedded SQL statements. A modified source file is produced, containing host language calls for the SQL statements and, by default, a package is created in the database.

**Scope:**

This command can be issued from any database partition in `db2nodes.cfg`. In a partitioned database environment, it can be issued from any database partition server defined in the db2nodes.cfg file. It updates the database catalogs on the catalog database partition. Its effects are visible to all database partitions.

**Authorization:**

One of the following:

- *sysadm* or *dbadm* authority
- BINDADD privilege if a package does not exist, and one of:
  - – IMPLICIT_SCHEMA authority on the database if the schema name of the package does not exist
  - – CREATEIN privilege on the schema if the schema name of the package exists
- ALTERIN privilege on the schema if the package exists
- BIND privilege on the package if it exists.

The user also needs all privileges required to compile any static SQL statements in the application. Privileges granted to groups are not used for authorization checking of static statements. If the user has *sysadm* authority, but not explicit privileges to complete the bind, the database manager grants explicit *dbadm* authority automatically.

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

**For DB2 for Windows and UNIX**

```
►►──┬─PRECOMPILE─┬──filename──────────────────────────────────────────────────────►
    └─PREP───────┘

►──┬────────────────────────────────────────────────────────────────────────┬─────►
   └─ACTION───┬─ADD────────────────────────────────────────────────────────┤
              │ ┌─REPLACE─┐                                              │
              └─┴─────────┴──┬───────────────────────┬──┬─────────────────────┬─┘
                             └─RETAIN───┬─NO──┬──┘  └─REPLVER──version-id─┘
                                        └─YES─┘
```

```
►─┬──────────────────────────────┬─┬──────────────────────────┬──────────────────►
  └─BINDFILE─┬─────────────────┬─┘ └─BLOCKING──┬─UNAMBIG─┬─┘
             └─USING──bind-file─┘              ├─ALL─────┤
                                               └─NO──────┘

►─┬──────────────────────────┬─┬──────────────────────────────────┬──────────────►
  └─COLLECTION──schema-name─┘ └─CALL_RESOLUTION──┬─IMMEDIATE─┬─┘
                                                  └─DEFERRED──┘

►─┬─────────────────┬─┬──────────────────┬─┬──────────────────────────┬──────────►
  └─CONNECT──┬─1─┬─┘ └─DATETIME──┬─DEF─┬─┘ └─DEFERRED_PREPARE──┬─NO──┬─┘
             └─2─┘               ├─EUR─┤                      ├─ALL─┤
                                 ├─ISO─┤                      └─YES─┘
                                 ├─JIS─┤
                                 ├─LOC─┤
                                 └─USA─┘

►─┬────────────────────────────────────┬─┬──────────────────────────────┬────────►
  └─DEGREE──┬─1────────────────────┬─┘ └─DISCONNECT──┬─EXPLICIT────┬─┘
            ├─degree-of-parallelism─┤                 ├─AUTOMATIC───┤
            └─ANY───────────────────┘                 └─CONDITIONAL─┘

►─┬─────────────────────────────┬─┬─────────────────────┬────────────────────────►
  └─DYNAMICRULES──┬─RUN────────┬─┘ └─EXPLAIN──┬─NO─────┬─┘
                  ├─BIND───────┤              ├─ALL────┤
                  ├─INVOKERUN──┤              ├─REOPT──┤
                  ├─INVOKEBIND─┤              └─YES────┘
                  ├─DEFINERUN──┤
                  └─DEFINEBIND─┘

►─┬──────────────────────┬─┬──────────────────────┬─┬─────────────────────────────┬─►
  └─EXPLSNAP──┬─NO────┬─┘ └─FEDERATED──┬─NO──┬─┘ │          ┌─,───────────┐  │
              ├─ALL───┤                 └─YES─┘   │          ▼             │  │
              ├─REOPT─┤                           └─FUNCPATH───schema-name─┴──┘
              └─YES───┘

►─┬───────────────────┬─┬───────────────────┬─┬────────────────────┬─────────────►
  └─GENERIC──string─┘ └─INSERT──┬─DEF─┬─┘ └─ISOLATION──┬─CS─┬─┘
                                └─BUF─┘                ├─RR─┤
                                                       ├─RS─┤
                                                       └─UR─┘

►─┬───────────────────────────┬─┬─────────────────────────────┬──────────────────►
  └─LANGLEVEL──┬─SAA1───┬─┘ └─LEVEL──consistency token─┘
               ├─MIA────┤
               └─SQL92E─┘

►─┬─────────────────────────────────┬─┬──────────────────────────┬─┬──────────────┬─►
  │ (1)                             │ └─MESSAGES──message-file─┘ └─NOLINEMACRO─┘
  └──────────LONGERROR──┬─NO──┬─┘
                        └─YES─┘

►─┬─────────────────┬─┬────────────────────┬─┬──────────────────────────┬────────►
  └─OPTLEVEL──┬─0─┬─┘ └─OUTPUT──filename─┘ └─OWNER──authorization-id─┘
              └─1─┘
```

```
►──┬──────────────────────────────────┬──────────────────────────────────────────────►
   └─PACKAGE──┬─────────────────────┬─┘
              └─USING──package-name─┘

►──┬──────────────────────────────────────────────────┬──┬───────────────────────────────┬──►
   └─PREPROCESSOR──┬─"preprocessor-command"─┬─────────┘  └─QUALIFIER──qualifier-name─────┘
                   └─'preprocessor-command'─┘

                                     ┌─REOPT NONE───┐
►──┬──────────────────────────────┬──┼──────────────┼──┬───────────────────────┬──────►
   └─QUERYOPT──optimization-level─┘  ├─REOPT ONCE───┤  └─SQLCA──┬─NONE─┬──────┘
                                     └─REOPT ALWAYS─┘           └─SAA──┘

►──┬──────────────────────────────────┬──┬──────────────────────────────────────┬──────►
   │  (2)                             │  └─SQLFLAG──┬─SQL92E────┬──SYNTAX───────┘
   └──────────SQLERROR──┬─NOPACKAGE─┬─┘             ├─MVSDB2V23─┤
                        ├─CHECK─────┤               ├─MVSDB2V31─┤
                        └─CONTINUE──┘               └─MVSDB2V41─┘

►──┬──────────────────────┬──┬───────────────────────┬──┬─────────────────────────────┬──►
   └─SQLRULES──┬─DB2─┬────┘  └─SQLWARN──┬─NO──┬─────┘  │                   ┌─NO──┐   │
               └─STD─┘                  └─YES─┘        └─STATICREADONLY────┴─YES─┴───┘

►──┬────────────────────────────┬──┬──────────┬──┬────────────────────────────┬──────►
   └─SYNCPOINT──┬─ONEPHASE─┬────┘  └─SYNTAX───┘  └─TARGET──┬─IBMCOB─────┬─────┘
                ├─NONE─────┤                               ├─MFCOB──────┤
                └─TWOPHASE─┘                               ├─ANSI_COBOL─┤
                                                           ├─C──────────┤
                                                           ├─CPLUSPLUS──┤
                                                           └─FORTRAN────┘

►──┬─────────────────────────────┬──┬───────────────────────┬──────────────────────►
   └─TRANSFORM GROUP──groupname──┘  └─VALIDATE──┬─BIND─┬────┘
                                                └─RUN──┘

►──┬──────────────────────────────┬──┬─────────────────────────────┬──────────────►◄
   └─WCHARTYPE──┬─NOCONVERT─┬─────┘  └─VERSION──┬─version-id─┬────┘
                └─CONVERT───┘                   └─AUTO───────┘
```

**Notes:**

1 NO is the default for 32 bit systems and for 64 bit NT systems where long host variables can be used as declarations for INTEGER columns. YES is the default for 64 bit UNIX systems.

2 SYNTAX is a synonym for SQLERROR(CHECK).

### For DB2 on servers other than Windows and UNIX

```
►►──┬─PRECOMPILE─┬──filename──────────────────────────────────────────────────────►
    └─PREP───────┘
```

```
►──┬──────────────────────────────────────────────────────────────────────────┬──►
   └─ACTION──┬─ADD─────────────────────────────────────────────────────────────┘
             │  ┌─REPLACE─┐                                               │
             └──┴─────────┴──┬───────────────────────┬──┬───────────────────────┬──┘
                             │          ┌─YES─┐      │  └─REPLVER──version-id─┘
                             └─RETAIN──┴─NO──┴─┘

►──┬──────────────────────────────────┬──┬──────────────────────────────┬──►
   └─BINDFILE──┬──────────────────┬─┘  └─BLOCKING──┬─UNAMBIG─┬─┘
               └─USING──bind-file─┘                 ├─ALL─────┤
                                                    └─NO──────┘

►──┬──────────────────────────────────────┬──┬────────────────────────┬──►
   └─CALL_RESOLUTION──┬─IMMEDIATE─┬─┘  └─CCSIDG──double-ccsid─┘
                      └─DEFERRED──┘

►──┬──────────────────────┬──┬─────────────────────┬──┬──────────────────────────┬──►
   └─CCSIDM──mixed-ccsid─┘  └─CCSIDS──sbcs-ccsid─┘  │           ┌─DEFAULT─┐  │
                                                     └─CHARSUB──┼─BIT─────┼─┘
                                                                ├─MIXED───┤
                                                                └─SBCS────┘

►──┬──────────────────────┬──┬──────────────────────────────┬──┬─COMPILE────┬──►
   │            ┌─YES─┐   │  └─COLLECTION──schema-name─┘  └─PRECOMPILE─┘
   └─CNULREQD──┴─NO──┴─┘

►──┬──────────────────┬──┬─────────────────────────────┬──┬───────────────────────────────┬──►
   │          ┌─1─┐   │  │ (1)                         │  └─DBPROTOCOL──┬─DRDA────┬─┘
   └─CONNECT──┴─2─┴─┘  └──────────DATETIME──┬─DEF─┬─┘                  └─PRIVATE─┘
                                              ├─EUR─┤
                                              ├─ISO─┤
                                              ├─JIS─┤
                                              ├─LOC─┤
                                              └─USA─┘

►──┬──────────────┬──┬─────────────────────────┬──┬────────────────────────────────────┬──►
   └─DEC──┬─15─┬─┘  │           ┌─PERIOD─┐   │  │                      ┌─NO──┐       │
          └─31─┘     └─DECDEL──┴─COMMA──┴─┘  └─DEFERRED_PREPARE──┼─ALL─┼─┘
                                                                  └─YES─┘

►──┬───────────────────────────────────────────────┬──┬──────────────────────────────────┬──►
   │ (2)              ┌─1─────────────────────┐   │  └─DISCONNECT──┬─EXPLICIT────┬─┘
   └───────────DEGREE──┼─degree-of-parallelism─┼─┘                 ├─AUTOMATIC───┤
                       └─ANY───────────────────┘                   └─CONDITIONAL─┘

►──┬─────────────────────────────────┬──┬──────────────────────────────┬──►
   │                 ┌─RUN────────┐  │  └─ENCODING──┬─ASCII───┬─┘
   └─DYNAMICRULES──┼─BIND───────┼─┘               ├─EBCDIC──┤
                   ├─INVOKERUN──┤                  ├─UNICODE─┤
                   ├─INVOKEBIND─┤                  └─CCSID───┘
                   ├─DEFINERUN──┤
                   └─DEFINEBIND─┘

►──┬─────────────────────┬──┬──────────────────────┬──┬────────────────────────┬──►
   │          ┌─NO──┐    │  └─GENERIC──string─┘  └─IMMEDWRITE──┬─NO──┬─┘
   └─EXPLAIN──┴─YES─┴─┘                                        ├─YES─┤
                                                               └─PH1─┘
```

```
►─┬────────────────────┬─┬─────────────────────────┬─┬──────────────────────────┬──────────────►
  │            ┌─CS─┐  │ └─KEEPDYNAMIC──┬─YES─┬─┘ └─LEVEL──consistency-token─┘
  └─ISOLATION──┼─NC─┼──┘                └─NO──┘
               ├─RR─┤
               ├─RS─┤
               └─UR─┘

►─┬─────────────────────────────┬─┬─────────────────────────┬─┬─────────────┬───────────────────►
  │ (3)              ┌─NO──┐    │ └─MESSAGES──message-file─┘ └─NOLINEMACRO─┘
  └──────LONGERROR───┴─YES─┴────┘

►─┬──────────────────────┬─┬──────────────────┬─┬─────────────────────────────┬──────────────────►
  └─OPTHINT────hint-id───┘ │          ┌─0─┐   │ └─OS400NAMING───┬─SYSTEM─┬─┘
                           └─OPTLEVEL─┴─1─┴───┘                  └─SQL────┘

►─┬──────────────────────────────┬─┬─────────────────────────────────────────────────┬──────────►
  └─OWNER──authorization-id──────┘ └─PREPROCESSOR──┬─"preprocessor-command"─┬─┘
                                                   └─'preprocessor-command'─┘

                                                          ┌─REOPT NONE────┐
►─┬──────────────────────────────┬─┬───────────────────────┬─┼───────────────┼──────────────────►
  └─QUALIFIER──qualifier-name────┘ │         ┌─COMMIT─────┐  │ ├─REOPT ONCE────┤
                                   └─RELEASE─┴─DEALLOCATE─┴──┘ └─REOPT ALWAYS──┘

►─┬─────────────────┬─┬──────────────────────────────────┬─┬─────────────────────────┬───────────►
  ├─REOPT VARS──────┤ └─SQLFLAG──┬─SQL92E─────┬──SYNTAX──┘ └─SORTSEQ───┬─JOBRUN─┬──┘
  └─NOREOPT VARS────┘            ├─MVSDB2V23──┤                       └─HEX────┘
                                 ├─MVSDB2V31──┤
                                 └─MVSDB2V41──┘

►─┬──────────────────────────┬─┬─────────────────────────────┬─┬─────────────────────────────┬──►
  │              ┌─DB2─┐     │ │          ┌─NOPACKAGE─┐      │ │          ┌─APOSTROPHE─┐     │
  └─SQLRULES─────┴─STD─┴─────┘ └─SQLERROR─┼─CHECK─────┼──────┘ └─STRDEL───┴─QUOTE──────┴─────┘
                                          └─CONTINUE──┘

►─┬──────────────────────────────┬─┬──────────┬─┬─────────────────────────────────┬─────────────►
  │               ┌─ONEPHASE─┐   │ └─SYNTAX───┘ │         ┌─IBMCOB────────────┐   │
  └─SYNCPOINT─────┼─NONE─────┼───┘              └─TARGET──┼─MFCOB─────────────┼───┘
                  └─TWOPHASE─┘                            ├─ANSI_COBOL────────┤
                                                          ├─C─────────────────┤
                                                          ├─CPLUSPLUS─────────┤
                                                          ├─FORTRAN───────────┤
                                                          ├─BORLAND_C─────────┤
                                                          └─BORLAND_CPLUSPLUS─┘

►─┬──────────────┬─┬─────────────────────────────┬─┬─────────────────────────┬──────────────────►
  └─TEXT──label──┘ └─VERSION────┬─version-id─┬───┘ └─VALIDATE────┬─BIND─┬───┘
                                └─AUTO───────┘                   └─RUN──┘

►─┬──────────────────────────────┬─────────────────────────────────────────────────────────────►◄
  │              ┌─NOCONVERT─┐   │
  └─WCHARTYPE────┴─CONVERT───┴───┘
```

**Notes:**

1 If the server does not support the DATETIME DEF option, it is mapped to DATETIME ISO.

2 The DEGREE option is only supported by DRDA Level 2 Application Servers.

3 NO is the default for 32 bit systems and for 64 bit NT systems where long host variables can be used as declarations for INTEGER columns. YES is the default for 64 bit UNIX systems.

**Command parameters:**

**filename**

Specifies the source file to be precompiled. An extension of:

- `.sqc` must be specified for C applications (generates a `.c` file)
- `.sqx` (Windows operating systems), or `.sqC` (UNIX based systems) must be specified for C++ applications (generates a `.cxx` file on Windows operating systems, or a `.C` file on UNIX based systems)
- `.sqb` must be specified for COBOL applications (generates a `.cbl` file)
- `.sqf` must be specified for FORTRAN applications (generates a `.for` file on Windows operating systems, or a `.f` file on UNIX based systems).

The preferred extension for C++ applications containing embedded SQL on UNIX based systems is `sqC`; however, the `sqx` convention, which was invented for systems that are not case sensitive, is tolerated by UNIX based systems.

**ACTION**

Indicates whether the package can be added or replaced.

**ADD** Indicates that the named package does not exist, and that a new package is to be created. If the package already exists, execution stops, and a diagnostic error message is returned.

**REPLACE**

Indicates that the existing package is to be replaced by a new one with the same package name and creator. This is the default value for the ACTION option.

**RETAIN**

Indicates whether EXECUTE authorities are to be preserved when a package is replaced. If ownership of the package changes, the new owner grants the BIND and EXECUTE authority to the previous package owner.

**NO** Does not preserve EXECUTE authorities when a package is replaced. This value is not supported by DB2.

**YES** Preserves EXECUTE authorities when a package is replaced. This is the default value.

**REPLVER version-id**

Replaces a specific version of a package. The version identifier specifies which version of the package is to be replaced. If the specified version does not exist, an error is returned. If the REPLVER option of REPLACE is not specified, and a package already exists that matches the package name and version of the package being precompiled, that package will be replaced; if not, a new package will be added.

**BINDFILE**

Results in the creation of a bind file. A package is not created unless the **package** option is also specified. If a bind file is requested, but no package is to be created, as in the following example:

```
db2 prep sample.sqc bindfile
```

object existence and authentication SQLCODEs will be treated as warnings instead of errors. This will allow a bind file to be successfully created, even if the database being used for precompilation does not have all of the objects referred to in static SQL statements within the application. The bind file can be successfully bound, creating a package, once the required objects have been created.

**USING bind-file**

The name of the bind file that is to be generated by the precompiler. The file name must have an extension of `.bnd`. If a file name is not entered, the precompiler uses the name of the program (entered as the *filename* parameter), and adds the `.bnd` extension. If a path is not provided, the bind file is created in the current directory.

**BLOCKING**

Specifies the type of row blocking for cursors.

**ALL** Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as read-only.

**NO** Specifies not to block any cursors. Ambiguous cursors are treated as updatable.

**UNAMBIG**

Specifies to block for:

- Read-only cursors
- Cursors not specified as FOR UPDATE OF

Ambiguous cursors are treated as updatable.

**CALL_RESOLUTION**

If set, the CALL_RESOLUTION DEFERRED option indicates that the CALL statement will be executed as an invocation of the deprecated sqleproc() API. If not set or if IMMEDIATE is set, the CALL statement will be executed as a normal SQL statement. Note that SQL0204 will be issued if the precompiler fails to resolve the procedure on a CALL statement with CALL_RESOLUTION IMMEDIATE.

**CCSIDG double-ccsid**

An integer specifying the coded character set identifier (CCSID) to be used for double byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**CCSIDM mixed-ccsid**

An integer specifying the coded character set identifier (CCSID) to be used for mixed byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**CCSIDS sbcs-ccsid**
An integer specifying the coded character set identifier (CCSID) to be used for single byte characters in character column definitions (without a specific CCSID clause) in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**CHARSUB**
Designates the default character sub-type that is to be used for column definitions in CREATE and ALTER TABLE SQL statements. This DRDA precompile/bind option is not supported by DB2.

**BIT** Use the FOR BIT DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**DEFAULT**
Use the target system defined default in all new character columns for which an explicit sub-type is not specified.

**MIXED**
Use the FOR MIXED DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**SBCS** Use the FOR SBCS DATA SQL character sub-type in all new character columns for which an explicit sub-type is not specified.

**CNULREQD**
This option is related to the **langlevel** precompile option, which is not supported by DRDA. It is valid only if the bind file is created from a C or a C++ application. This DRDA bind option is not supported by DB2.

**NO** The application was coded on the basis of the **langlevel** SAA1 precompile option with respect to the null terminator in C string host variables.

**YES** The application was coded on the basis of the **langlevel** MIA precompile option with respect to the null terminator in C string host variables.

**COLLECTION schema-name**
Specifies a 30-character collection identifier for the package. If not specified, the authorization identifier for the user processing the package is used.

**CONNECT**

**1** Specifies that a CONNECT statement is to be processed as a type 1 CONNECT.

**2** Specifies that a CONNECT statement is to be processed as a type 2 CONNECT.

**DATETIME**
Specifies the date and time format to be used.

**DEF** Use a date and time format associated with the territory code of the database.

**EUR** Use the IBM standard for Europe date and time format.

**ISO** Use the date and time format of the International Standards Organization.

**JIS** Use the date and time format of the Japanese Industrial Standard.

**LOC** Use the date and time format in local form associated with the territory code of the database.

**USA** Use the IBM standard for U.S. date and time format.

**DBPROTOCOL**
Specifies what protocol to use when connecting to a remote site that is identified by a three-part name statement. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**DEC** Specifies the maximum precision to be used in decimal arithmetic operations. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**15** 15-digit precision is used in decimal arithmetic operations.

**31** 31-digit precision is used in decimal arithmetic operations.

**DECDEL**
Designates whether a period (.) or a comma (,) will be used as the decimal point indicator in decimal and floating point literals. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**COMMA**
Use a comma (,) as the decimal point indicator.

**PERIOD**
Use a period (.) as the decimal point indicator.

**DEFERRED_PREPARE**
Provides a performance enhancement when accessing DB2 common server databases or DRDA databases. This option combines the SQL PREPARE statement flow with the associated OPEN, DESCRIBE, or EXECUTE statement flow to minimize inter-process or network flow.

**NO** The PREPARE statement will be executed at the time it is issued.

**YES** Execution of the PREPARE statement will be deferred until the corresponding OPEN, DESCRIBE, or EXECUTE statement is issued.

The PREPARE statement will not be deferred if it uses the INTO clause, which requires an SQLDA to be returned immediately. However, if the PREPARE INTO statement is issued for a cursor that does not use any parameter markers, the processing will be optimized by pre-OPENing the cursor when the PREPARE is executed.

**ALL** Same as YES, except that a PREPARE INTO statement is also deferred. If the PREPARE statement uses the INTO clause to return an SQLDA, the application must not reference the content of this SQLDA until the OPEN, DESCRIBE, or EXECUTE statement is issued and returned.

**DEGREE**
Specifies the degree of parallelism for the execution of static SQL statements in an SMP system. This option does not affect CREATE INDEX parallelism.

**1** The execution of the statement will not use parallelism.

**degree-of-parallelism**
Specifies the degree of parallelism with which the statement can be executed, a value between 2 and 32 767 (inclusive).

**ANY** Specifies that the execution of the statement can involve parallelism using a degree determined by the database manager.

**DISCONNECT**

**AUTOMATIC**
Specifies that all database connections are to be disconnected at commit.

**CONDITIONAL**
Specifies that the database connections that have been marked RELEASE or have no open WITH HOLD cursors are to be disconnected at commit.

**EXPLICIT**
Specifies that only database connections that have been explicitly marked for release by the RELEASE statement are to be disconnected at commit.

**DYNAMICRULES**
Defines which rules apply to dynamic SQL at run time for the initial setting of the values used for authorization ID and for the implicit qualification of unqualified object references.

**RUN** Specifies that the authorization ID of the user executing the package is to be used for authorization checking of dynamic SQL statements. The authorization ID will also be used as the default package qualifier for implicit qualification of unqualified object references within dynamic SQL statements. This is the default value.

**BIND** Specifies that all of the rules that apply to static SQL for authorization and qualification are to be used at run time. That is, the authorization ID of the package owner is to be used for authorization checking of dynamic SQL statements, and the default package qualifier is to be used for implicit qualification of unqualified object references within dynamic SQL statements.

**DEFINERUN**
If the package is used within a routine context, the authorization ID of the routine definer is to be used for authorization checking and for implicit qualification of unqualified object references within dynamic SQL statements within the routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES RUN.

**DEFINEBIND**
If the package is used within a routine context, the authorization ID of the routine definer is to be used for authorization checking and for implicit qualification of unqualified object references within dynamic SQL statements within the routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES BIND.

**INVOKERUN**

If the package is used within a routine context, the current statement authorization ID in effect when the routine is invoked is to be used for authorization checking of dynamic SQL statements and for implicit qualification of unqualified object references within dynamic SQL statements within that routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES RUN.

**INVOKEBIND**

If the package is used within a routine context, the current statement authorization ID in effect when the routine is invoked is to be used for authorization checking of dynamic SQL statements and for implicit qualification of unqualified object references within dynamic SQL statements within that routine.

If the package is used as a standalone application, dynamic SQL statements are processed as if the package were bound with DYNAMICRULES BIND.

**Note:** Because dynamic SQL statements will be using the authorization ID of the package owner in a package exhibiting bind behavior, the binder of the package should not have any authorities granted to them that the user of the package should not receive. Similarly, when defining a routine that will exhibit define behavior, the definer of the routine should not have any authorities granted to them that the user of the package should not receive since a dynamic statement will be using the authorization ID of the routine's definer.

The following dynamically prepared SQL statements cannot be used within a package that was not bound with DYNAMICRULES RUN: GRANT, REVOKE, ALTER, CREATE, DROP, COMMENT ON, RENAME, SET INTEGRITY, and SET EVENT MONITOR STATE.

**ENCODING**

Specifies the encoding for all host variables in static statements in the plan or package. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**EXPLAIN**

Stores information in the Explain tables about the access plans chosen for each SQL statement in the package. DRDA does not support the ALL value for this option.

**NO** Explain information will not be captured.

**YES** Explain tables will be populated with information about the chosen access plan at prep/bind time for static statements and at run time for incremental bind statements.

If the package is to be used for a routine and the package contains incremental bind statements, then the routine must be defined as

MODIFIES SQL DATA. If this is not done, incremental bind statements in the package will cause a run time error (SQLSTATE 42985).

**REOPT**
Explain information for each reoptimizable incremental bind SQL statement will be placed in the Explain tables at run time. In addition, Explain information will be gathered for reoptimizable dynamic SQL statements at run time, even if the CURRENT EXPLAIN MODE special register is set to NO.

If the package is to be used for a routine, then the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**ALL** Explain information for each eligible static SQL statement will be placed in the Explain tables at prep/bind time. Explain information for each eligible incremental bind SQL statement will be placed in the Explain tables at run time. In addition, Explain information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN MODE special register is set to NO.

If the package is to be used for a routine, then the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**Note:** This value for EXPLAIN is not supported by DRDA.

**EXPLSNAP**
Stores Explain Snapshot information in the Explain tables. This DB2 precompile/bind option is not supported by DRDA.

**NO** An Explain Snapshot will not be captured.

**YES** An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables at prep/bind time for static statements and at run time for incremental bind statements.

If the package is to be used for a routine and the package contains incremental bind statements, then the routine must be defined as MODIFIES SQL DATA or incremental bind statements in the package will cause a run time error (SQLSTATE 42985).

**REOPT**
Explain Snapshot information for each reoptimizable incremental bind SQL statement will be placed in the Explain tables at run time. In addition, Explain Snapshot information will be gathered for reoptimizable dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT special register is set to NO.

If the package is to be used for a routine, then the routine must be defined as MODIFIES SQL DATA, otherwise incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**ALL** An Explain Snapshot for each eligible static SQL statement will be placed in the Explain tables at prep/bind time. Explain Snapshot information for each eligible incremental bind SQL statement will be placed in the Explain tables at run time. In addition, Explain Snapshot information will be gathered for eligible dynamic SQL statements at run time, even if the CURRENT EXPLAIN SNAPSHOT special register is set to `NO`.

If the package is to be used for a routine, then the routine must be defined as MODIFIES SQL DATA, or incremental bind and dynamic statements in the package will cause a run time error (SQLSTATE 42985).

**FEDERATED**

Specifies whether a static SQL statement in a package references a nickname or a federated view. If this option is not specified and a static SQL statement in the package references a nickname or a federated view, a warning is returned and the package is created.

**Note:** This option is not supported by DRDA servers.

**NO** A nickname or federated view is not referenced in the static SQL statements of the package. If a nickname or federated view is encountered in a static SQL statement during the prepare or bind phase of this package, an error is returned and the package is *not* created.

**YES** A nickname or federated view can be referenced in the static SQL statements of the package. If no nicknames or federated views are encountered in static SQL statements during the prepare or bind of the package, no errors or warnings are returned and the package is created.

**FUNCPATH**

Specifies the function path to be used in resolving user-defined distinct types and functions in static SQL. If this option is not specified, the default function path is ″SYSIBM″,″SYSFUN″,USER where USER is the value of the USER special register. This DB2 precompile/bind option is not supported by DRDA.

**schema-name**

An SQL identifier, either ordinary or delimited, which identifies a schema that exists at the application server. No validation that the schema exists is made at precompile or at bind time. The same schema cannot appear more than once in the function path. The number of schemas that can be specified is limited by the length of the resulting function path, which cannot exceed 254 bytes. The schema SYSIBM does not need to be explicitly specified; it is implicitly assumed to be the first schema if it is not included in the function path.

**INSERT**

Allows a program being precompiled or bound against a DB2 Enterprise Server Edition server to request that data inserts be buffered to increase performance.

**BUF** Specifies that inserts from an application should be buffered.

**DEF** Specifies that inserts from an application should not be buffered.

**GENERIC string**

Supports the binding of new options that are defined in the target database, but are not supported by DRDA. Do not use this option to pass bind options that *are* defined in BIND or PRECOMPILE. This option can substantially improve dynamic SQL performance. The syntax is as follows:

```
generic "option1 value1 option2 value2 ..."
```

Each option and value must be separated by one or more blank spaces. For example, if the target DRDA database is DB2 Universal Database, Version 8, one could use:

```
generic "explsnap all queryopt 3 federated yes"
```

to bind each of the EXPLSNAP, QUERYOPT, and FEDERATED options.

The maximum length of the string is 1023 bytes.

**IMMEDWRITE**

Indicates whether immediate writes will be done for updates made to group buffer pool dependent pagesets or partitions. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**ISOLATION**

Determines how far a program bound to this package can be isolated from the effect of other executing programs.

**CS** Specifies Cursor Stability as the isolation level.

**NC** No Commit. Specifies that commitment control is not to be used. This isolation level is not supported by DB2.

**RR** Specifies Repeatable Read as the isolation level.

**RS** Specifies Read Stability as the isolation level. Read Stability ensures that the execution of SQL statements in the package is isolated from other application processes for rows read and changed by the application.

**UR** Specifies Uncommitted Read as the isolation level.

**LANGLEVEL**

Specifies the SQL rules that apply for both the syntax and the semantics for both static and dynamic SQL in the application. This option is not supported by DRDA servers.

**MIA** Select the ISO/ANS SQL92 rules as follows:

- To support error SQLCODE or SQLSTATE checking, an SQLCA must be declared in the application code.
- C null-terminated strings are padded with blanks and always include a null-terminating character, even if truncation occurs.
- The FOR UPDATE clause is optional for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE requires SELECT privilege on the object table of the UPDATE or DELETE statement if a column of the object table is referenced in the search condition or on the right hand side of the assignment clause.
- A column function that can be resolved using an index (for example MIN or MAX) will also check for nulls and return warning SQLSTATE 01003 if there were any nulls.
- An error is returned when a duplicate unique constraint is included in a CREATE or ALTER TABLE statement.

- An error is returned when no privilege is granted and the grantor has no privileges on the object (otherwise a warning is returned).

**SAA1** Select the common IBM DB2 rules as follows:

- To support error SQLCODE or SQLSTATE checking, an SQLCA must be declared in the application code.
- C null-terminated strings are not terminated with a null character if truncation occurs.
- The FOR UPDATE clause is required for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE will not require SELECT privilege on the object table of the UPDATE or DELETE statement unless a fullselect in the statement references the object table.
- A column function that can be resolved using an index (for example MIN or MAX) will not check for nulls and warning SQLSTATE 01003 is not returned.
- A warning is returned and the duplicate unique constraint is ignored.
- An error is returned when no privilege is granted.

**SQL92E**
Defines the ISO/ANS SQL92 rules as follows:

- To support checking of SQLCODE or SQLSTATE values, variables by this name can be declared in the host variable declare section (if neither is declared, SQLCODE is assumed during precompilation).
- C null-terminated strings are padded with blanks and always include a null-terminating character, even if truncation occurs.
- The FOR UPDATE clause is optional for all columns to be updated in a positioned UPDATE.
- A searched UPDATE or DELETE requires SELECT privilege on the object table of the UPDATE or DELETE statement if a column of the object table is referenced in the search condition or on the right hand side of the assignment clause.
- A column function that can be resolved using an index (for example MIN or MAX) will also check for nulls and return warning SQLSTATE 01003 if there were any nulls.
- An error is returned when a duplicate unique constraint is included in a CREATE or ALTER TABLE statement.
- An error is returned when no privilege is granted and the grantor has no privileges on the object (otherwise a warning is returned).

**KEEPDYNAMIC**
Specifies whether dynamic SQL statements are to be kept after commit points. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**LEVEL consistency-token**
Defines the level of a module using the consistency token. The consistency token is any alphanumeric value up to 8 characters in length. The RDB

package consistency token verifies that the requester's application and the relational database package are synchronized.

**Note:** This option is not recommended for general use.

**LONGERROR**
Indicates whether long host variable declarations will be treated as an error. For portability, sqlint32 can be used as a declaration for an INTEGER column in precompiled C and C++ code.

**NO** Does not generate errors for the use of long host variable declarations. This is the default for 32 bit systems and for 64 bit NT systems where long host variables can be used as declarations for INTEGER columns. The use of this option on 64 bit UNIX platforms will allow long host variables to be used as declarations for BIGINT columns.

**YES** Generates errors for the use of long host variable declarations. This is the default for 64 bit UNIX systems.

**MESSAGES message-file**
Specifies the destination for warning, error, and completion status messages. A message file is created whether the bind is successful or not. If a message file name is not specified, the messages are written to standard output. If the complete path to the file is not specified, the current directory is used. If the name of an existing file is specified, the contents of the file are overwritten.

**NOLINEMACRO**
Suppresses the generation of the #line macros in the output `.c` file. Useful when the file is used with development tools which require source line information such as profiles, cross-reference utilities, and debuggers.

**Note:** This precompile option is used for the C/C$^{++}$ programming languages only.

**OPTHINT**
Controls whether query optimization hints are used for static SQL. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**OPTLEVEL**
Indicates whether the C/C++ precompiler is to optimize initialization of internal SQLDAs when host variables are used in SQL statements. Such optimization can increase performance when a single SQL statement (such as FETCH) is used inside a tight loop.

**0** Instructs the precompiler not to optimize SQLDA initialization.

**1** Instructs the precompiler to optimize SQLDA initialization. This value should not be specified if the application uses:

- pointer host variables, as in the following example:

```
exec sql begin declare section;
char (*name)[20];
short *id;
exec sql end declare section;
```

- C++ data members directly in SQL statements.

**OUTPUT filename**
Overrides the default name of the modified source file produced by the compiler. It can include a path.

**OS400NAMING**

Specifies which naming option is to be used when accessing DB2 UDB for iSeries data. Supported by DB2 UDB for iSeries only. For a list of supported option values, refer to the documentation for DB2 for iSeries.

Please note that because of the slashes used as separators, a DB2 utility can still report a syntax error at execution time on certain SQL statements which use the iSeries system naming convention, even though the utility might have been precompiled or bound with the OS400NAMING SYSTEM option. For example, the Command Line Processor will report a syntax error on an SQL CALL statement if the iSeries system naming convention is used, whether or not it has been precompiled or bound using the OS400NAMING SYSTEM option.

**OWNER authorization-id**

Designates a 30-character authorization identifier for the package owner. The owner must have the privileges required to execute the SQL statements contained in the package. Only a user with SYSADM or DBADM authority can specify an authorization identifier other than the user ID. The default value is the primary authorization ID of the precompile/bind process. SYSIBM, SYSCAT, and SYSSTAT are not valid values for this option.

**PACKAGE**

Creates a package. If neither **package**, **bindfile**, nor **syntax** is specified, a package is created in the database by default.

**USING package-name**

The name of the package that is to be generated by the precompiler. If a name is not entered, the name of the application program source file (minus extension and folded to uppercase) is used. Maximum length is 8 characters.

**PREPROCESSOR ″preprocessor-command″**

Specifies the preprocessor command that can be executed by the precompiler before it processes embedded SQL statements. The preprocessor command string (maximum length 1024 bytes) must be enclosed either by double or by single quotation marks.

This option enables the use of macros within the declare section. A valid preprocessor command is one that can be issued from the command line to invoke the preprocessor without specifying a source file. For example,

```
xlc -P -DMYMACRO=0
```

**QUALIFIER qualifier-name**

Provides an 30-character implicit qualifier for unqualified objects contained in the package. The default is the owner's authorization ID, whether or not **owner** is explicitly specified.

**QUERYOPT optimization-level**

Indicates the desired level of optimization for all static SQL statements contained in the package. The default value is 5. The SET CURRENT QUERY OPTIMIZATION statement describes the complete range of optimization levels available. This DB2 precompile/bind option is not supported by DRDA.

**RELEASE**

Indicates whether resources are released at each COMMIT point, or when the application terminates. This DRDA precompile/bind option is not supported by DB2.

**COMMIT**
Release resources at each COMMIT point. Used for dynamic SQL statements.

**DEALLOCATE**
Release resources only when the application terminates.

**REOPT**
Specifies whether to have DB2 optimize an access path using values for host variables, parameter markers, and special registers. Valid values are:

**NONE**
The access path for a given SQL statement containing host variables, parameter markers or special registers will not be optimized using real values for these variables. The default estimates for the these variables will be used instead, and this plan is cached and used subsequently. This is the default behavior.

**ONCE** The access path for a given SQL statement will be optimized using the real values of the host variables, parameter markers or special registers when the query is first executed. This plan is cached and used subsequently.

**ALWAYS**
The access path for a given SQL statement will always be compiled and reoptimized using the values of the host variables, parameter markers or special registers known at each execution time.

**REOPT / NOREOPT VARS**
These options have been replaced by REOPT ALWAYS and REOPT NONE; however, they are still supported for back-level compatibility. Specifies whether to have DB2 determine an access path at run time using values for host variables, parameter markers, and special registers. Supported by DB2 for OS/390 only. For a list of supported option values, refer to the documentation for DB2 for OS/390.

**SQLCA**
For FORTRAN applications only. This option is ignored if it is used with other languages.

**NONE**
Specifies that the modified source code is not consistent with the SAA definition.

**SAA** Specifies that the modified source code is consistent with the SAA definition.

**SQLERROR**
Indicates whether to create a package or a bind file if an error is encountered.

**CHECK**
Specifies that the target system performs all syntax and semantic checks on the SQL statements being bound. A package will not be created as part of this process. If, while binding, an existing package with the same name and version is encountered, the existing package is neither dropped nor replaced even if **action replace** was specified.

**CONTINUE**
Creates a package, even if errors occur when binding SQL statements. Those statements that failed to bind for authorization

or existence reasons can be incrementally bound at execution time if VALIDATE RUN is also specified. Any attempt to execute them at run time generates an error (SQLCODE -525, SQLSTATE 51015).

**NOPACKAGE**

A package or a bind file is not created if an error is encountered.

**SQLFLAG**

Identifies and reports on deviations from the SQL language syntax specified in this option.

A bind file or a package is created only if the **bindfile** or the **package** option is specified, in addition to the **sqlflag** option.

Local syntax checking is performed only if one of the following options is specified:

- **bindfile**
- **package**
- **sqlerror check**
- **syntax**

If **sqlflag** is not specified, the flagger function is not invoked, and the bind file or the package is not affected.

**SQL92E SYNTAX**

The SQL statements will be checked against ANSI or ISO SQL92 Entry level SQL language format and syntax with the exception of syntax rules that would require access to the database catalog. Any deviation is reported in the precompiler listing.

**MVSDB2V23 SYNTAX**

The SQL statements will be checked against MVS DB2 Version 2.3 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

**MVSDB2V31 SYNTAX**

The SQL statements will be checked against MVS DB2 Version 3.1 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

**MVSDB2V41 SYNTAX**

The SQL statements will be checked against MVS DB2 Version 4.1 SQL language syntax. Any deviation from the syntax is reported in the precompiler listing.

**SORTSEQ**

Specifies which sort sequence table to use on the iSeries system. Supported by DB2 UDB for iSeries only. For a list of supported option values, refer to the documentation for DB2 for iSeries.

**SQLRULES**

Specifies:

- Whether type 2 CONNECTs are to be processed according to the DB2 rules or the Standard (STD) rules based on ISO/ANS SQL92.
- How a user or application can specify the format of LOB answer set columns.

**DB2**

- Permits the SQL CONNECT statement to switch the current connection to another established (*dormant*) connection.

- The user or application can specify the format of a LOB column only during the first fetch request.

**STD**

- Permits the SQL CONNECT statement to establish a *new* connection only. The SQL SET CONNECTION statement must be used to switch to a dormant connection.
- The user or application can change the format of a LOB column with each fetch request.

**SQLWARN**

Indicates whether warnings will be returned from the compilation of dynamic SQL statements (via PREPARE or EXECUTE IMMEDIATE), or from describe processing (via PREPARE...INTO or DESCRIBE). This DB2 precompile/bind option is not supported by DRDA.

**NO** Warnings will not be returned from the SQL compiler.

**YES** Warnings will be returned from the SQL compiler.

**Note:** SQLCODE +238 is an exception. It is returned regardless of the **sqlwarn** option value.

**STATICREADONLY**

Determines whether static cursors will be treated as being READ ONLY. This DB2 precompile/bind option is not supported by DRDA.

**NO** All static cursors will take on the attributes as would normally be generated given the statement text and the setting of the LANGLEVEL precompile option.

**YES** Any static cursor that does not contain the FOR UPDATE or FOR READ ONLY clause will be considered READ ONLY.

**STRDEL**

Designates whether an apostrophe (′) or double quotation marks (") will be used as the string delimiter within SQL statements. This DRDA precompile/bind option is not supported by DB2. The DRDA server will use a system defined default value if this option is not specified.

**APOSTROPHE**

Use an apostrophe (′) as the string delimiter.

**QUOTE**

Use double quotation marks (") as the string delimiter.

**SYNCPOINT**

Specifies how commits or rollbacks are to be coordinated among multiple database connections.

**NONE**

Specifies that no Transaction Manager (TM) is to be used to perform a two-phase commit, and does not enforce single updater, multiple reader. A COMMIT is sent to each participating database. The application is responsible for recovery if any of the commits fail.

**ONEPHASE**

Specifies that no TM is to be used to perform a two-phase commit. A one-phase commit is to be used to commit the work done by each database in multiple database transactions.

**TWOPHASE**
Specifies that the TM is required to coordinate two-phase commits among those databases that support this protocol.

**SYNTAX**
Suppresses the creation of a package or a bind file during precompilation. This option can be used to check the validity of the source file without modifying or altering existing packages or bind files. **Syntax** is a synonym for **sqlerror check**.

If **syntax** is used together with the **package** option, **package** is ignored.

**TARGET**
Instructs the precompiler to produce modified code tailored to one of the supported compilers on the current platform.

**IBMCOB**
On AIX, code is generated for the IBM COBOL Set for AIX compiler.

**MFCOB**
Code is generated for the Micro Focus COBOL compiler. This is the default if a **target** value is not specified with the COBOL precompiler on all UNIX platforms and Windows NT.

**ANSI_COBOL**
Code compatible with the ANS X3.23-1985 standard is generated.

**C** Code compatible with the C compilers supported by DB2 on the current platform is generated.

**CPLUSPLUS**
Code compatible with the C++ compilers supported by DB2 on the current platform is generated.

**FORTRAN**
Code compatible with the FORTRAN compilers supported by DB2 on the current platform is generated.

**TEXT label**
The description of a package. Maximum length is 255 characters. The default value is blanks. This DRDA precompile/bind option is not supported by DB2.

**TRANSFORM GROUP**
Specifies the transform group name to be used by static SQL statements for exchanging user-defined structured type values with host programs. This transform group is not used for dynamic SQL statements or for the exchange of parameters and results with external functions or methods. This option is not supported by DRDA servers.

**groupname**
An SQL identifier of up to 18 characters in length. A group name cannot include a qualifier prefix and cannot begin with the prefix SYS since this is reserved for database use. In a static SQL statement that interacts with host variables, the name of the transform group to be used for exchanging values of a structured type is as follows:

- The group name in the TRANSFORM GROUP bind option, if any

- The group name in the TRANSFORM GROUP prep option as specified at the original precompilation time, if any
- The DB2_PROGRAM group, if a transform exists for the given type whose group name is DB2_PROGRAM
- No transform group is used if none of the above conditions exist.

The following errors are possible during the bind of a static SQL statement:

- SQLCODE yyy, SQLSTATE xxxxx: A transform is needed, but no static transform group has been selected.
- SQLCODE yyy, SQLSTATE xxxxx: The selected transform group does not include a necessary transform (TO SQL for input variables, FROM SQL for output variables) for the data type that needs to be exchanged.
- SQLCODE yyy, SQLSTATE xxxxx: The result type of the FROM SQL transform is not compatible with the type of the output variable, or the parameter type of the TO SQL transform is not compatible with the type of the input variable.

In these error messages, *yyyyy* is replaced by the SQL error code, and *xxxxx* by the SQL state code.

**VALIDATE**

Determines when the database manager checks for authorization errors and object not found errors. The package owner authorization ID is used for validity checking.

**BIND** Validation is performed at precompile/bind time. If all objects do not exist, or all authority is not held, error messages are produced. If **sqlerror continue** is specified, a package/bind file is produced despite the error message, but the statements in error are not executable.

**RUN** Validation is attempted at bind time. If all objects exist, and all authority is held, no further checking is performed at execution time.

If all objects do not exist, or all authority is not held at precompile/bind time, warning messages are produced, and the package is successfully bound, regardless of the **sqlerror continue** option setting. However, authority checking and existence checking for SQL statements that failed these checks during the precompile/bind process can be redone at execution time.

**VERSION**

Defines the version identifier for a package. If this option is not specified, the package version will be "" (the empty string).

**version-id**

Specifies a version identifier that is any alphanumeric value, $, #, @, _, -, or ., up to 64 characters in length.

**AUTO**

The version identifier will be generated from the consistency token. If the consistency token is a timestamp (it will be if the LEVEL option is not specified), the timestamp is converted into ISO character format and is used as the version identifier.

**WCHARTYPE**

Specifies the format for graphic data.

**CONVERT**

Host variables declared using the wchar_t base type will be treated as containing data in wchar_t format. Since this format is not directly compatible with the format of graphic data stored in the database (DBCS format), input data in wchar_t host variables is implicitly converted to DBCS format on behalf of the application, using the ANSI C function `wcstombs()`. Similarly, output DBCS data is implicitly converted to wchar_t format, using `mbstowcs()`, before being stored in host variables.

**NOCONVERT**

Host variables declared using the wchar_t base type will be treated as containing data in DBCS format. This is the format used within the database for graphic data; it is, however, different from the native wchar_t format implemented in the C language. Using NOCONVERT means that graphic data will not undergo conversion between the application and the database, which can improve efficiency. The application is, however, responsible for ensuring that data in wchar_t format is not passed to the database manager. When this option is used, wchar_t host variables should not be manipulated with the C wide character string functions, and should not be initialized with wide character literals (*L-literals*).

**Usage notes:**

A modified source file is produced, which contains host language equivalents to the SQL statements. By default, a package is created in the database to which a connection has been established. The name of the package is the same as the file name (minus the extension and folded to uppercase), up to a maximum of 8 characters.

Following connection to a database, PREP executes under the transaction that was started. PREP then issues a COMMIT or a ROLLBACK to terminate the current transaction and start another one.

Creating a package with a schema name that does not already exist results in the implicit creation of that schema. The schema owner is SYSIBM. The CREATEIN privilege on the schema is granted to PUBLIC.

During precompilation, an Explain Snapshot is not taken unless a package is created and **explsnap** has been specified. The snapshot is put into the Explain tables of the user creating the package. Similarly, Explain table information is only captured when **explain** is specified, and a package is created.

Precompiling stops if a fatal error or more than 100 errors occur. If a fatal error occurs, the utility stops precompiling, attempts to close all files, and discards the package.

When a package exhibits bind behavior, the following will be true:

1. The implicit or explicit value of the BIND option OWNER will be used for authorization checking of dynamic SQL statements.

2. The implicit or explicit value of the BIND option QUALIFIER will be used as the implicit qualifier for qualification of unqualified objects within dynamic SQL statements.
3. The value of the special register CURRENT SCHEMA has no effect on qualification.

In the event that multiple packages are referenced during a single connection, all dynamic SQL statements prepared by those packages will exhibit the behavior as specified by the DYNAMICRULES option for that specific package and the environment they are used in.

If an SQL statement was found to be in error and the PRECOMPILE option SQLERROR CONTINUE was specified, the statement will be marked as invalid and another PRECOMPILE must be issued in order to change the state of the SQL statement. Implicit and explicit rebind will not change the state of an invalid statement in a package bound with VALIDATE RUN. A statement can change from static to incremental bind or incremental bind to static across implicit and explicit rebinds depending on whether or not object existence or authority problems exist during the rebind.

Binding a package with REOPT ONCE or REOPT ALWAYS might change static and dynamic statement compilation and performance.

**Related concepts:**

- "Authorization Considerations for Dynamic SQL" in the *Application Development Guide: Programming Client Applications*
- "WCHARTYPE Precompiler Option in C and C++" in the *Application Development Guide: Programming Client Applications*
- "Effect of DYNAMICRULES bind option on dynamic SQL" in the *Application Development Guide: Programming Client Applications*
- "Effects of REOPT on static SQL" in the *Application Development Guide: Programming Client Applications*
- "Effects of REOPT on dynamic SQL" in the *Application Development Guide: Programming Client Applications*

**Related tasks:**

- "Specifying row blocking to reduce overhead" in the *Administration Guide: Performance*

**Related reference:**

- "SET CURRENT QUERY OPTIMIZATION statement" in the *SQL Reference, Volume 2*
- "BIND" on page 286
- "Datetime values" in the *SQL Reference, Volume 1*

# PRUNE HISTORY/LOGFILE

Used to delete entries from the recovery history file, or to delete log files from the active log file path. Deleting entries from the recovery history file might be necessary if the file becomes excessively large and the retention period is high.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

Database

**Command syntax:**

```
►►──PRUNE───┬─HISTORY──timestamp──┬───────────────────┬──┬────────────┬─┬──────────►◄
            │                     └─WITH FORCE OPTION─┘  └─AND DELETE─┘ │
            └─LOGFILE PRIOR TO──log-file-name───────────────────────────┘
```

**Command parameters:**

**HISTORY timestamp**
: Identifies a range of entries in the recovery history file that will be deleted. A complete time stamp (in the form *yyyymmddhhmmss*), or an initial prefix (minimum *yyyy*) can be specified. All entries with time stamps equal to or less than the time stamp provided are deleted from the recovery history file.

**WITH FORCE OPTION**
: Specifies that the entries will be pruned according to the time stamp specified, even if some entries from the most recent restore set are deleted from the file. A restore set is the most recent full database backup including any restores of that backup image. If this parameter is not specified, all entries from the backup image forward will be maintained in the history.

**AND DELETE**
: Specifies that the associated log archives will be physically deleted (based on the location information) when the history file entry is removed. This option is especially useful for ensuring that archive storage space is recovered when log archives are no longer needed.

**Note:** If you are archiving logs via a user exit program, the logs cannot be deleted using this option.

**LOGFILE PRIOR TO log-file-name**
: Specifies a string for a log file name, for example `S0000100.LOG`. All log files prior to (but not including) the specified log file will be deleted. The LOGRETAIN database configuration parameter must be set to `RECOVERY` or `CAPTURE`.

**Examples:**

To remove the entries for all restores, loads, table space backups, and full database backups taken before and including December 1, 1994 from the recovery history file, enter:

```
db2 prune history 199412
```

**Note:** 199412 is interpreted as 19941201000000.

**Usage notes:**

If the FORCE option is used, you might delete entries that are required for automatic incremental restoration of databases. Manual restores will still work correctly. Use of this command can also prevent the **dbckrst** utility from being able to correctly analyze the complete chain of required backup images. Using the PRUNE HISTORY command without the FORCE option prevents required entries from being deleted.

Pruning backup entries from the history file causes related file backups on DB2 Data Links Manager servers to be deleted.

# PUT ROUTINE

Uses the specified routine SQL Archive (SAR) file to define a routine in the database.

**Authorization:**

*dbadm*

**Required connection:**

Database. If implicit connect is enabled, a connection to the default database is established.

**Command syntax:**

```
►►──PUT ROUTINE──FROM──file-name──┬─────────────────────────────────────┬──────►◄
                                  └─OWNER──new-owner──┬───────────────┬─┘
                                                      └─USE REGISTERS─┘
```

**Command parameters:**

**FROM file-name**
Names the file where routine SQL archive (SAR) is stored.

**OWNER new-owner**
Specifies a new authorization name that will be used for authorization checking of the routine. The new owner must have the necessary privileges for the routine to be defined. If the OWNER clause is not specified, the authorization name that was originally defined for the routine is used.

**USE REGISTERS**
Indicates that the CURRENT SCHEMA and CURRENT PATH special registers are used to define the routine. If this clause is not specified, the settings for the default schema and SQL path are the settings used when the routine is defined.

**Note:** CURRENT SCHEMA is used as the schema name for unqualified object names in the routine definition (including the name of the routine) and CURRENT PATH is used to resolve unqualified routines and data types in the routine definition.

**Examples:**

```
PUT ROUTINE FROM procs/proc1.sar;
```

**Usage Notes:**

No more than one procedure can be concurrently installed under a given schema.

If a GET ROUTINE or a PUT ROUTINE operation (or their corresponding procedure) fails to execute successfully, it will always return an error (SQLSTATE 38000), along with diagnostic text providing information about the cause of the failure. For example, if the procedure name provided to GET ROUTINE does not identify an SQL procedure, diagnostic ″-204, 42704″ text will be returned, where ″-204″ and ″42704″ are the SQLCODE and SQLSTATE, respectively, that identify the cause of the problem. The SQLCODE and SQLSTATE in this example indicate that the procedure name provided in the GET ROUTINE command is undefined.

# QUERY CLIENT

Returns current connection settings for an application process.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──QUERY CLIENT──────────────────────────────────────────────────────►◄
```

**Command parameters:**
None
**Examples:**
The following is sample output from QUERY CLIENT:

```
The current connection settings of the application process are:
                          CONNECT   = 1
                       DISCONNECT   = EXPLICIT
          MAX_NETBIOS_CONNECTIONS   = 1
                         SQLRULES   = DB2
                        SYNCPOINT   = ONEPHASE
           CONNECT_DBPARTITIONNUM   = CATALOG_DBPARTITIONNUM
            ATTACH_DBPARTITIONNUM   = -1
```

If CONNECT_DBPARTITIONNUM and ATTACH_DBPARTITIONNUM are not set using the SET CLIENT command, these parameters have values identical to that of the environment variable DB2NODE. If the displayed value of the CONNECT_DBPARTITIONNUM or the ATTACH_DBPARTITIONNUM parameter is -1, the parameter has not been set; that is, either the environment variable DB2NODE has not been set, or the parameter was not specified in a previously issued SET CLIENT command.

**Usage notes:**

The connection settings for an application process can be queried at any time during execution.

**Related reference:**

- "SET CLIENT" on page 678

# QUIESCE

Forces all users off the specified instance and database and puts it into a quiesced mode. In quiesced mode, users cannot connect from outside of the database engine. While the database instance or database is in quiesced mode, you can perform administrative tasks on it. After administrative tasks are complete, use the UNQUIESCE command to activate the instance and database and allow other users to connect to the database but avoid having to shut down and perform another database start.

In this mode only users with authority in this restricted mode are allowed to attach or connect to the instance/database. Users with *sysadm*, *sysmaint*, and *sysctrl* authority always have access to an instance while it is quiesced, and users with *sysadm* and *dbadm*authority always have access to a database while it is quiesced.

**Scope:**

QUIESCE DATABASE results in all objects in the database being in the quiesced mode. Only the allowed user/group and *sysadm*, *sysmaint*, *dbadm*, or *sysctrl* will be able to access the database or its objects.

QUIESCE INSTANCE *instance-name* means the instance and the databases in the instance *instance-name* will be in quiesced mode. The instance will be accessible just for *sysadm*, *sysmaint*, and *sysctrl* and allowed user/group.

If an instance is in quiesced mode, a database in the instance cannot be put in quiesced mode.

**Authorization:**

One of the following:

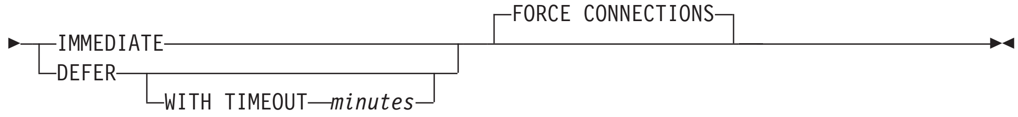
For database level quiesce:
- *sysadm*
- *dbadm*

For instance level quiesce:
- *sysadm*
- *sysctrl*

**Command syntax:**

```
►►──QUIESCE──┬─DATABASE─┬──┬─IMMEDIATE──────────────────────┬──────────────────►
             └─DB───────┘  └─DEFER──┬────────────────────┬──┘
                                    └─WITH TIMEOUT──minutes─┘

  ┌─FORCE CONNECTIONS─┐
►─┴───────────────────┴─────────────────────────────────────────────────────►◄
```

```
►►──QUIESCE INSTANCE──instance-name──┬───────────────────────┬──────────────►
                                     ├─USER──user-name───────┤
                                     └─GROUP──group-name─────┘
```



**Required connection:**
Database

(Database connection is not required for an instance quiesce.)

**Command parameters:**

**DEFER**
Wait for applications until they commit the current unit of work.

**WITH TIMEOUT**
Specifies a time, in minutes, to wait for applications to commit the current unit of work. If no value is specified, in a single-partition database environment, the default value is 10 minutes. In a partitioned database environment the value specified by the *start_stop_timeout* database manager configuration parameter will be used.

**IMMEDIATE**
Do not wait for the transactions to be committed, immediately rollback the transactions.

**FORCE CONNECTIONS**
Force the connections off.

**DATABASE**
Quiesce the database. All objects in the database will be placed in quiesced mode. Only specified users in specified groups and users with *sysadm*, *sysmaint*, and *sysctrl* authority will be able to access to the database or its objects.

**INSTANCE** *instance-name*
The instance *instance-name* and the databases in the instance will be placed in quiesced mode. The instance will be accessible only to users with *sysadm*, *sysmaint*, and *sysctrl* authority and specified users in specified groups.

**USER** *user-name*
Specifies the name of a user who will be allowed access to the instance while it is quiesced.

**GROUP** *group-name*
Specifies the name of a group that will be allowed access to the instance while the instance is quiesced.

**Examples:**

In the following example, the default behavior is to force connections, so it does not need to be explicitly stated and can be removed from this example.

```
db2 quiesce instance crankarm user frank immediate force connections
```

The following example forces off all users with connections to the database.

```
db2 quiesce db immediate
```

- The first example will quiesce the instance `crankarm`, while allowing user `frank` to continue using the database.

The second example will quiesce the database you are attached to, preventing access by all users except those with one of the following authorities: *sysadm*, *sysmaint*, *sysctrl*, or *dbadm*.

- This command will force all users off the database or instance if FORCE CONNECTION option is supplied. FORCE CONNECTION is the default behavior; the parameter is allowed in the command for compatibility reasons.
- The command will be synchronized with the FORCE and will only complete once the FORCE has completed.

**Usage notes:**

- After QUIESCE INSTANCE, only users with *sysadm*, *sysmaint*, or *sysctrl* authority or a user name and group name provided as parameters to the command can connect to the instance.
- After QUIESCE DATABASE, users with *sysadm*, *sysmaint*, *sysctrl*, or *dbadm* authority, and GRANT/REVOKE privileges can designate who will be able to connect. This information will be stored permanently in the database catalog tables.

  For example,

  ```
  grant quiesce_connect on database to <username/groupname>
  revoke quiesce_connect on database from <username/groupname>
  ```

# QUIESCE TABLESPACES FOR TABLE

Quiesces table spaces for a table. There are three valid quiesce modes: share, intent to update, and exclusive. There are three possible states resulting from the quiesce function: QUIESCED SHARE, QUIESCED UPDATE, and QUIESCED EXCLUSIVE.

**Scope:**

In a single-partition environment, this command quiesces all table spaces involved in a load operation in exclusive mode for the duration of the load operation. In a partitioned database environment, this command acts locally on a node. It quiesces only that portion of table spaces belonging to the node on which the load operation is performed.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*
- *load*

**Required connection:**

Database

**Command syntax:**

```
►►──QUIESCE TABLESPACES FOR TABLE──┬─tablename────────┬──┬─SHARE────────────┬──────►◄
                                   └─schema.tablename─┘  ├─INTENT TO UPDATE─┤
                                                         ├─EXCLUSIVE────────┤
                                                         └─RESET────────────┘
```

**Command parameters:**

**TABLE**

**tablename**
Specifies the unqualified table name. The table cannot be a system catalog table.

**schema.tablename**
Specifies the qualified table name. If *schema* is not provided, the CURRENT SCHEMA will be used. The table cannot be a system catalog table.

**SHARE**
Specifies that the quiesce is to be in share mode.

When a ″quiesce share″ request is made, the transaction requests intent share locks for the table spaces and a share lock for the table. When the transaction obtains the locks, the state of the table spaces is changed to QUIESCED SHARE. The state is granted to the quiescer only if there is no conflicting state held by other users. The state of the table spaces, along with the authorization ID and the database agent ID of the quiescer, are recorded in the table space table, so that the state is persistent. The table cannot be changed while the table spaces for the table are in QUIESCED SHARE state. Other share mode requests to the table and table spaces are allowed. When the transaction commits or rolls back, the locks are released, but the table spaces for the table remain in QUIESCED SHARE state until the state is explicitly reset.

**INTENT TO UPDATE**

Specifies that the quiesce is to be in intent to update mode.

When a ″quiesce intent to update″ request is made, the table spaces are locked in intent exclusive (IX) mode, and the table is locked in update (U) mode. The state of the table spaces is recorded in the table space table.

**EXCLUSIVE**

Specifies that the quiesce is to be in exclusive mode.

When a ″quiesce exclusive″ request is made, the transaction requests super exclusive locks on the table spaces, and a super exclusive lock on the table. When the transaction obtains the locks, the state of the table spaces changes to QUIESCED EXCLUSIVE. The state of the table spaces, along with the authorization ID and the database agent ID of the quiescer, are recorded in the table space table. Since the table spaces are held in super exclusive mode, no other access to the table spaces is allowed. The user who invokes the quiesce function (the quiescer) has exclusive access to the table and the table spaces.

**RESET**

Specifies that the state of the table spaces is to be reset to normal.

**Examples:**

```
db2 quiesce tablespaces for table staff share

db2 quiesce tablespaces for table boss.org intent to update
```

**Usage notes:**

This command is not supported for declared temporary tables.

A quiesce is a persistent lock. Its benefit is that it persists across transaction failures, connection failures, and even across system failures (such as power failure, or reboot).

A quiesce is owned by a connection. If the connection is lost, the quiesce remains, but it has no owner, and is called a *phantom quiesce*. For example, if a power outage caused a load operation to be interrupted during the delete phase, the table spaces for the loaded table would be left in delete pending, quiesce exclusive state. Upon database restart, this quiesce would be an unowned (or phantom) quiesce. The removal of a phantom quiesce requires a connection with the same user ID used when the quiesce mode was set.

To remove a phantom quiesce:

1. Connect to the database with the same user ID used when the quiesce mode was set.
2. Use the LIST TABLESPACES command to determine which table space is quiesced.
3. Re-quiesce the table space using the current quiesce state. For example:
   ```
   db2 quiesce tablespaces for table mytable exclusive
   ```

Once completed, the new connection owns the quiesce, and the load operation can be restarted.

There is a limit of five quiescers on a table space at any given time.

A quiescer can upgrade the state of a table space from a less restrictive state to a more restrictive one (for example, S to U, or U to X). If a user requests a state lower than one that is already held, the original state is returned. States are not downgraded.

**Related reference:**

- "LOAD" on page 520

# QUIT

Exits the command line processor interactive input mode and returns to the operating system command prompt. If a batch file is being used to input commands to the command line processor, commands are processed until QUIT, TERMINATE, or the end-of-file is encountered.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──QUIT──────────────────────────────────────────────────────────►◄
```

**Command parameters:**
None

**Usage notes:**

QUIT does not terminate the command line processor back-end process or break a database connection. CONNECT RESET breaks a connection, but does not terminate the back-end process. The TERMINATE command does both.

**Related reference:**

- "TERMINATE" on page 705

# REBIND

Allows the user to recreate a package stored in the database without the need for a bind file.

**Authorization:**

One of the following:
- *sysadm* or *dbadm* authority
- ALTERIN privilege on the schema
- BIND privilege on the package.

The authorization ID logged in the BOUNDBY column of the SYSCAT.PACKAGES system catalog table, which is the ID of the most recent binder of the package, is used as the binder authorization ID for the rebind, and for the default *schema* for table references in the package. Note that this default qualifier can be different from the authorization ID of the user executing the rebind request. REBIND will use the same bind options that were specified when the package was created.

**Required connection:**

Database. If no database connection exists, and if implicit connect is enabled, a connection to the default database is made.

**Command syntax:**

```
►►──REBIND──┬─────────┬──package-name──┬──────────────────────┬──────────────────►
            └─PACKAGE─┘                └─VERSION──version-name─┘

►─RESOLVE──┬─ANY──────────┬──┬───────────────┬──────────────────────────────────►◄
           └─CONSERVATIVE─┘  ├─REOPT NONE────┤
                             ├─REOPT ONCE────┤
                             └─REOPT ALWAYS──┘
```

**Command parameters:**

**PACKAGE package-name**
: The qualified or unqualified name that designates the package to be rebound.

**VERSION version-name**
: The specific version of the package to be rebound. When the version is not specified, it is taken to be "" (the empty string).

**RESOLVE**
: Specifies whether rebinding of the package is to be performed with or without conservative binding semantics. This affects whether new functions and data types are considered during function resolution and type resolution on static DML statements in the package. This option is not supported by DRDA. Valid values are:

**ANY** Any of the functions and types in the SQL path are considered for function and type resolution. Conservative binding semantics are not used. This is the default.

**CONSERVATIVE**
Only functions and types in the SQL path that were defined before

the last explicit bind time stamp are considered for function and type resolution. Conservative binding semantics are used. This option is not supported for an inoperative package.

**REOPT**

Specifies whether to have DB2 optimize an access path using values for host variables, parameter markers, and special registers.

**NONE**

The access path for a given SQL statement containing host variables, parameter markers or special registers will not be optimized using real values for these variables. The default estimates for the these variables will be used instead, and this plan is cached and used subsequently. This is the default behavior.

**ONCE** The access path for a given SQL statement will be optimized using the real values of the host variables, parameter markers or special registers when the query is first executed. This plan is cached and used subsequently.

**ALWAYS**

The access path for a given SQL statement will always be compiled and reoptimized using the values of the host variables, parameter markers or special registers known at each execution time.

**Usage notes:**

REBIND does not automatically commit the transaction following a successful rebind. The user must explicitly commit the transaction. This enables ″what if″ analysis, in which the user updates certain statistics, and then tries to rebind the package to see what changes. It also permits multiple rebinds within a unit of work.

**Note:** The REBIND command *will* commit the transaction if auto-commit is enabled.

This command:

- Provides a quick way to recreate a package. This enables the user to take advantage of a change in the system without a need for the original bind file. For example, if it is likely that a particular SQL statement can take advantage of a newly created index, the REBIND command can be used to recreate the package. REBIND can also be used to recreate packages after RUNSTATS has been executed, thereby taking advantage of the new statistics.
- Provides a method to recreate inoperative packages. Inoperative packages must be explicitly rebound by invoking either the bind utility or the rebind utility. A package will be marked inoperative (the VALID column of the `SYSCAT.PACKAGES` system catalog will be set to `X`) if a function instance on which the package depends is dropped.
- Gives users control over the rebinding of invalid packages. Invalid packages will be automatically (or implicitly) rebound by the database manager when they are executed. This might result in a noticeable delay in the execution of the first SQL request for the invalid package. It may be desirable to explicitly rebind invalid packages, rather than allow the system to automatically rebind them, in order to eliminate the initial delay and to prevent unexpected SQL error messages which might be returned in case the implicit rebind fails. For example, following migration, all packages stored in the database will be invalidated by the DB2 Version 8 migration process. Given that this might involve a large number of

packages, it may be desirable to explicitly rebind all of the invalid packages at one time. This explicit rebinding can be accomplished using BIND, REBIND, or the **db2rbind** tool).

If multiple versions of a package (many versions with the same package name and creator) exist, only one version can be rebound at once. If not specified in the VERSION option, the package version defaults to be "". Even if there exists only one package with a name that matches, it will not be rebound unless its version matches the one specified or the default.

The choice of whether to use BIND or REBIND to explicitly rebind a package depends on the circumstances. It is recommended that REBIND be used whenever the situation does not specifically require the use of BIND, since the performance of REBIND is significantly better than that of BIND. BIND *must* be used, however:

- When there have been modifications to the program (for example, when SQL statements have been added or deleted, or when the package does not match the executable for the program).
- When the user wishes to modify any of the bind options as part of the rebind. REBIND does not support any bind options. For example, if the user wishes to have privileges on the package granted as part of the bind process, BIND must be used, since it has a **grant** option.
- When the package does not currently exist in the database.
- When detection of *all* bind errors is desired. REBIND only returns the first error it detects, whereas the BIND command returns the first 100 errors that occur during binding.

REBIND is supported by DB2 Connect.

If REBIND is executed on a package that is in use by another user, the rebind will not occur until the other user's logical unit of work ends, because an exclusive lock is held on the package's record in the SYSCAT.PACKAGES system catalog table during the rebind.

When REBIND is executed, the database manager recreates the package from the SQL statements stored in the SYSCAT.STATEMENTS system catalog table.

If REBIND encounters an error, processing stops, and an error message is returned.

REBIND will re-explain packages that were created with the **explsnap** bind option set to YES or ALL (indicated in the EXPLAIN_SNAPSHOT column in the SYSCAT.PACKAGES catalog table entry for the package) or with the **explain** bind option set to YES or ALL (indicated in the EXPLAIN_MODE column in the SYSCAT.PACKAGES catalog table entry for the package). The Explain tables used are those of the REBIND requester, not the original binder.

If an SQL statement was found to be in error and the BIND option SQLERROR CONTINUE was specified, the statement will be marked as invalid even if the problem has been corrected. REBIND will not change the state of an invalid statement. In a package bound with VALIDATE RUN, a statement can change from static to incremental bind or incremental bind to static across a REBIND depending on whether or not object existence or authority problems exist during the REBIND.

Rebinding a package with REOPT ONCE/ALWAYS might change static and dynamic statement compilation and performance.

If REOPT is not specified, REBIND will preserve the existing REOPT value used at precompile or bind time.

**Related reference:**

- "BIND" on page 286
- "RUNSTATS" on page 667
- "db2rbind - Rebind all Packages" on page 189

# RECONCILE

Validates the references to files for the DATALINK data of a table. The rows for which the references to files cannot be established are copied to the exception table (if specified), and modified in the input table.

Reconcile produces a message file (reconcil.msg) in the instance path on UNIX based systems, and in the install path on Windows platforms. This file will contain warning and error messages that are generated during validation of the exception table.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*
- CONTROL privilege on the table.

**Required connection:**

Database

**Command syntax:**

```
►►──RECONCILE──table-name──DLREPORT──filename──┬──────────────────────────────┬──────►◄
                                               └─FOR EXCEPTION──table-name─┘
```

**Command parameters:**

**RECONCILE table-name**
: Specifies the table on which reconciliation is to be performed. An alias, or the fully qualified or unqualified table name can be specified. A qualified table name is in the form *schema.tablename*. If an unqualified table name is specified, the table will be qualified with the current authorization ID.

**DLREPORT filename**
: Specifies the file that will contain information about the files that are unlinked during reconciliation. The name must be fully qualified (for example, `/u/johnh/report`). The reconcile utility appends a `.ulk` extension to the specified file name (for example, `report.ulk`). When no table is provided with the FOR EXCEPTION clause, a `.exp` file extension is appended to the exception report file.

**FOR EXCEPTION table-name**
: Specifies the exception table into which rows that encounter link failures for DATALINK values are to be copied. If no table is specified, an exception report file is generated in the directory specified in the ″DLREPORT″ option.

**Examples:**

The following command reconciles the table DEPT, and writes exceptions to the exception table EXCPTAB, which was created by the user. Information about files that were unlinked during reconciliation is written into the file `report.ulk`, which is created in the directory /u/johnh. If FOR EXCEPTION excptab had not been specified, the exception information would have been written to the file report.exp, created in the /u/johnh directory.

```
db2 reconcile dept dlreport /u/johnh/report for exception excptab
```

**Usage notes:**

During reconciliation, attempts are made to link files which exist according to table data, but which do not exist according to Data Links File Manager metadata, if no other conflict exists. A required DB2 Data Links Manager is one which has a referenced DATALINK value in the table. Reconcile tolerates the unavailability of a required DB2 Data Links Manager as well as other DB2 Data Links Managers that are configured to the database but are not part of the table data.

Reconciliation is performed with respect to all DATALINK data in the table. If file references cannot be reestablished, the violating rows are inserted into the exception table (if specified). These rows are not deleted from the input table. To ensure file reference integrity, the offending DATALINK values are nulled. If the column is defined as not nullable, the DATALINK values are replaced by a zero length URL.

If a file is linked under a DATALINK column defined with WRITE PERMISSION ADMIN and modified but not yet committed (that is, the file is still in the update-in-progress state), the reconciliation process renames the modified file to a filename with .mod as the suffix. It also removes the file from the update-in-progress state. If the DATALINK column is defined with RECOVERY YES, the previous archive version is restored.

If an exception table is not specified, the host name, file name, column ID, and reason code for each of the DATALINK column values for which file references could not be reestablished are copied to an exception report file (<*filename*>.exp). If the file reference could not be reestablished because the DB2 Data Links Manager is unavailable or was dropped from the database using the DROP DATALINKS MANAGER command, the file name reported in the exception report file is not the full file name. The prefix will be missing. For example, if the original DATALINK value was http://host.com/dlfs/x/y/a.b, the value reported in the exception table will be http://host.com/x/y/a.b. The prefix name 'dlfs' will not be included.

If the DATALINK column is defined with RECOVERY YES, the previous archive version is restored.

At the end of the reconciliation process, the table is taken out of datalink reconcile pending (DRP) state only if reconcile processing is complete on all the required DB2 Data Links Managers. If reconcile processing is pending on any of the required DB2 Data Links Managers (because they were unavailable), the table will remain, or be placed, in DRP state. If for some reason, an exception occurred on one of the affected Data Links Managers such that the reconciliation could not be completed successfully, the table might also be placed in a DRNP state, for which further manual intervention will be required before full referential integrity for that table can be restored.

The exception table, if specified, should be created before the reconcile utility is run. The exception table used with the reconcile utility is identical to the exception table used by the load utility.

The exception table mimics the definition of the table being reconciled. It can have one or two optional columns following the data columns. The first optional column is the TIMESTAMP column. It will contain the time stamp for when the reconcile operation was started. The second optional column should be of type CLOB (32KB or larger). It will contain the IDs of columns with link failures, and the reasons for those failures. The DATALINK columns in the exception table should specify NO LINK CONTROL. This ensures that a file is not linked when a row (with a DATALINK column) is inserted, and that an access token is not generated when rows are selected from the exception table.

Information in the MESSAGE column is organized according to the following structure:

| Field number | Content | Size | Comments |
|---|---|---|---|
| 1 | Number of violations | 5 characters | Right justified padded with '0' |
| 2 | Type of violation | 1 character | 'L' - DATALINK violation |
| 3 | Length of violation | 5 characters | Right justified padded with '0' |
| 4 | Number of violating DATALINK columns | 4 characters | Right justified padded with '0' |
| 5 | DATALINK column number of the first violating column | 4 characters | Right justified padded with '0' |
| 6 | Reason for violation | 5 characters | Right justified padded with '0' |
| | | | Repeat Fields 5 and 6 for each violating column |

The following is a list of possible violations:

```
00001-File could not be found by DB2 Data Links Manager.
00002-File already linked.
00003-File in modified state.
00004-Prefix name not registered.
00005-File could not be retrieved.
00006-File entry missing. This will happen for
      RECOVERY NO, READ PERMISSION FS, WRITE PERMISSION FS DATALINK
      columns. Use update to relink the file.
00007-File is in unlink state.
00008-File restored but modified file has been copied to
      <filename>.MOD
00009-File is already linked to another table.
00010-DB2 Data Links Manager referenced by the DATALINK
      value has been dropped from the database using the
      DROP DATALINKS MANAGER command.
00999-File could not be linked.

Example:

00001L000220002000400002000500001

   00001 - Specifies that the number of violations is 1.
   L     - Specifies that the type of violation is 'DATALINK violation'.
```

```
00022 - Specifies that the length of the violation is 12 bytes.
0002  - Specifies that there are 2 columns in the row which
        encountered link failures.
0004,00002
0005,00001 - Specifies the column ID and the reason for the violation.
```

If the message column is present, the time stamp column must also be present.

**Related concepts:**

- “Failure and recovery overview” in the *DB2 Data Links Manager Administration Guide and Reference*

# RECOVER DATABASE

Restores and rolls forward a database to a particular point in time or to the end of the logs.

**Scope:**

In a partitioned database environment, this command can only be invoked from the catalog partition. A database recover operation to a specified point in time affects all partitions that are listed in the `db2nodes.cfg` file. A database recover operation to the end of logs affects the partitions that are specified. If no partitions are specified, it affects all partitions that are listed in the `db2nodes.cfg` file.

**Authorization:**

To recover an existing database, one of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

To recover to a new database, one of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

To recover an existing database, a database connection is required. This command automatically establishes a connection to the specified database and will release the connection when the recover operation finishes. To recover to a new database, an instance attachment and a database connection are required. The instance attachment is required to create the database.

**Command syntax:**

```
▶▶──RECOVER──┬─DATABASE─┬──source-database-alias──────────────────────────▶
             └─DB───────┘

▶──┬─────────────────────────────────────────────────────────────────┬──▶
   │               ┌─USING LOCAL TIME─┐                               │
   └─TO──┬─isotime─┤                  ├──┬────────────────────────┬─┤
         │         └─USING GMT TIME───┘  └─ON ALL DBPARTITIONNUMS─┘ │
         └─END OF LOGS──┬──────────────────────────────────┬────────┘
                        └─┤ On Database Partition clause ├─┘

▶──┬─────────────────────────────────────┬──┬────────────────────────────────┬──▶
   └─USER──username──┬─────────────────┬─┘  └─OPEN──num-sessions──SESSIONS─┘
                     └─USING──password─┘

▶──┬────────────────────────────────────────────────────────────────────┬──▶
   └─USING HISTORY FILE──(──history-file──┬─────────────────────────┬──)─┘
                                          └─,─┤ History File clause ├─┘

▶──┬────────────────────────────────────────────────────────────────────┬──▶◀
   └─OVERFLOW LOG PATH──(──log-directory──┬─────────────────────────┬──)─┘
                                          └─,─┤ Log Overflow clause ├─┘
```

### On Database Partition clause:

```
├──ON──┬─┤ Database Partition List clause ├──────────────────────────────┬──────────┤
       └─ALL DBPARTITIONNUMS──┬─────────────────────────────────────────┬─┘
                              └─EXCEPT──┤ Database Partition List clause ├─┘
```

### Database Partition List clause:

```
                                    ┌─,──────────────────────────────────────────────────┐
                                    ▼                                                    │
├──┬─DBPARTITIONNUM──┬──(────db-partition-number1──┬────────────────────────────┬──┴──)──────────┤
   └─DBPARTITIONNUMS─┘                             └─TO──db-partition-number2─┘
```

### Log Overflow clause:

```
   ┌─,─────────────────────────────────────────────────────────┐
   ▼                                                           │
├────log-directory──ON DBPARTITIONNUM──db-partition-number1──┴──────────────┤
```

### History File clause:

```
   ┌─,─────────────────────────────────────────────────────────┐
   ▼                                                           │
├────history-file──ON DBPARTITIONNUM──db-partition-number1──┴──────────────┤
```

**Command parameters:**

**DATABASE** *database-alias*
The alias of the database that is to be recovered.

**USER** *username*
The user name under which the database is to be recovered.

**USING** *password*
The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

**TO**

*isotime* The point in time to which all committed transactions are to be recovered (including the transaction committed precisely at that time, as well as all transactions committed previously).

This value is specified as a time stamp, a 7-part character string that identifies a combined date and time. The format is *yyyy-mm-dd-hh.mm.ss.nnnnnn* (year, month, day, hour, minutes, seconds, microseconds), expressed in Coordinated Universal Time (UTC). UTC helps to avoid having the same time stamp associated with different logs (because of a change in time associated with daylight savings time, for example). The time stamp in a backup image is based on the local time at which the backup operation started. The CURRENT TIMEZONE special register specifies the difference between UTC and local time at the application server. The difference is represented by a time duration (a decimal number in which the first two digits represent the number of hours, the next two digits represent the number of minutes, and the last two digits represent the number of seconds). Subtracting CURRENT TIMEZONE from a local time converts that local time to UTC.

**USING LOCAL TIME**

Specifies the point in time to which to recover. This option allows the user to recover to a point in time that is the user's local time rather than GMT time. This makes it easier for users to recover to a specific point in time on their local machines, and eliminates potential user errors due to the translation of local to GMT time.

**Notes:**

1. If the user specifies a local time for recovery, all messages returned to the user will also be in local time. Note that all times are converted on the server, and in partitioned database environments, on the catalog database partition.
2. The timestamp string is converted to GMT on the server, so the time is local to the server's time zone, not the client's. If the client is in one time zone and the server in another, the server's local time should be used. This is different from the local time option from the Control Center, which is local to the client.
3. If the timestamp string is close to the time change of the clock due to daylight saving time, it is important to know if the stop time is before or after the clock change, and specify it correctly.

**USING GMT TIME**

Specifies the point in time to which to recover.

**END OF LOGS**

Specifies that all committed transactions from all online archive log files listed in the database configuration parameter *logpath* are to be applied.

**ON ALL DBPARTITIONNUMS**

Specifies that transactions are to be rolled forward on all partitions specified in the `db2nodes.cfg` file. This is the default if a database partition clause is not specified.

**EXCEPT**

Specifies that transactions are to be rolled forward on all partitions specified in the `db2nodes.cfg` file, except those specified in the database partition list.

**ON DBPARTITIONNUM / ON DBPARTITIONNUMS**

Roll the database forward on a set of database partitions.

*db-partition-number1*

Specifies a database partition number in the database partition list.

*db-partition-number2*

Specifies the second database partition number, so that all partitions from *db-partition-number1* up to and including *db-partition-number2* are included in the database partition list.

**OPEN** *num-sessions* **SESSIONS**

Specifies the number of I/O sessions that are to be used with Tivoli Storage Manager (TSM) or the vendor product.

**USING HISTORY FILE** *history-file*

*history-file* **ON DBPARTITIONNUM**

In a partitioned database environment, allows a different history file

**OVERFLOW LOG PATH** *log-directory*

Specifies an alternate log path to be searched for archived logs during recovery. Use this parameter if log files were moved to a location other than that specified by the *logpath* database configuration parameter. In a partitioned database environment, this is the (fully qualified) default overflow log path *for all partitions*. A relative overflow log path can be specified for single-partition databases.

**Note:** The OVERFLOW LOG PATH command parameter will overwrite the value (if any) of the database configuration parameter *overflowlogpath*.

*log-directory* **ON DBPARTITIONNUM**

In a partitioned database environment, allows a different log path to override the default overflow log path for a specific partition.

**Examples:**

In a single-partition database environment, where the database being recovered currently exists, and so the most recent version of the history file is available in the *dftdbpath*:

1. To use the latest backup image and rollforward to the end of logs using all default values:
```
RECOVER DB SAMPLE
```
2. To recover the database to a PIT, issue the following. The most recent image that can be used will be restored, and logs applied until the PIT is reached.
```
RECOVER DB SAMPLE TO  2001-12-31-04:00:00
```
3. To recover the database using a saved version of the history file. issue the following. For example, if the user needs to recover to an extremely old PIT which is no longer contained in the current history file, the user will have to provide a version of the history file from this time period. If the user has saved a history file from this time period, this version can be used to drive the recover.
```
RECOVER DB SAMPLE TO  1999-12-31-04:00:00
   USING HISTORY FILE (/home/user/old1999files/db2rhist.asc)
```

In a single-partition database environment, where the database being recovered does not exist, you must use the USING HISTORY FILE clause to point to a history file.

1. If you have not made any backups of the history file, so that the only version available is the copy in the backup image, the recommendation is to issue a RESTORE followed by a ROLLFORWARD. However, to use RECOVER, you would first have to extract the history file from the image to some location, for example `/home/user/oldfiles/db2rhist.asc`, and then issue this command. (This version of the history file does not contain any information about log files that are required for rollforward, so this history file is not useful for RECOVER.)
```
RECOVER DB SAMPLE TO END OF LOGS
   USING HISTORY FILE (/home/user/fromimage/db2rhist.asc)
```
2. If you have been making periodic or frequent backup copies of the history, the USING HISTORY clause should be used to point to this version of the history file. If the file is /home/user/myfiles/db2rhist.asc, issue the command:
```
RECOVER DB SAMPLE TO PIT
   USING HISTORY FILE (/home/user/myfiles/db2rhist.asc)
```

   (In this case, you can use any copy of the history file, not necessarily the latest, as long as it contains a backup taken before the point-in-time (PIT) requested.)

In a partitioned database envrionment, where the database exists on all database partitions, and the latest history file is available on *dftdbpath* on all database partitions:

1. To recover the database to a PIT on all nodes. DB2 will verify that the PIT is reachable on all nodes before starting any restore operations.

   ```
   RECOVER DB SAMPLE TO  2001:12:31:04:00:00
   ```

2. To recover the database to this PIT on all nodes. DB2 will verify that the PIT is reachable on all nodes before starting any restore operations. The RECOVER operation on each node is identical to a single-partition RECOVER.

   ```
   RECOVER DB SAMPLE TO END OF LOGS
   ```

3. Even though the most recent version of the history file is in the *dftdbpath*, you might want to use several specific history files. Unless otherwise specified, each partition will use the history file found locally at /home/user/oldfiles/db2rhist.asc. The exceptions are nodes 2 and 4. Node 2 will use: /home/user/node2files/db2rhist.asc, and node 4 will use: /home/user/node4files/db2rhist.asc.

   ```
   RECOVER DB SAMPLE TO  1999:12:31:04:00:00
      USING HISTORY FILE (/home/user/oldfiles/db2rhist.asc,
         /home/user/node2files/db2rhist.asc ON DBPARTITIONNUM 2,
         /home/user/node4files/db2rhist.asc ON DBPARTITIONNUM 4)
   ```

4. It is possible to recover a subset of nodes instead of all nodes, however a PIT RECOVER can not be done in this case, the recover must be done to EOL.

   ```
   RECOVER DB SAMPLE TO END OF LOGS ON DBPARTITIONNUMS(2 TO 4, 7, 9)
   ```

In a partitioned database environment, where the database does not exist:

1. If you have not made any backups of the history file, so that the only version available is the copy in the backup image, the recommendation is to issue a RESTORE followed by a ROLLFORWARD. However, to use RECOVER, you would first have to extract the history file from the image to some location, for example, /home/user/oldfiles/db2rhist.asc, and then issue this command. (This version of the history file does not contain any information about log files that are required for rollforward, so this history file is not useful for the recover.)

   ```
   RECOVER DB SAMPLE TO PIT
      USING HISTORY FILE (/home/user/fromimage/db2rhist.asc)
   ```

2. If you have been making periodic or frequent backup copies of the history, the USING HISTORY clause should be used to point to this version of the history file. If the file is /home/user/myfiles/db2rhist.asc, you can issue the following command:

   ```
   RECOVER DB SAMPLE TO END OF LOGS
      USING HISTORY FILE (/home/user/myfiles/db2rhist.asc)
   ```

**Usage notes:**

- Recovering a database might require a load recovery using tape devices. If prompted for another tape, the user can respond with one of the following:

  **c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted).

  **d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes).

  **t** Terminate. Terminate all devices.

- If there is a failure during the restore portion of the recover operation, you can reissue the RECOVER DATABASE command. If the restore operation was successful, but there was an error during the rollforward operation, you can issue a ROLLFORWARD DATABASE command, since it is not necessary (and it is time-consuming) to redo the entire recover operation.

- In a partitioned database environment, if there is an error during the restore portion of the recover operation, it is possible that it is only an error on a single database partition. Instead of reissuing the RECOVER DATABASE command, which restores the database on all database partitions, it is more efficient to issue a RESTORE DATABASE command for the database partition that failed, followed by a ROLLFORWARD DATABASE command.

# REDISTRIBUTE DATABASE PARTITION GROUP

Redistributes data across the database partitions in a database partition group. The current data distribution, whether it is uniform or skewed, can be specified. The redistribution algorithm selects the partitions to be moved based on the current data distribution.

This command can only be issued from the catalog database partition. Use the LIST DATABASE DIRECTORY command to determine which database partition is the catalog database partition for each database.

**Scope:**

This command affects all database partitions in the database partition group.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *dbadm*

**Command syntax:**

```
►►──REDISTRIBUTE DATABASE PARTITION GROUP──database partition group──────────────►

►──┬─┬─UNIFORM─────────────────┬─┬───────────────────────────────────────────────►◄
   │ └─USING DISTFILE──distfile─┘ │
   ├─USING TARGETMAP──targetmap───┤
   ├─CONTINUE─────────────────────┤
   └─ROLLBACK─────────────────────┘
```

**Command parameters:**

**DATABASE PARTITION GROUP database partition group**

The name of the database partition group. This one-part name identifies a database partition group described in the SYSCAT.DBPARTITIONGROUPS catalog table. The database partition group cannot currently be undergoing redistribution.

**Note:** Tables in the IBMCATGROUP and the IBMTEMPGROUP database partition groups cannot be redistributed.

**UNIFORM**

Specifies that the data is uniformly distributed across hash partitions (that is, every hash partition is assumed to have the same number of rows), but the same number of hash partitions do not map to each database partition. After redistribution, all database partitions in the database partition group have approximately the same number of hash partitions.

**USING DISTFILE distfile**

If the distribution of partitioning key values is skewed, use this option to achieve a uniform redistribution of data across the database partitions of a database partition group.

Use the *distfile* to indicate the current distribution of data across the 4 096 hash partitions.

Use row counts, byte volumes, or any other measure to indicate the amount of data represented by each hash partition. The utility reads the integer value associated with a partition as the weight of that partition. When a *distfile* is specified, the utility generates a target partitioning map that it uses to redistribute the data across the database partitions in the database partition group as uniformly as possible. After the redistribution, the weight of each database partition in the database partition group is approximately the same (the weight of a database partition is the sum of the weights of all partitions that map to that database partition).

For example, the input distribution file might contain entries as follows:

```
10223
1345
112000
0
100
...
```

In the example, hash partition 2 has a weight of 112 000, and partition 3 (with a weight of 0) has no data mapping to it at all.

The *distfile* should contain 4 096 positive integer values in character format. The sum of the values should be less than or equal to 4 294 967 295.

If the path for *distfile* is not specified, the current directory is used.

**USING TARGETMAP targetmap**

The file specified in *targetmap* is used as the target partitioning map. Data redistribution is done according to this file. If the path is not specified, the current directory is used.

If a database partition included in the target map is not in the database partition group, an error is returned. Issue ALTER DATABASE PARTITION GROUP ADD DBPARTITIONNUM before running REDISTRIBUTE DATABASE PARTITION GROUP.

If a database partition excluded from the target map *is* in the database partition group, that database partition will not be included in the partitioning. Such a database partition can be dropped using ALTER DATABASE PARTITION GROUP DROP DBPARTITIONNUM either before or after REDISTRIBUTE DATABASE PARTITION GROUP.

**CONTINUE**

Continues a previously failed REDISTRIBUTE DATABASE PARTITION GROUP operation. If none occurred, an error is returned.

**ROLLBACK**

Rolls back a previously failed REDISTRIBUTE DATABASE PARTITION GROUP operation. If none occurred, an error is returned.

**Usage notes:**

When a redistribution operation is done, a message file is written to:

- The `/sqllib/redist` directory on UNIX based systems, using the following format for subdirectories and file name: *database-name.database-partition-group-name.timestamp*.
- The `\sqllib\redist\` directory on Windows operating systems, using the following format for subdirectories and file name: *database-name\first-eight-characters-of-the-database-partition-group-name\date\time*.

The time stamp value is the time when the command was issued.

This utility performs intermittent COMMITs during processing.

Use the ALTER DATABASE PARTITION GROUP statement to add database partitions to a database partition group. This statement permits one to define the containers for the table spaces associated with the database partition group.

**Note:** DB2 Parallel Edition for AIX Version 1 syntax, with ADD DBPARTITIONNUM and DROP DBPARITITIONNUM options, is supported for users with *sysadm* or *sysctrl* authority. For ADD DBPARTITIONNUM, containers are created like the containers on the lowest node number of the existing nodes within the database partition group.

All packages having a dependency on a table that has undergone redistribution are invalidated. It is recommended to explicitly rebind such packages after the redistribute database partition group operation has completed. Explicit rebinding eliminates the initial delay in the execution of the first SQL request for the invalid package. The redistribute message file contains a list of all the tables that have undergone redistribution.

It is also recommended to update statistics by issuing RUNSTATS after the redistribute database partition group operation has completed.

Database partition groups containing replicated summary tables or tables defined with DATA CAPTURE CHANGES cannot be redistributed.

Redistribution is not allowed if there are user temporary table spaces with existing declared temporary tables in the database partition group.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODEGROUP can be substituted for DATABASE PARTITION GROUP.

**Related reference:**

- "LIST DATABASE DIRECTORY" on page 483
- "RUNSTATS" on page 667
- "REBIND" on page 595

# REFRESH LDAP

Refreshes the cache on a local machine with updated information when the information in Lightweight Directory Access Protocol (LDAP) has been changed.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──REFRESH LDAP──┬─CLI CFG──┬──────────────────────────────────────────────►◄
                  ├─DB DIR───┤
                  └─NODE DIR─┘
```

**Command parameters:**

**CLI CFG**
Specifies that the CLI configuration is to be refreshed.

**Note:** This parameter is not supported on AIX or the Solaris Operating Environment.

**DB DIR**
Specifies that the database directory is to be refreshed.

**NODE DIR**
Specifies that the node directory is to be refreshed.

**Usage notes:**

If the object in LDAP is removed during refresh, the corresponding LDAP entry on the local machine is also removed. If the information in LDAP is changed, the corresponding LDAP entry is modified accordingly. If the `DB2CLI.INI` file is manually updated, the REFRESH LDAP CLI CFG command must be run to update the cache for the current user.

The REFRESH LDAP DB DIR and REFRESH LDAP NODE DIR commands remove the LDAP database or node entries found in the local database or node directories. The database or node entries will be added to the local database or node directories again when the user connects to a database or attaches to an instance found in LDAP, and DB2LDAPCACHE is either not set or set to YES.

# REGISTER

Registers the DB2 server in the network directory server.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──REGISTER──┬────────────┬──┬────┬──┬───────┬──┤ LDAP path ├──────────────►◄
              └─DB2 SERVER─┘  └─IN─┘  └─ADMIN─┘
```

## LDAP path:

```
├──LDAP──┬─NODE─┬──nodename──────────────────────────────────────────────────►
         └─AS───┘

►─PROTOCOL──┬─TCPIP──┬─────────────────────┬──┬─────────────────────┬──┬─────────────────┬─┬──►
            │        └─HOSTNAME──hostname─┘  └─SVCNAME──svcename─┘  └─SECURITY──SOCKS─┘ │
            ├─NETBIOS──┬──────────────┬──────────────────────────────────────────────────┤
            │          └─NNAME──nname─┘                                                   │
            ├─NPIPE─────────────────────────────────────────────────────────────────────┤
            └─┤ APPN path ├─────────────────────────────────────────────────────────────┘

►──┬────────────────────┬──┬──────────────────────┬──┬───────────────────────────┬──┬───────────────┬──►
   └─REMOTE──computer─┘  └─INSTANCE──instance─┘  │             ┌─SERVER─┐    │  └─OSTYPE──ostype─┘
                                                  └─NODETYPE──┼─MPP────┼────┘
                                                              └─DCS────┘

►──┬────────────────────┬──┬──────────────────────────────────────────┬──────────────────────┤
   └─WITH──"comments"──┘  └─USER──username──┬─────────────────────┬─┘
                                           └─PASSWORD──password─┘
```

## APPN path:

```
├──APPN──NETWORK──net_id──PARTNERLU──partner_lu──MODE──mode──────────────────►

►──┬──────────────────┬──┬───────────────────────┬──┬─────────────────────────┬──►
   └─TPNAME──tp_name─┘  └─SECURITY──┬─NONE────┬─┘  └─LANADDRESS──lan_address─┘
                                    ├─SAME────┤
                                    └─PROGRAM─┘

►──┬───────────────────────────────────┬────────────────────────────────────────┤
   └─CHGPWDLU──change_password_lu─┘
```

**Command parameters:**

**IN** Specifies the network directory server on which to register the DB2 server. The valid value is: LDAP for an LDAP (Lightweight Directory Access Protocol) directory server.

**ADMIN**
Specifies that an administration server node is to be registered.

**NODE/AS nodename**
Specify a short name to represent the DB2 server in LDAP. A node entry

will be cataloged in LDAP using this node name. The client can attach to the server using this node name. The protocol associated with this LDAP node entry is specified through the PROTOCOL parameter.

**NODE nodename**
Specify a short name to represent the DB2 server in LDAP. A node entry will be cataloged in LDAP using this node name. The client can attach to the server using this node name. The protocol associated with this LDAP node entry is specified through the PROTOCOL parameter.

**PROTOCOL**
Specifies the protocol type associated with the LDAP node entry. Since the database server can support more than one protocol type, this value specifies the protocol type used by the client applications. The DB2 server must be registered once per protocol. Valid values are: `TCPIP`, `NETBIOS`, `APPN`, and `NPIPE`. Specify the latter to use Windows Named Pipe. This protocol type is only supported by DB2 servers that run on Windows operating systems.

**Note:** NETBIOS and NPIPE are not supported on AIX and Solaris operating systems, however these protocols can be registered for a remote server using an operating system such as Windows NT.

**HOSTNAME hostname**
Specifies the TCP/IP host name (or IP address).

**SVCENAME svcename**
Specifies the TCP/IP service name or port number.

**SECURITY SOCKS**
Specifies that TCP/IP socket security is to be used.

**NNAME nname**
Specifies the NetBIOS workstation name.

**NETWORK net_id**
Specifies the APPN network ID.

**PARTNERLU partner_lu**
Specifies the APPN partner LU name for the DB2 server machine.

**MODE mode**
Specifies the APPN mode name.

**TPNAME tpname**
Specifies the APPN transaction program name. The default is DB2DRDA.

**SECURITY**
Specifies the APPN security level. Valid values are:

**NONE**
Specifies that no security information is to be included in the allocation request sent to the server. This is the default security type for DB2 UDB server.

**SAME** Specifies that a user name is to be included in the allocation request sent to the server, together with an indicator that the user name has been ″already verified″. The server must be configured to accept ″already verified″ security.

**PROGRAM**
Specifies that both a user name and a password are to be included

in the allocation request sent to the server. This is the default security type for host database servers such as DB2 for OS/390 or z/OS, DB2 for iSeries.

**LANADDRESS lan_address**
Specifies the APPN network adaptor address.

**CHGPWDLU change_password_lu**
Specifies the name of the partner LU that is to be used when changing the password for a host database server.

**REMOTE computer**
Specifies the computer name of the machine on which the DB2 server resides. Specify this parameter only if registering a remote DB2 server in LDAP. The value must be the same as the value specified when adding the server machine to LDAP. For Windows operating systems, this is the computer name. For UNIX based systems, this is the TCP/IP host name.

**INSTANCE instance**
Specifies the instance name of the DB2 server. The instance name must be specified for a remote instance (that is, when a value for the REMOTE parameter has been specified).

**NODETYPE**
Specifies the node type for the database server. Valid values are:

**SERVER**
Specify the SERVER node type for a DB2 UDB Enterprise Edition server. This is the default.

**MPP** Specify the MPP node type for a DB2 UDB Enterprise Edition - Extended (partitioned database) server.

**DCS** Specify the DCS node type when registering a host database server; this directs the client or gateway to use DRDA as the database protocol.

**OSTYPE ostype**
Specifies the operating system type of the server machine. Valid values are: AIX, NT, HPUX, SUN, MVS, OS400, VM, VSE, SNI, SCO and LINUX. If an operating system type is not specified, the local operating system type will be used for a local server and no operating system type will be used for a remote server.

**WITH "comments"**
Describes the DB2 server. Any comment that helps to describe the server registered in the network directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

**Usage notes:**

Register the DB2 server once for each protocol that the server supports. For example, if the DB2 server supports both NetBIOS and TCP/IP, the REGISTER command must be invoked twice:

```
db2 register db2 server in ldap as tcpnode protocol tcpip
db2 register db2 server in ldap as nbnode protocol netbios
```

The REGISTER command should be issued once for each DB2 server instance to publish the server in the directory server. If the communication parameter fields are reconfigured, or the server network address changes, update the DB2 server on the network directory server.

To update the DB2 server in LDAP, use the UPDATE LDAP NODE command after the changes have been made.

If any protocol configuration parameter is specified when registering a DB2 server locally, it will override the value specified in the database manager configuration file.

For APPN, only the TPNAME is found in the database manager configuration file. To register APPN properly, values for the following mandatory parameters must be specified: NETWORK, PARTNERLU, MODE, TPNAME, and SECURITY. Values for the following optional parameters can also be specified: LANADDRESS and CHGPWDLU.

If the REGISTER command is used to register a local DB2 instance in LDAP, and one or both of NODETYPE and OSTYPE are specified, they will be replaced with the values retrieved from the local system. If the REGISTER command is used to register a remote DB2 instance in LDAP, and one or both of NODETYPE and OSTYPE are not specified, the default value of SERVER and `Unknown` will be used, respectively.

If the REGISTER command is used to register a remote DB2 server in LDAP, the computer name and the instance name of the remote server must be specified along with the communication protocol for the remote server.

When registering a host database server, a value of DCS must be specified for the NODETYPE parameter.

**Related reference:**

- "DEREGISTER" on page 344
- "UPDATE LDAP NODE" on page 738

# REORG INDEXES/TABLE

Reorganizes an index or a table.

The index option reorganizes all indexes defined on a table by rebuilding the index data into unfragmented, physically contiguous pages. If you specify the CLEANUP ONLY option of the index option, cleanup is performed without rebuilding the indexes. This command cannot be used against indexes on declared temporary tables (SQLSTATE 42995).

The table option reorganizes a table by reconstructing the rows to eliminate fragmented data, and by compacting information.

**Scope:**

This command affects all database partitions in the database partition group.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*
- CONTROL privilege on the table.

**Required connection:**

Database

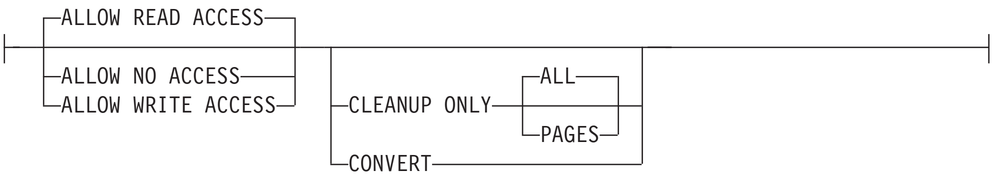
**Command syntax:**

```
►►──REORG──┬─TABLE──table-name──┤ Table Clause ├──────────────────────┬──►
           └─INDEXES ALL FOR TABLE──table-name──┤ Index Clause ├──────┘

►──┬───────────────────────────────────┬──────────────────────────────►◄
   └─┤ Database Partition Clause ├─────┘
```

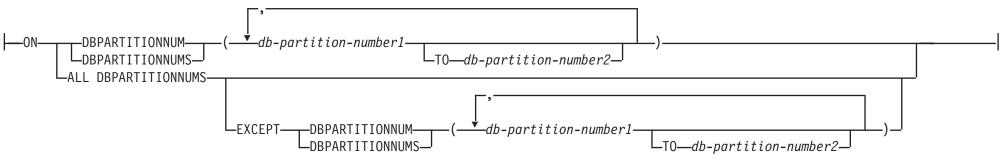
### Table Clause:

```
├──┬──────────────────────┬──────────────────────────────────────────────────►
   └─INDEX──index-name────┘

     ┌─ALLOW READ ACCESS─┐
►──┬─┼───────────────────┼──┬──────────────┬──┬───────────┬──┬─────────────┬─┬────┤
   │ └─ALLOW NO ACCESS───┘  └─USE──tbspace─┘  └─INDEXSCAN─┘  └─LONGLOBDATA─┘ │
   │           ┌─ALLOW WRITE ACCESS─┐                        ┌─START──┐       │
   └─INPLACE───┬─┼────────────────────┼──┬──────────────────┬──┼────────┼─┬─────┘
               │ └─ALLOW READ ACCESS──┘  └─NOTRUNCATE TABLE─┘  └─RESUME─┘ │
               ├─STOP─────────────────────────────────────────────────────┘
               └─PAUSE─┘
```

### Index Clause:



### Database Partition Clause:



**Command parameters:**

**INDEXES ALL FOR TABLE table-name**
Specifies the table whose indexes are to be reorganized. The table can be in a local or a remote database.

**ALLOW NO ACCESS**
Specifies that no other users can access the table while the indexes are being reorganized.

**ALLOW READ ACCESS**
Specifies that other users can have read-only access to the table while the indexes are being reorganized. This is the default.

**ALLOW WRITE ACCESS**
Specifies that other users can read from and write to the table while the indexes are being reorganized.

**CLEANUP ONLY**
When CLEANUP ONLY is requested, a cleanup rather than a full reorganization will be done. The indexes will not be rebuilt and any pages freed up will be available for reuse by indexes defined on this table only.

The CLEANUP ONLY PAGES option will search for and free committed pseudo empty pages. A committed pseudo empty page is one where all the keys on the page are marked as deleted and all these deletions are known to be committed. The number of pseudo empty pages in an indexes can be determined by running runstats and looking at the NUM EMPTY LEAFS column in SYSCAT.INDEXES. The PAGES option will clean the NUM EMPTY LEAFS if they are determined to be committed.

The CLEANUP ONLY ALL option will free committed pseudo empty pages, as well as remove committed pseudo deleted keys from pages that are not pseudo empty. This option will also try to merge adjacent leaf pages if doing so will result in a merged leaf page that has at least PCTFREE free space on the merged leaf page, where PCTFREE is the percent free space defined for the index at index creation time. The default PCTFREE is ten percent. If two pages can be merged, one of the pages will be freed. The number of pseudo deleted keys in an index , excluding those on pseudo empty pages, can be determined by running runstats and then selecting the NUMRIDS DELETED from SYSCAT.INDEXES. The ALL option will clean the NUMRIDS DELETED and the NUM EMPTY LEAFS if they are determined to be committed.

**ALL** Specifies that indexes should be cleaned up by removing committed pseudo deleted keys and committed pseudo empty pages.

**PAGES**

Specifies that committed pseudo empty pages should be removed from the index tree. This will not clean up pseudo deleted keys on pages that are not pseudo empty. Since it is only checking the pseudo empty leaf pages, it is considerably faster than using the ALL option in most cases.

**CONVERT**

If you are not sure whether the table you are operating on has a type-1 or type-2 index, but want type-2 indexes, you can use the CONVERT option. If the index is type 1, this option will convert it into type 2. If the index is already type 2, this option has no effect.

All indexes created by DB2 prior to Version 8 are type-1 indexes. All indexes created by Version 8 are Type 2 indexes, except when you create an index on a table that already has a type 1 index. In this case the new index will also be of type 1.

Using the INSPECT command to determine the index type can be slow. CONVERT allows you to ensure that the new index will be Type 2 without your needing to determine its original type.

Use the ALLOW READ ACCESS or ALLOW WRITE ACCESS option to allow other transactions either read-only or read-write access to the table while the indexes are being reorganized. Note that, while ALLOW READ ACCESS and ALLOW WRITE ACCESS allow access to the table, during the period in which the reorganized copies of the indexes are made available, no access to the table is allowed.

**TABLE table-name**

Specifies the table to reorganize. The table can be in a local or a remote database. The name or alias in the form: *schema.table-name* can be used. The *schema* is the user name under which the table was created. If you omit the schema name, the default schema is assumed.

**Note:** For typed tables, the specified table name must be the name of the hierarchy's root table.

You cannot specify an index for the reorganization of a multidimensional clustering (MDC) table. Also note that inplace reorganization of tables cannot be used for MDC tables.

**INDEX index-name**

Specifies the index to use when reorganizing the table. If you do not specify the fully qualified name in the form: *schema.index-name*, the default schema is assumed. The *schema* is the user name under which the index was created. The database manager uses the index to physically reorder the records in the table it is reorganizing.

For an inplace table reorganization, if a clustering index is defined on the table and an index is specified, it must be clustering index.

If the inplace option is not specified, any index specified will be used. If you do not specify the name of an index, the records are reorganized without regard to order. If the table has a clustering index defined, however, and no index is specified, then the clustering index is used to cluster the table. You cannot specify an index if you are reorganizing an MDC table.

**INPLACE**
Reorganizes the table while permitting user access.

Inplace table reorganization is allowed only on tables with type-2 indexes and without extended indexes. Inplace table reorganization takes place asynchronously, and might not be effective immediately.

**ALLOW READ ACCESS**
Allow only read access to the table during reorganization.

**ALLOW WRITE ACCESS**
Allow write access to the table during reorganization. This is the default behavior.

**NOTRUNCATE TABLE**
Do not truncate the table after inplace reorganization. During truncation, the table is S-locked.

**START**
Start the inplace REORG processing. Because this is the default, this keyword is optional.

**STOP** Stop the inplace REORG processing at its current point.

**PAUSE**
Suspend or pause inplace REORG for the time being.

**RESUME**
Continue or resume a previously paused inplace table reorganization.

**USE tablespace-name**
Specifies the name of a system temporary table space in which to store a temporary copy of the table being reorganized. If you do not provide a table space name, the database manager stores a working copy of the table in the table spaces that contain the table being reorganized.

For an 8KB, 16KB, or 32KB table object, the page size of any system temporary table space that you specify must match the page size of the table spaces in which the table data resides, including any LONG or LOB column data.

**INDEXSCAN**
For a clustering REORG an index scan will be used to re-order table records. Reorganize table rows by accessing the table through an index. The default method is to scan the table and sort the result to reorganize the table, using temporary table spaces as necessary. Even though the index keys are in sort order, scanning and sorting is typically faster than fetching rows by first reading the row identifier from an index.

**LONGLOBDATA**
Long field and LOB data are to be reorganized.

This is not required even if the table contains long or LOB columns. The default is to avoid reorganizing these objects because it is time consuming and does not improve clustering.

**Examples:**

For a classing REORG TABLE like the default in DB2, Version 7, enter the following command:

```
db2 reorg table employee index empid allow no access indexscan
 longlobdata
```

Note that the defaults are different in DB2, Version 8.

To reorganize a table to reclaim space and use the temporary table space mytemp1, enter the following command:

```
db2 reorg table homer.employee use mytemp1
```

To reorganize tables in a partitiongroup consisting of nodes 1, 2, 3, and 4 of a four-node system, you can enter either of the following commands:

```
db2 reorg table employee index empid on dbpartitionnum (1,3,4)

db2 reorg table homer.employee index homer.empid on all
  dbpartitionnums except dbpartitionnum (2)
```

To clean up the pseudo deleted keys and pseudo empty pages in all the indexes on the EMPLOYEE table while allowing other transactions to read and update the table, enter:

```
db2 reorg indexes all for table homer.employee allow write
 access cleanup only
```

To clean up the pseudo empty pages in all the indexes on the EMPLOYEE table while allowing other transactions to read and update the table, enter:

```
db2 reorg indexes all for table homer.employee allow write
 access cleanup only pages
```

To reorganize the EMPLOYEE table using the system temporary table space TEMPSPACE1 as a work area, enter:

```
db2 reorg table homer.employee use tempspace1
```

To start, pause, and resume an inplace reorg of the EMPLOYEE table with the default schema HOMER, which is specified explicitly in previous examples, enter the following commands:

```
db2 reorg table employee index empid inplace start
db2 reorg table employee inplace pause
db2 reorg table homer.employee inplace allow read access
 notruncate table resume
```

Note that the command to resume the reorg contains additional keywords to specify read access only and to skip the truncation step, which share-locks the table.

**Usage notes:**

Restrictions:

- The REORG utility does not support the use of nicknames.
- The REORG TABLE command is not supported for declared temporary tables.

- The REORG TABLE command cannot be used on views.
- Reorganization of a table is not compatible with range-clustered tables, because the range area of the table always remains clustered.
- REORG TABLE cannot be used on a DMS table while an online backup of a table space in which the table resides is being performed.
- REORG TABLE cannot use an index that is based on an index extension.

Information about the current progress of table reorganization is written to the history file for database activity. The history file contains a record for each reorganization event. To view this file, execute the LIST HISTORY command for the database that contains the table you are reorganizing.

You can also use table snapshots to monitor the progress of table reorganization. Table reorganization monitoring data is recorded regardless of the Database Monitor Table Switch setting.

If an error occurs, an SQLCA dump is written to the history file. For an in-place table reorganization, the status is recorded as PAUSED.

When an indexed table has been modified many times, the data in the indexes might become fragmented. If the table is clustered with respect to an index, the table and index can get out of cluster order. Both of these factors can adversely affect the performance of scans using the index, and can impact the effectiveness of index page prefetching. REORG INDEXES can be used to reorganize all of the indexes on a table, to remove any fragmentation and restore physical clustering to the leaf pages. Use REORGCHK to help determine if an index needs reorganizing. Be sure to complete all database operations and release all locks before invoking REORG INDEXES. This can be done by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK.

Indexes might not be optimal following an in-place REORG TABLE operation, since only the data object and not the indexes are reorganized. It is recommended that you perform a REORG INDEXES after an inplace REORG TABLE operation. Indexes are completely rebuilt during the last phase of a classic REORG TABLE, however, so reorganizing indexes is not necessary.

Tables that have been modified so many times that data is fragmented and access performance is noticeably slow are candidates for the REORG TABLE command. You should also invoke this utility after altering the inline length of a structured type column in order to benefit from the altered inline length. Use REORGCHK to determine whether a table needs reorganizing. Be sure to complete all database operations and release all locks before invoking REORG TABLE. This can be done by issuing a COMMIT after closing all cursors opened WITH HOLD, or by issuing a ROLLBACK. After reorganizing a table, use RUNSTATS to update the table statistics, and REBIND to rebind the packages that use this table. The reorganize utility will implicitly close all the cursors.

If the table contains mixed row format because the table value compression has been activated or deactivated, an offline table reorganization can convert all the existing rows into the target row format.

If the table is partitioned onto several database partitions, and the table reorganization fails on any of the affected database partitions, only the failing database partitions will have the table reorganization rolled back.

**Note:** If the reorganization is not successful, temporary files should not be deleted. The database manager uses these files to recover the database.

If the name of an index is specified, the database manager reorganizes the data according to the order in the index. To maximize performance, specify an index that is often used in SQL queries. If the name of an index is *not* specified, and if a clustering index exists, the data will be ordered according to the clustering index.

The PCTFREE value of a table determines the amount of free space designated per page. If the value has not been set, the utility will fill up as much space as possible on each page.

To complete a table space roll-forward recovery following a table reorganization, both regular and large table spaces must be enabled for roll-forward recovery.

If the table contains LOB columns that do not use the COMPACT option, the LOB DATA storage object can be significantly larger following table reorganization. This can be a result of the order in which the rows were reorganized, and the types of table spaces used (SMS or DMS).

**Related reference:**

- "GET SNAPSHOT" on page 419
- "REORGCHK" on page 624
- "RUNSTATS" on page 667
- "REBIND" on page 595
- "SNAPSHOT_TBREORG table function" in the *SQL Administrative Routines*

# REORGCHK

Calculates statistics on the database to determine if tables or indexes, or both, need to be reorganized or cleaned up.

**Scope:**

This command can be issued from any database partition in the `db2nodes.cfg` file. It can be used to update table and index statistics in the catalogs.

**Authorization:**

One of the following:
- *sysadm* or *dbadm* authority
- CONTROL privilege on the table.

**Required connection:**

Database

**Command syntax:**

```
           ┌─UPDATE STATISTICS──┐  ┌─ON TABLE USER───────────────────┐
►►─REORGCHK─┼────────────────────┼──┼─────────────────────────────────┼──────────►◄
           └─CURRENT STATISTICS─┘  └─ON───┬─SCHEMA──schema-name─────┬─┘
                                          │          ┌─USER───────┐ │
                                          └─TABLE────┼─SYSTEM─────┼─┘
                                                     ├─ALL────────┤
                                                     └─table-name─┘
```

**Command parameters:**

**UPDATE STATISTICS**
Calls the RUNSTATS routine to update table statistics, and then uses the updated statistics to determine if table reorganization is required.

If a table partition exists on the node where REORGCHK has been issued, RUNSTATS executes on this node. If a table partition does not exist on this node, the request is sent to the first node in the database partition group that holds a partition for the table. RUNSTATS then executes on that node.

**CURRENT STATISTICS**
Uses the current table statistics to determine if table reorganization is required.

**ON SCHEMA schema-name**
Checks all the tables created under the specified schema.

**ON TABLE**

**USER** Checks the tables that are owned by the run time authorization ID.

**SYSTEM**
Checks the system tables.

**ALL** Checks all user and system tables.

**table-name**
Specifies the table to check. The fully qualified name or alias in the form: *schema.table-name* must be used. The *schema* is the user name

under which the table was created. If the table specified is a system catalog table, the *schema* is SYSIBM.

**Note:** For typed tables, the specified table name must be the name of the hierarchy's root table.

**Examples:**

The following shows sample output from the command. Only a portion of the *Index Statistics* output is shown.

```
db2 reorgchk update statistics on table system
```

run against the SAMPLE database:

```
Doing RUNSTATS ....

Table statistics:

F1: 100 * OVERFLOW / CARD < 5
F2: 100 * (Effective Space Utilization of Data Pages) > 70
F3: 100 * (Required Pages / Total Pages) > 80
```

```
SCHEMA    NAME                  CARD    OV    NP    FP ACTBLK    TSIZE  F1  F2  F3 REORG
----------------------------------------------------------------------------------------
SYSIBM    SYSATTRIBUTES            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSBUFFERPOOLNODES       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSBUFFERPOOLS           1     0     1     1      -       52   0   - 100 ---
SYSIBM    SYSCHECKS                -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCODEPROPERTIES        -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLAUTH               -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLCHECKS             -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLDIST               -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLGROUPDIST          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLGROUPDISTCO>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLGROUPS             -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLGROUPSCOLS         -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLOPTIONS            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLPROPERTIES         -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCOLUMNS            2812    34   141   143      -   553964   1  95  98 ---
SYSIBM    SYSCOLUSE                4     0     1     1      -      156   0   - 100 ---
SYSIBM    SYSCOMMENTS              -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSCONSTDEP              -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSDATATYPES            17     0     3     3      -    14399   0 100 100 ---
SYSIBM    SYSDBAUTH                2     0     1     1      -       92   0   - 100 ---
SYSIBM    SYSDEPENDENCIES          6     0     1     1      -      468   0   - 100 ---
SYSIBM    SYSEVENTMONITORS         -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSEVENTS                -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSEVENTTABLES           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSFUNCMAPOPTIONS        -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSFUNCMAPPARMOPT>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSFUNCMAPPINGS          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSHIERARCHIES           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSINDEXAUTH             6     0     1     1      -      408   0   - 100 ---
SYSIBM    SYSINDEXCOLUSE         506     0     7     7      -    23782   0  97 100 ---
SYSIBM    SYSINDEXES             188    89    17    31      -   159988  47 100  54 *-*
SYSIBM    SYSINDEXEXPLOITRU>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSINDEXEXTENSION>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSINDEXEXTENSION>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSINDEXEXTENSIONS       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSINDEXOPTIONS          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSJARCONTENTS           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSJAROBJECTS            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSKEYCOLUSE             -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSLIBRARIES             -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSLIBRARYAUTH           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSLIBRARYBINDFIL>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSLIBRARYVERSIONS       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSNAMEMAPPINGS          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSNODEGROUPDEF          2     0     1     1      -       60   0   - 100 ---
SYSIBM    SYSNODEGROUPS            3     0     1     1      -      174   0   - 100 ---
SYSIBM    SYSPARTITIONMAPS         4     0     1     1      -      160   0   - 100 ---
SYSIBM    SYSPASSTHRUAUTH          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSPLAN                 81     0     7     7      -    66987   0 100 100 ---
SYSIBM    SYSPLANAUTH            159     0     3     3      -     9858   0 100 100 ---
SYSIBM    SYSPLANDEP             112     0     3     3      -    20720   0 100 100 ---
SYSIBM    SYSPREDICATESPECS        -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSPROCOPTIONS           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSPROCPARMOPTIONS       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSRELS                  -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSROUTINEAUTH          58     0     2     2      -     5046   0 100 100 ---
SYSIBM    SYSROUTINEPARMS       1077     0    42    42      -   161550   0  96 100 ---
SYSIBM    SYSROUTINEPROPERT>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSROUTINES            195     0    21    21      -   134160   0 100 100 ---
SYSIBM    SYSSCHEMAAUTH            2     0     1     1      -      100   0   - 100 ---
SYSIBM    SYSSCHEMATA              7     0     1     1      -      427   0   - 100 ---
SYSIBM    SYSSECTION             133     0     5     5      -    55195   0 100 100 ---
SYSIBM    SYSSEQUENCEAUTH          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSSEQUENCES             -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSSERVEROPTIONS         -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSSERVERS               -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSSTMT                133     0     5     5      -    51205   0 100 100 ---
SYSIBM    SYSTABAUTH             242     0     5     5      -    17666   0 100 100 ---
SYSIBM    SYSTABCONST              -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSTABLES              243     0    27    27      -   388071   0 100 100 ---
SYSIBM    SYSTABLESPACES           3     0     1     1      -      321   0   - 100 ---
SYSIBM    SYSTABOPTIONS            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSTBSPACEAUTH           1     0     1     1      -       54   0   - 100 ---
SYSIBM    SYSTRANSFORMS            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSTRIGGERS              -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSTYPEMAPPINGS          -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSUSERAUTH            242     0     8     8      -    52998   0 100 100 ---
SYSIBM    SYSUSEROPTIONS           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSVERSIONS              1     0     1     1      -       36   0   - 100 ---
SYSIBM    SYSVIEWDEP             214     0     5     5      -    18404   0 100 100 ---
SYSIBM    SYSVIEWS               144     0     7     7      -    37872   0 100 100 ---
SYSIBM    SYSWRAPOPTIONS           -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSWRAPPERS              -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLOBJECTAUTH         -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLOBJECTAUTHP>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLOBJECTPROPE>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLOBJECTRELDEP       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLOBJECTS            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLOBJECTXMLDEP       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLPHYSICALCOL>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLQUERIES            -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLRELATIONSHI>       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLRSPROPERTIES       -     -     -     -      -        -   -   -   - ---
SYSIBM    SYSXMLSTATS              -     -     -     -      -        -   -   -   - ---
----------------------------------------------------------------------------------------

                       Index statistics:

                       F4: CLUSTERRATIO or normalized CLUSTERFACTOR > 80
                       F5: 100 * (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) / ((NLEAF - NUM_EMPTY_LEAFS)
                       * INDEXPAGESIZE) > 50
                       F6: (100 - PCTFREE) * ((INDEXPAGESIZE - 96) / (ISIZE + 12)) ** (NLEVELS - 2)
                       * (INDEXPAGESIZE - 96) / (KEYS * (ISIZE + 9) + (CARD - KEYS) * 5) < 100
                       F7: 100 * (NUMRIDS DELETED / (NUMRIDS DELETED + CARD)) < 20
                       F8: 100 * (NUM EMPTY LEAFS / NLEAF) < 20
```

```
SCHEMA   NAME                 CARD  LEAF ELEAF  LVLS ISIZE  NDEL   KEYS  F4  F5  F6  F7  F8 REORG
-------------------------------------------------------------------------------------------------
Table: SYSIBM.SYSATTRIBUTES
SYSIBM   IBM83                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM84                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM85                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSBUFFERPOOLNODES
SYSIBM   IBM69                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSBUFFERPOOLS
SYSIBM   IBM67                   1     1     0     1    22     0      1 100   -   -   0   0 -----
SYSIBM   IBM68                   1     1     0     1    10     0      1 100   -   -   0   0 -----
Table: SYSIBM.SYSCHECKS
SYSIBM   IBM37                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCODEPROPERTIES
SYSIBM   IBM161                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLAUTH
SYSIBM   IBM42                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM43                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM64                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLCHECKS
SYSIBM   IBM38                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM39                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLDIST
SYSIBM   IBM46                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLGROUPDIST
SYSIBM   IBM157                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLGROUPDISTCOUNTS
SYSIBM   IBM158                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLGROUPS
SYSIBM   IBM154                  -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM155                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLGROUPSCOLS
SYSIBM   IBM156                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLOPTIONS
SYSIBM   IBM89                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLPROPERTIES
SYSIBM   IBM79                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM80                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM82                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSCOLUMNS
SYSIBM   IBM01                2812    59     0     2    41    12   2812  95  58   2   0   0 -----
SYSIBM   IBM24                2812     7     0     2    23     3     11  82  50  25   0   0 -*---
Table: SYSIBM.SYSCOLUSE
SYSIBM   IBM146                  4     1     0     1    22     1      4 100   -   -  20   0 ---*-
SYSIBM   IBM147                  4     1     0     1    23     1      4 100   -   -  20   0 ---*-
Table: SYSIBM.SYSCOMMENTS
SYSIBM   IBM73                   -     -     -     -     -     -      -   -   -   -   -   - -----
```

```
Table: SYSIBM.SYSCONSTDEP
SYSIBM   IBM44                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM45                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSDATATYPES
SYSIBM   IBM40                  17     1     0     1    23     0     17 100   -   -   0   0 -----
SYSIBM   IBM41                  17     1     0     1     2     0     17 100   -   -   0   0 -----
SYSIBM   IBM56                  17     1     0     1    23     0     17 100   -   -   0   0 -----
Table: SYSIBM.SYSDBAUTH
SYSIBM   IBM12                   2     1     0     1    25     0      2 100   -   -   0   0 -----
Table: SYSIBM.SYSDEPENDENCIES
SYSIBM   IBM51                   6     1     0     1    66     0      6 100   -   -   0   0 -----
SYSIBM   IBM52                   6     1     0     1    25     0      5 100   -   -   0   0 -----
Table: SYSIBM.SYSEVENTMONITORS
SYSIBM   IBM47                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSEVENTS
SYSIBM   IBM48                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSEVENTTABLES
SYSIBM   IBM165                  -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM167                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSFUNCMAPOPTIONS
SYSIBM   IBM90                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSFUNCMAPPARMOPTIONS
SYSIBM   IBM91                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSFUNCMAPPINGS
SYSIBM   IBM92                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM93                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM94                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM95                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM96                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSHIERARCHIES
SYSIBM   IBM86                   -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM87                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSINDEXAUTH
SYSIBM   IBM17                   6     2     0     2    57    96      6 100   4 909 100   0 -***-
SYSIBM   IBM18                   6     2     0     2    32   151      6 100   3 +++ 100   0 -*-*-
Table: SYSIBM.SYSINDEXCOLUSE
SYSIBM   IBM139                506     9     0     2    36   114    506  96  61  15  20   0 ---*-
Table: SYSIBM.SYSINDEXES
SYSIBM   IBM02                 188     4     1     2    22    40    188  80  47  61  20   0 **-*-
SYSIBM   IBM03                 188     3     0     2    31     8    188  77  61  47   4   0 *----
SYSIBM   IBM126                188     1     0     1    12     4      1 100   -   -   2   0 -----
Table: SYSIBM.SYSINDEXEXPLOITRULES
SYSIBM   IBM97                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSINDEXEXTENSIONMETHODS
SYSIBM   IBM100                  -     -     -     -     -     -      -   -   -   -   -   - -----
SYSIBM   IBM101                  -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSINDEXEXTENSIONPARMS
SYSIBM   IBM98                   -     -     -     -     -     -      -   -   -   -   -   - -----
Table: SYSIBM.SYSINDEXEXTENSIONS
SYSIBM   IBM99                   -     -     -     -     -     -      -   -   -   -   -   - -----
...
-------------------------------------------------------------------------------------------------
```

```
CLUSTERRATIO or normalized CLUSTERFACTOR (F4) will indicate REORG is necessary
for indexes that are not in the same sequence as the base table. When multiple
indexes are defined on a table, one or more indexes might be flagged as needing
REORG.  Specify the most important index for REORG sequencing.

Tables defined using the DIMENSION clause and the corresponding dimension
indexes have a '*' suffix to their names. The cardinality of a dimension index
is equal to the Active blocks statistic of the table.
```

The terms for the table statistics (formulas 1-3) mean:

**CARD**

(CARDINALITY) Number of rows in base table.

**OV** (OVERFLOW) Number of overflow rows.

**NP** (NPAGES) Number of pages that contain data.

**FP** (FPAGES) Total number of pages.

**ACTBLK**
Total number of active blocks for a multidimensional clustering (MDC) table. This field is only applicable to tables defined using the ORGANIZE BY clause. It indicates the number of blocks of the table that contain data.

**TSIZE** Table size in bytes. Calculated as the product of the number of rows in the table (CARD) and the average row length. The average row length is computed as the sum of the average column lengths (AVGCOLLEN in SYSCOLUMNS) plus 10 bytes of row overhead. For long fields and LOBs only the approximate length of the descriptor is used. The actual long field or LOB data is not counted in TSIZE.

**TABLEPAGESIZE**
Page size of the table space in which the table data resides.

**F1** Results of Formula 1.

**F2** Results of Formula 2.

**F3** Results of Formula 3. This formula indicates the amount of space that is wasted in a table. This is measured in terms of the number of empty pages and the number of pages that include data that exists in the pages of a table. In multi-dimensional clustering (MDC) tables, the number of empty blocks and the number of blocks that include data is measured.

**REORG**
Each hyphen (-) displayed in this column indicates that the calculated results were within the set bounds of the corresponding formula, and each asterisk (*) indicates that the calculated results exceeded the set bounds of its corresponding formula.

- `-` or `*` on the left side of the column corresponds to F1 (Formula 1)
- `-` or `*` in the middle of the column corresponds to F2 (Formula 2)
- `-` or `*` on the right side of the column corresponds to F3 (Formula 3).

Table reorganization is suggested when the results of the calculations exceed the bounds set by the formula.

For example, `---` indicates that, since the formula results of F1, F2, and F3 are within the set bounds of the formula, no table reorganization is suggested. The notation `*-*` indicates that the results of F1 and F3 suggest table reorganization, even though F2 is still within its set bounds. The notation `*--` indicates that F1 is the only formula exceeding its bounds.

**Note:** The table name is truncated to 30 characters, and the ″>″ symbol in the thirty-first column represents the truncated portion of the table name. An “*” suffix to a table name indicates it is an MDC table. An “*” suffix to an index name indicates it is an MDC dimension index.

The terms for the index statistics (formulas 4-8) mean:

**CARD**
Number of rows in base table.

**LEAF** Total number of index leaf pages (NLEAF).

**ELEAF**
Number of pseudo empty index leaf pages (NUM_EMPTY_LEAFS)

A pseudo empty index leaf page is a page on which all the RIDs are marked as deleted, but have not been physically removed.

**NDEL** Number of pseudo deleted RIDs (NUMRIDS_DELETED)

A pseudo deleted RID is a RID that is marked deleted. This statistic reports pseudo deleter RIDs on leaf pages that are not pseudo empty. It does not include RIDs marked as deleted on leaf pages where all the RIDs are marked deleted.

**LVLS** Number of index levels (NLEVELS)

**ISIZE** Index size, calculated from the average column length of all columns participating in the index.

**KEYS** Number of unique index entries that are not marked deleted (FULLKEYCARD)

**INDEXPAGESIZE**
Page size of the table space in which the table indexes reside, specified at the time of table creation. If not specified, INDEXPAGESIZE has the same value as TABLEPAGESIZE.

**PCTFREE**
Specifies the percentage of each index page to leave as free space, a value that is assigned when defining the index. Values can range from 0 to 99. The default value is 10.

**F4** Results of Formula 4.

**F5** Results of Formula 5. The notation +++ indicates that the result exceeds 999, and is invalid. Rerun REORGCHK with the UPDATE STATISTICS option, or issue RUNSTATS, followed by the REORGCHK command.

**F6** Results of Formula 6. The notation +++ indicates that the result exceeds 9999, and might be invalid. Rerun REORGCHK with the UPDATE STATISTICS option, or issue RUNSTATS, followed by the REORGCHK command. If the statistics are current and valid, you should reorganize.

**F7** Results of Formula 7.

**F8** Results of Formula 8.

**REORG**
Each hyphen (-) displayed in this column indicates that the calculated results were within the set bounds of the corresponding formula, and each asterisk (*) indicates that the calculated result exceeded the set bounds of its corresponding formula.

- - or * on the left column corresponds to F4 (Formula 4)
- - or * in the second from left column corresponds to F5 (Formula 5)
- - or * in the middle column corresponds to F6 (Formula 6).
- - or * in the second column from the right corresponds to F7 (Formula 7)
- - or * on the right column corresponds to F8 (Formula 8).

Index reorganization advice is as follows:

- If the results of the calculations for Formula 1,2 and 3 do not exceed the bounds set by the formula and the results of the calculations for Formula 4,5 or 6 do exceed the bounds set, then index reorganization is recommended.
- If only the results of the calculations Formula 7 exceed the bounds set, but the results of Formula 1,2,3,4,5 and 6 are within the set bounds, then cleanup of the indexes using the CLEANUP ONLY option of REORG INDEXES is recommended.
- If the only calculation result to exceed the set bounds is the that of Formula 8, then a cleanup of the pseudo empty pages of the indexes using the CLEANUP ONLY PAGES option of REORG INDEXES is recommended.

**Usage notes:**

This command does not display declared temporary table statistical information.

This utility does not support the use of nicknames.

Unless you specify the CURRENT STATISTICS option, REORGCHK gathers statistics on all columns using the default options only. Specifically, column group are not gathered and if LIKE statistics were previously gathered, they are not gathered by REORGCHK.

The statistics gathered depend on the kind of statistics currently stored in the catalog tables:

- If detailed index statistics are present in the catalog for any index, table statistics and detailed index statistics (without sampling) for all indexes are collected.
- If detailed index statistics are not detected, table statistics as well as regular index statistics are collected for every index.
- If distribution statistics are detected, distribution statistics are gathered on the table. If distribution statistics are gathered, the number of frequent values and quantiles are based on the database configuration parameter settings.

REORGCHK calculates statistics obtained from eight different formulas to determine if performance has deteriorated or can be improved by reorganizing a table or its indexes.

**Attention:** These statistics should not be used to determine if empty tables (TSIZE=0) need reorganization. If TSIZE=0 and FPAGE>0, the table needs to be reorganized. If TSIZE=0 and FPAGE=0, no reorganization is necessary.

REORGCHK uses the following formulas to analyze the physical location of rows and the size of the table:

- Formula F1:

  ```
  100*OVERFLOW/CARD < 5
  ```

  The total number of overflow rows in the table should be less than 5 percent of the total number of rows. Overflow rows can be created when rows are updated and the new rows contain more bytes than the old ones (VARCHAR fields), or when columns are added to existing tables.
- Formula F2:

  For regular tables:

  ```
  100*TSIZE / ((FPAGES-1) * (TABLEPAGESIZE-76)) > 68
  ```

The table size in bytes (TSIZE) should be more than 68 percent of the total space allocated for the table. (There should be less than 32% free space.) The total space allocated for the table depends upon the page size of the table space in which the table resides (minus an overhead of 76 bytes). Because the last page allocated is not usually filled, 1 is subtracted from FPAGES.

For MDC tables:

```
100*TSIZE / ((ACTBLK-FULLKEYCARD) * EXTENTSIZE * (TABLEPAGESIZE-76)) > 68
```

FULLKEYCARD represents the cardinality of the composite dimension index for the MDC table. Extentsize is the number of pages per block. The formula checks if the table size in bytes is more than the 68 percent of the remaining blocks for a table after subtracting the minimum required number of blocks.

- Formula F3:

  ```
  100*NPAGES/FPAGES > 80
  ```

  The number of pages that contain no rows at all should be less than 20 percent of the total number of pages. (Pages can become empty after rows are deleted.)

  For MDC tables, the formla is:

  ```
  100 * activeblocks  /  ( ( fpages /  ExtentSize ) - 1 )
  ```

REORGCHK uses the following formulas to analyze the indexes and their the relationship to the table data:

- Formula F4:

  ```
  CLUSTERRATIO or normalized CLUSTERFACTOR > 80
  ```

  The clustering ratio of an index should be greater than 80 percent. When multiple indexes are defined on one table, some of these indexes have a low cluster ratio. (The index sequence is not the same as the table sequence.) This cannot be avoided. Be sure to specify the most important index when reorganizing the table. The cluster ratio is usually not optimal for indexes that contain many duplicate keys and many entries.

- Formula F5:

  ```
  100*(KEYS*(ISIZE+9)+(CARD-KEYS)*5) /
     ((NLEAF-NUM_EMPTY_LEAFS)*INDEXPAGESIZE) > 50
  ```

  Less than 50 percent of the space reserved for index entries should be empty (only checked when NLEAF>1).

- Formula F6:

  ```
  (100-PCTFREE)*((INDEXPAGESIZE-96)/(ISIZE+12))**
      (NLEVELS-2)*(INDEXPAGESIZE-96) / (KEYS*(ISIZE+9)+(CARD-KEYS)*5)
      < 100
  ```

  To determine if recreating the index would result in a tree having fewer levels. This formula checks the ratio between the amount of space in an index tree that has one less level than the current tree, and the amount of space needed. If a tree with one less level could be created and still leave PCTFREE available, then a reorganization is recommended. The actual number of index entries should be more than 90% (or 100-PCTFREE) of the number of entries an NLEVELS-1 index tree can handle (only checked if NLEVELS>1).

- Formula F7:

  ```
  100 * (NUMRIDS_DELETED / (NUMRIDS_DELETED + CARD)) < 20
  ```

  The number of pseudo-deleted RIDs on non-pseudo-empty pages should be less than 20 percent.

- Formula F8:

  ```
  100 * (NUM_EMPTY_LEAFS/NLEAF) < 20
  ```

The number of pseudo-empty leaf pages should be less than 20 percent of the total number of leaf pages.

**Note:** Running statistics on many tables can take time, especially if the tables are large.

**Related reference:**

- "REORG INDEXES/TABLE" on page 617
- "RUNSTATS" on page 667

# RESET ADMIN CONFIGURATION

Resets entries in the DB2 Administration Server (DAS) configuration file on the node to which you are connected. The DAS is a special administrative tool that enables remote administration of DB2 servers. The values are reset by node type, which is always a server with remote clients. For a list of DAS parameters, see the description of the UPDATE ADMINISTRATION CONFIGURATION command.

**Scope:**

This command resets the DAS configuration file on the administration node of the system to which you are connected.

**Authorization:**

*dasadm*

**Required connection:**

Partition. To reset the DAS configuration for a remote system, specify the system using the FOR NODE option with the administration node name.

**Command syntax:**

```
►►──RESET ADMIN───┬─CONFIGURATION─┬──────────────────────────────────────────────►
                  ├─CONFIG────────┤
                  └─CFG───────────┘

►──┬────────────────────────────────────────────────────────────┬──────────────►◄
   └─FOR NODE──node-name──┬──────────────────────────────────┬─┘
                          └─USER──username──USING──password─┘
```

**Command parameters:**

**FOR NODE**
Enter the name of an administrator node to reset DAS configuration parameters there.

**USER username USING** *password*
If connection to the remote system requires user name and password, enter this information.

**Usage notes:**

To reset the DAS configuration parameters on a remote system, specify the system using the administrator node name as an argument to the FOR NODE option and specify the user name and password if the connection to that node requires username and password authorization.

To view or print a list of the DAS configuration parameters, use the GET ADMIN CONFIGURATION command. To change the value of an admin parameter, use the UPDATE ADMIN CONFIGURATION command.

Changes to the DAS configuration parameters that can be updated on-line take place immediately. Other changes become effective only after they are loaded into memory when you restart the DAS with the **db2admin** command.

If an error occurs, the DAS configuration file does not change.

The DAS configuration file cannot be reset if the checksum is invalid. This might occur if you edit the DAS configuration file manually and do not use the appropriate command. If the checksum is invalid, you must drop and recreate the DAS to reset the its configuration file.

**Related reference:**

- "GET ADMIN CONFIGURATION" on page 374
- "UPDATE ADMIN CONFIGURATION" on page 715
- "Configuration parameters summary" in the *Administration Guide: Performance*

# RESET ALERT CONFIGURATION

Resets the health indicator settings for specific objects to the current defaults for that object type or resets the current default health indicator settings for an object type to the install defaults.

**Authorization:**

One of the following:

- sysadm
- sysmaint
- sysctrl

**Required connection:**

Instance. An explicit attachment is not required.

**Command syntax:**

```
►►──RESET ALERT──┬─CONFIGURATION─┬──FOR──────────────────────────────────────────────►
                 ├─CONFIG────────┤
                 └─CFG───────────┘

►──┬─┬─┬─DATABASE MANAGER─┬─┬─────────────────────────────────────┬──────────────────►
   │ │ ├─DB MANAGER───────┤ │                                     │
   │ │ └─DBM──────────────┘ │                                     │
   │ ├─DATABASES────────────┤                                     │
   │ ├─TABLESPACES──────────┤                                     │
   │ └─CONTAINERS───────────┘                                     │
   └─┬─DATABASE───────────────────────────────┬─ON──database alias─┘
     ├─TABLESPACE──name───────────────────────┤
     └─CONTAINER──name──FOR──tablespace-id────┘

►──┬─────────────────────────────────┬────────────────────────────────────────────►◄
   └─USING──health indicator name────┘
```

**Command parameters:**

**DATABASE MANAGER**
Resets alert settings for the database manager.

**DATABASES**
Resets alert settings for all databases managed by the database manager. These are the settings that apply to all databases that do not have custom settings. Custom settings are defined using the DATABASE ON *database alias* clause.

**CONTAINERS**
Resets default alert settings for all table space containers managed by the database manager to the install default. These are the settings that apply to all table space containers that do not have custom settings. Custom settings are defined using the ″CONTAINER *name* ON *database alias*″ clause.

**CONTAINER** *name* **FOR** *tablespace-id* **FOR** *tablespace-id* **ON** *database alias*
Resets the alert settings for the table space container called *name*, for the table space specified using the ″FOR *tablespace-id*″ clause, on the database

specified using the "ON *database alias*" clause. If this table space container has custom settings, then these settings are removed and the current table space containers default is used.

**TABLESPACES**
Resets default alert settings for all table spaces managed by the database manager to the install default. These are the settings that apply to all table spaces that do not have custom settings. Custom settings are defined using the "TABLESPACE *name* ON *database alias*" clause.

**DATABASE ON** *database alias*
Resets the alert settings for the database specified using the ON *database alias* clause. If this database has custom settings, then these settings are removed and the install default is used.

**BUFFERPOOL** *name* **ON** *database alias*
Resets the alert settings for the buffer pool called *name*, on the database specified using the ON *database alias* clause. If this buffer pool has custom settings, then these settings are removed and the install default is used.

**TABLESPACE** *name* **ON** *database alias*
Resets the alert settings for the table space called *name*, on the database specified using the ON *database alias* clause. If this table space has custom settings, then these settings are removed and the install default is used.

**USING** *health indicator name*
Specifies the set of health indicators for which alert configuration will be reset. Health indicator names consist of a two-letter object identifier followed by a name that describes what the indicator measures. For example:

```
db.sort_privmem_util
```

If you do not specify this option, all health indicators for the specified object or object type will be reset.

# RESET DATABASE CONFIGURATION

Resets the configuration of a specific database to the system defaults.

**Scope:**

This command only affects the node on which it is executed.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance. An explicit attachment is not required. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command.

**Command syntax:**

```
►►──RESET──┬─DATABASE─┬──┬─CONFIGURATION─┬──FOR──database-alias──────────────►◄
           └─DB───────┘  ├─CONFIG────────┤
                         └─CFG───────────┘
```

**Command parameters:**

**FOR database-alias**
: Specifies the alias of the database whose configuration is to be reset to the system defaults.

**Usage notes:**

To view or print a list of the database configuration parameters, use the GET DATABASE CONFIGURATION command.

To change the value of a configurable parameter, use the UPDATE DATABASE CONFIGURATION command.

Changes to the database configuration file become effective only after they are loaded into memory. All applications must disconnect from the database before this can occur.

If an error occurs, the database configuration file does not change.

The database configuration file cannot be reset if the checksum is invalid. This might occur if the database configuration file is changed without using the appropriate command. If this happens, the database must be restored to reset the database configuration file.

**Related tasks:**

- “Configuring DB2 with configuration parameters” in the *Administration Guide: Performance*

**Related reference:**

- "GET DATABASE CONFIGURATION" on page 389
- "UPDATE DATABASE CONFIGURATION" on page 730
- "Configuration parameters summary" in the *Administration Guide: Performance*

# RESET DATABASE MANAGER CONFIGURATION

Resets the parameters in the database manager configuration file to the system defaults. The values are reset by node type.

**Authorization:**

*sysadm*

**Required connection:**

None or instance. An instance attachment is not required to perform local database manage configuration operations, but is required to perform remote database manager configuration operations. To update the database manager configuration for a remote instance, it is necessary to first attach to that instance. To update a configuration parameter on-line, it is also necessary to first attach to the instance.

**Command syntax:**

```
►►──RESET──┬─DATABASE MANAGER─┬──┬─CONFIGURATION─┬─────────────────────────►◄
           ├─DB MANAGER───────┤  ├─CONFIG────────┤
           └─DBM──────────────┘  └─CFG───────────┘
```

**Command parameters:**
None
**Usage notes:**
It is important to note that this command resets all parameters set by the installation program. This could cause error messages to be returned when restarting DB2. For example, if the SVCENAME parameter is reset, the user will receive the SQL5043N error message when trying to restart DB2.

Before running this command, save the output from the GET DATABASE MANAGER CONFIGURATION command to a file so that you can refer to the existing settings. Individual settings can then be updated using the UPDATE DATABASE MANAGER CONFIGURATION command.

**Note:** It is not recommended that the SVCENAME parameter, set by the installation program, be modified by the user. The administration server service name is set to use the DB2 registered TCP/IP port (523).

To view or print a list of the database manager configuration parameters, use the GET DATABASE MANAGER CONFIGURATION command. To change the value of a configurable parameter, use the UPDATE DATABASE MANAGER CONFIGURATION command.

For more information about these parameters, refer to the summary list of configuration parameters and the individual parameters.

Some changes to the database manager configuration file become effective only after they are loaded into memory. For more information on which parameters are configurable on-line and which ones are not, see the configuration parameter summary. Server configuration parameters that are not reset immediately are reset during execution of **db2start**. For a client configuration parameter, parameters are reset the next time you restart the application. If the client is the command line processor, it is necessary to invoke TERMINATE.

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be reset if the checksum is invalid. This might occur if the database manager you edit the configuration file manually and do not use the appropriate command. If the checksum is invalid, you must reinstall the database manager to reset the database manager configuration file.

**Related tasks:**

- "Configuring DB2 with configuration parameters" in the *Administration Guide: Performance*

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "TERMINATE" on page 705
- "UPDATE DATABASE MANAGER CONFIGURATION" on page 733
- "Configuration parameters summary" in the *Administration Guide: Performance*

# RESET MONITOR

Resets the internal database system monitor data areas of a specified database, or of all active databases, to zero. The internal database system monitor data areas include the data areas for all applications connected to the database, as well as the data areas for the database itself.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance. If there is no instance attachment, a default instance attachment is created.

To reset the monitor switches for a remote instance (or a different local instance), it is necessary to first attach to that instance.

**Command syntax:**

```
►►──RESET MONITOR──┬─ALL──┬─────┬───────────────────────────────┬──────────►
                   │      └─DCS─┘                               │
                   └─FOR──┬─────┬──┬─DATABASE─┬──database-alias─┘
                          └─DCS─┘  └─DB───────┘

►──┬───────────────────────────────────────┬───────────────────────────────►◄
   ├─AT DBPARTITIONNUM──db-partition-number─┤
   └─GLOBAL─────────────────────────────────┘
```

**Command parameters:**

**ALL**
This option indicates that the internal counters should be reset for all databases.

**FOR DATABASE database-alias**
This option indicates that only the database with alias *database-alias* should have its internal counters reset.

**DCS**
Depending on which clause it is specified, this keyword resets the internal counters of:

- All DCS databases
- A specific DCS database.

**AT DBPARTITIONNUM db-partition-number**
Specifies the database partition for which the status of the monitor switches is to be displayed.

**GLOBAL**
Returns an aggregate result for all database partitions in a partition database system.

**Usage notes:**

Each process (attachment) has its own private view of the monitor data. If one user resets, or turns off a monitor switch, other users are not affected. Change the setting of the monitor switch configuration parameters to make global changes to the monitor switches.

If ALL is specified, some database manager information is also reset to maintain consistency of the returned data, and some partition-level counters are reset.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "GET SNAPSHOT" on page 419
- "GET MONITOR SWITCHES" on page 410
- "UPDATE DATABASE MANAGER CONFIGURATION" on page 733

# RESTART DATABASE

Restarts a database that has been abnormally terminated and left in an inconsistent state. At the successful completion of RESTART DATABASE, the application remains connected to the database if the user has CONNECT privilege.

**Scope:**

This command affects only the node on which it is executed.

**Authorization:**

None

**Required connection:**

This command establishes a database connection.

**Command syntax:**

```
►►──RESTART──┬─DATABASE─┬──database-alias────────────────────────────────►
             └─DB───────┘

►──┬─────────────────────────────────────┬────────────────────────────────►
   └─USER──username──┬──────────────────┬─┘
                     └─USING──password──┘

►──┬───────────────────────────────────────────────────────┬──┬──────────────┬──►◄
   │                               ┌─────────────────┐      │  └─WRITE RESUME─┘
   │                               ▼                 │      │
   └─DROP PENDING TABLESPACES──(────tablespace-name──┴──)───┘
```

**Command parameters:**

**DATABASE database-alias**
: Identifies the database to restart.

**USER username**
: Identifies the user name under which the database is to be restarted.

**USING password**
: The password used to authenticate *username*. If the password is omitted, the user is prompted to enter it.

**DROP PENDING TABLESPACES tablespace-name**
: Specifies that the database restart operation is to be successfully completed even if table space container problems are encountered.

If a problem occurs with a container for a specified table space during the restart process, the corresponding table space will not be available (it will be in drop-pending state) after the restart operation. If a table space is in the drop-pending state, the only possible action is to drop the table space.

In the case of circular logging, a troubled table space will cause a restart failure. A list of troubled table space names can found in the administration notification log if a restart database operation fails because of container problems. If there is only one system temporary table space in the database, and it is in drop pending state, a new system temporary table space must be created immediately following a successful database restart operation.

**WRITE RESUME**

Allows you to force a database restart on databases that failed while I/O writes were suspended. Before performing crash recovery, this option will resume I/O writes by removing the SUSPEND_WRITE state from every table space in the database.

The WRITE RESUME option can also be used in the case where the connection used to suspend I/O writes is currently hung and all subsequent connection attempts are also hanging. When used in this circumstance, RESTART DATABASE will resume I/O writes to the database without performing crash recovery. RESTART DATABASE with the WRITE RESUME option will only perform crash recovery when you use it after a database crash.

**Note:** The WRITE RESUME parameter can only be applied to the primary database, not to mirrored databases.

**Usage notes:**

Execute this command if an attempt to connect to a database returns an error message, indicating that the database must be restarted. This action occurs only if the previous session with this database terminated abnormally (due to power failure, for example).

At the completion of RESTART DATABASE, a shared connection to the database is maintained if the user has CONNECT privilege, and an SQL warning is issued if any indoubt transactions exist. In this case, the database is still usable, but if the indoubt transactions are not resolved before the last connection to the database is dropped, another RESTART DATABASE must be issued before the database can be used again. Use the LIST INDOUBT TRANSACTIONS command to generate a list of indoubt transactions.

If the database is only restarted on a single node within an MPP system, a message might be returned on a subsequent database query indicating that the database needs to be restarted. This occurs because the database partition on a node on which the query depends must also be restarted. Restarting the database on all nodes solves the problem.

**Related tasks:**

- "Manually resolving indoubt transactions" in the *Administration Guide: Planning*

**Related reference:**

- "LIST INDOUBT TRANSACTIONS" on page 500

# RESTORE DATABASE

Rebuilds a damaged or corrupted database that has been backed up using the DB2 backup utility. The restored database is in the same state it was in when the backup copy was made. This utility can also overwrite a database with a different image, or restore to a new database.

You can restore databases created on a DB2 Version 8 32-bit Windows platform to a DB2 Version 8 64-bit Windows platform, or the reverse. You can restore databases created on a DB2 Version 8 32-bit Linux (Intel) platform to a DB2 Version 8 64-bit Linux (Intel) platform, or the reverse. You can restore databases created on DB2 Version 8 AIX, HP-UX, or the Solaris Operating Environment platforms, in 32-bit or 64-bit, to DB2 Version 8 AIX, HP-UX, or Solaris Operating Environment platforms (32-bit or 64-bit).

The restore utility can also be used to restore backup images that were produced on a previous version of DB2 (up to two versions earlier) as long as the word size (32-bit or 64-bit) is the same. Cross-platform restore operations from a backup image created with a previous version of DB2 are not supported. If a migration is required, it will be invoked automatically at the end of the restore operation.

If, at the time of the backup operation, the database was enabled for rollforward recovery, the database can be brought to the state it was in prior to the occurrence of the damage or corruption by invoking the rollforward utility after successful completion of a restore operation.

This utility can also restore a table space level backup.

Incremental and delta images cannot be restored when there is a difference in operating systems or word size (32-bit or 64-bit).

Following a successful restore from one environment to a different environment, no incremental or delta backups are allowed until a non-incremental backup is taken. (This is not a limitation following a restore to the same environment.)

Even with a successful restore from one environment to a different environment, there are some considerations: packages must be rebound before use (using the BIND command, the REBIND command, or the db2rbind utility); SQL procedures must be dropped and recreated; and all external libraries must be rebuilt on the new platform. (These are not considerations when restoring to the same environment.)

**Scope:**

This command only affects the node on which it is executed.

**Authorization:**

To restore to an existing database, one of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

To restore to a new database, one of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

The required connection will vary based on the type of restore action you wish to do:

- Database, to restore to an existing database. This command automatically establishes an exclusive connection to the specified database.
- Instance and database, to restore to a new database. The instance attachment is required to create the database.

  To restore to a new database at an instance different from the current instance, it is necessary to first attach to the instance where the new database will reside. The new instance can be local or remote. The current instance is defined by the value of the DB2INSTANCE environment variable.

**Command syntax:**

```
>>--RESTORE--+-DATABASE-+--source-database-alias--+-| restore-options |-+-------><
             '-DB-------'                         +-CONTINUE------------+
                                                  '-ABORT---------------'
```

**restore-options:**

```
|--+-------------------------------------+------------------------------------------------->
   '-USER--username--+-----------------+-'
                     '-USING--password-'

>--+----------------------------------------------------------------+--+-----------------------------+-->
   '-+-TABLESPACE--+--------------------------------+-+--+--------+-'  '-INCREMENTAL--+-----------+-'
     |             |    .-,------------------.      | |  '-ONLINE-'                   +-AUTO------+
     |             |    V                    |      | |                             +-AUTOMATIC-+
     |             '-(------tablespace-name--+--)---' |                             '-ABORT-----'
     +-HISTORY FILE-----------------------------------+
     +-COMPRESSION LIBRARY----------------------------+
     '-LOGS-------------------------------------------'

>--+-------------------------------------------------------------------------------------------------+-->
   +-USE--+-TSM--+--+------------------------------------+--+---------------------------------+-----+
   |      '-XBSA-'  '-OPTIONS--+-"options-string"-+------'  '-OPEN--num-sessions--SESSIONS----'     |
   |                           '-@--file-name-----'                                                 |
   |        .-,-------------.                                                                       |
   |        V               |                                                                       |
   +-FROM------+-directory-+-+----------------------------------------------------------------------+
   |           '-device----'                                                                        |
   '-LOAD--shared-library--+---------------------------------------+--+-------------------------------+-'
                           '-OPTIONS--+-"options-string"-+---------'  '-OPEN--num-sessions--SESSIONS--'
                                      '-@--file-name-----'

>--+----------------------+--+-------------------------+--+----------------------------------+-->
   '-TAKEN AT--date-time--'  '-TO--target-directory----'  '-INTO--target-database-alias------'

>--+-------------------------+--+-------------------------+--+---------------------------------+-->
   '-LOGTARGET--directory----'  '-NEWLOGPATH--directory---'  '-WITH--num-buffers--BUFFERS-----'

>--+----------------------+--+---------------------+--+----------------------+--+------------------+-->
   '-BUFFER--buffer-size--'  '-DLREPORT--filename--'  '-REPLACE HISTORY FILE-'  '-REPLACE EXISTING-'

>--+----------+--+-----------------+--+----------------+--+--------------------+-->
   '-REDIRECT-'  '-PARALLELISM--n--'  '-COMPRLIB--name-'  '-COMPROPTS--string--'

>--+-------------------------+--+------------------+--+--------------------+--------------------|
   '-WITHOUT ROLLING FORWARD-'  '-WITHOUT DATALINK-'  '-WITHOUT PROMPTING--'
```

**Command parameters:**

**DATABASE** *source-database-alias*
: Alias of the source database from which the backup was taken.

**CONTINUE**
Specifies that the containers have been redefined, and that the final step in a redirected restore operation should be performed.

**ABORT**
This parameter:

- Stops a redirected restore operation. This is useful when an error has occurred that requires one or more steps to be repeated. After RESTORE DATABASE with the ABORT option has been issued, each step of a redirected restore operation must be repeated, including RESTORE DATABASE with the REDIRECT option.
- Terminates an incremental restore operation before completion.

**USER** *username*
Identifies the user name under which the database is to be restored.

**USING** *password*
The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

**TABLESPACE** *tablespace-name*
A list of names used to specify the table spaces that are to be restored.

**ONLINE**
This keyword, applicable only when performing a table space-level restore operation, is specified to allow a backup image to be restored online. This means that other agents can connect to the database while the backup image is being restored, and that the data in other table spaces will be available while the specified table spaces are being restored.

**HISTORY FILE**
This keyword is specified to restore only the history file from the backup image.

**COMPRESSION LIBRARY**
This keyword is specified to restore only the compression library from the backup image. If the object exists in the backup image, it will be restored into the database directory. If the object does not exist in the backup image, the restore operation will fail.

**LOGS** This keyword is specified to restore only the set of log files contained in the backup image. If the backup image does not contain any log files, the restore operation will fail. If this option is specified, the LOGTARGET option must also be specified.

**INCREMENTAL**
Without additional parameters, INCREMENTAL specifies a manual cumulative restore operation. During manual restore the user must issue each restore command manually for each image involved in the restore. Do so according to the following order: last, first, second, third and so on up to and including the last image.

**INCREMENTAL AUTOMATIC/AUTO**
Specifies an automatic cumulative restore operation.

**INCREMENTAL ABORT**
Specifies abortion of an in-progress manual cumulative restore operation.

**USE TSM**
Specifies that the database is to be restored from TSM-managed output.

**OPTIONS**

*"options-string"*
Specifies options to be used for the restore operation.The string will be passed to the vendor support library, for example TSM, exactly as it was entered, without the quotes.

**Note:** Specifying this option overrides the value specified by the VENDOROPT database configuration parameter.

*@file-name*
Specifies that the options to be used for the restore operation are contained in a file located on the DB2 server. The string will be passed to the vendor support library, for example TSM. The file must be a fully qualified file name.

**OPEN** *num-sessions* **SESSIONS**
Specifies the number of I/O sessions that are to be used with TSM or the vendor product.

**USE XBSA**
Specifies that the XBSA interface is to be used. Backup Services APIs (XBSA) are an open application programming interface for applications or facilities needing data storage management for backup or archiving purposes.

**FROM** *directory/device*
The fully qualified path name of the directory or device on which the backup image resides. If USE TSM, FROM, and LOAD are omitted, the default value is the current working directory of the client machine. This target directory or device must exist on the database server.

On Windows operating systems, the specified directory must not be a DB2-generated directory. For example, given the following commands:

```
db2 backup database sample to c:\backup
db2 restore database sample from c:\backup
```

Using these commands, DB2 generates subdirectories under the `c:\backup` directory to allow more than one backup to be placed in the specified top level directory. The DB2-generated subdirectories should be ignored. To specify precisely which backup image to restore, use the TAKEN AT parameter. There can be several backup images stored on the same path.

If several items are specified, and the last item is a tape device, the user is prompted for another tape. Valid response options are:

**c** Continue. Continue using the device that generated the warning message (for example, continue when a new tape has been mounted).

**d** Device terminate. Stop using *only* the device that generated the warning message (for example, terminate when there are no more tapes).

**t** Terminate. Abort the restore operation after the user has failed to perform some action requested by the utility.

**LOAD** *shared-library*
The name of the shared library (DLL on Windows operating systems) containing the vendor backup and restore I/O functions to be used. The name can contain a full path. If the full path is not given, the value defaults to the path on which the user exit program resides.

**TAKEN AT** *date-time*

The time stamp of the database backup image. The time stamp is displayed after successful completion of a backup operation, and is part of the path name for the backup image. It is specified in the form *yyyymmddhhmmss*. A partial time stamp can also be specified. For example, if two different backup images with time stamps 20021001010101 and 20021002010101 exist, specifying `20021002` causes the image with time stamp `20021002010101` to be used. If a value for this parameter is not specified, there must be only one backup image on the source media.

**TO** *target-directory*

The target database directory. This parameter is ignored if the utility is restoring to an existing database. The drive and directory that you specify must be local.

**Note:** On Windows operating systems, when using this parameter, specify only the drive letter. If you specify a path, an error is returned.

**INTO** *target-database-alias*

The target database alias. If the target database does not exist, it is created.

When you restore a database backup to an existing database, the restored database inherits the alias and database name of the existing database. When you restore a database backup to a nonexistent database, the new database is created with the alias and database name that you specify. This new database name must be unique on the system where you restore it.

**LOGTARGET** *directory*

The absolute path name of an existing directory on the database server, to be used as the target directory for extracting log files from a backup image. If this option is specified, any log files contained within the backup image will be extracted into the target directory. If this option is not specified, log files contained within a backup image will not be extracted. To extract only the log files from the backup image, specify the LOGS option.

**NEWLOGPATH** *directory*

The absolute pathname of a directory that will be used for active log files after the restore operation. This parameter has the same function as the *newlogpath* database configuration parameter, except that its effect is limited to the restore operation in which it is specified. The parameter can be used when the log path in the backup image is not suitable for use after the restore operation; for example, when the path is no longer valid, or is being used by a different database.

**WITH** *num-buffers* **BUFFERS**

The number of buffers to be used. DB2 will automatically choose an optimal value for this parameter unless you explicitly enter a value. A larger number of buffers can be used to improve performance when multiple sources are being read from, or if the value of PARALLELISM has been increased.

**BUFFER** *buffer-size*

The size, in pages, of the buffer used for the restore operation. DB2 will automatically choose an optimal value for this parameter unless you explicitly enter a value. The minimum value for this parameter is 8 pages.

The restore buffer size must be a positive integer multiple of the backup buffer size specified during the backup operation. If an incorrect buffer size is specified, the buffers are allocated to be of the smallest acceptable size.

**DLREPORT** *filename*

The file name, if specified, must be specified as an absolute path. Reports the files that become unlinked, as a result of a fast reconcile, during a restore operation. This option is only to be used if the table being restored has a DATALINK column type and linked files.

**REPLACE HISTORY FILE**

Specifies that the restore operation should replace the history file on disk with the history file from the backup image.

**REPLACE EXISTING**

If a database with the same alias as the target database alias already exists, this parameter specifies that the restore utility is to replace the existing database with the restored database. This is useful for scripts that invoke the restore utility, because the command line processor will not prompt the user to verify deletion of an existing database. If the WITHOUT PROMPTING parameter is specified, it is not necessary to specify REPLACE EXISTING, but in this case, the operation will fail if events occur that normally require user intervention.

**REDIRECT**

Specifies a redirected restore operation. To complete a redirected restore operation, this command should be followed by one or more SET TABLESPACE CONTAINERS commands, and then by a RESTORE DATABASE command with the CONTINUE option.

**Note:** All commands associated with a single redirected restore operation must be invoked from the same window or CLP session.

**WITHOUT ROLLING FORWARD**

Specifies that the database is not to be put in rollforward pending state after it has been successfully restored.

If, following a successful restore operation, the database is in rollforward pending state, the ROLLFORWARD command must be invoked before the database can be used again.

If this option is specified when restoring from an online backup image, error SQL2537N will be returned.

**WITHOUT DATALINK**

Specifies that any tables with DATALINK columns are to be put in DataLink_Reconcile_Pending (DRP) state, and that no reconciliation of linked files is to be performed.

**PARALLELISM** *n*

Specifies the number of buffer manipulators that are to be spawned during the restore operation. DB2 will automatically choose an optimal value for this parameter unless you explicitly enter a value.

**COMPRLIB** *name*

Indicates the name of the library to be used to perform the decompression. The name must be a fully qualified path referring to a file on the server. If this parameter is not specified, DB2 will attempt to use the library stored in the image. If the backup was not compressed, the value of this parameter will be ignored. If the specified library cannot be loaded, the restore will fail.

**COMPROPTS** *string*

Describes a block of binary data that will be passed to the initialization routine in the decompression library. DB2 will pass this string directly from the client to the server, so any issues of byte reversal or code page conversion will have to be handled by the decompression library. If the first character of the data block is '@', the remainder of the data will be interpreted by DB2 as the name of a file residing on the server. DB2 will then replace the contents of *string* with the contents of this file and will pass this new value to the initialization routine instead. The maximum length for string is 1024 bytes.

**WITHOUT PROMPTING**

Specifies that the restore operation is to run unattended. Actions that normally require user intervention will return an error message. When using a removable media device, such as tape or diskette, the user is prompted when the device ends, even if this option is specified.

**Examples:**

1. In the following example, the database WSDB is defined on all 4 partitions, numbered 0 through 3. The path /dev3/backup is accessible from all partitions. The following offline backup images are available from /dev3/backup:

   ```
   wsdb.0.db2inst1.NODE0000.CATN0000.20020331234149.001
   wsdb.0.db2inst1.NODE0001.CATN0000.20020331234427.001
   wsdb.0.db2inst1.NODE0002.CATN0000.20020331234828.001
   wsdb.0.db2inst1.NODE0003.CATN0000.20020331235235.001
   ```

   To restore the catalog partition first, then all other database partitions of the WSDB database from the /dev3/backup directory, issue the following commands from one of the database partitions:

   ```
   db2_all '<<+0< db2 RESTORE DATABASE wsdb FROM /dev3/backup
   TAKEN AT 20020331234149
     INTO wsdb REPLACE EXISTING'
   db2_all '<<+1< db2 RESTORE DATABASE wsdb FROM /dev3/backup
   TAKEN AT 20020331234427
     INTO wsdb REPLACE EXISTING'
   db2_all '<<+2< db2 RESTORE DATABASE wsdb FROM /dev3/backup
   TAKEN AT 20020331234828
     INTO wsdb REPLACE EXISTING'
   db2_all '<<+3< db2 RESTORE DATABASE wsdb FROM /dev3/backup
   TAKEN AT 20020331235235
     INTO wsdb REPLACE EXISTING'
   ```

   The db2_all utility issues the restore command to each specified database partition.

2. Following is a typical redirected restore scenario for a database whose alias is MYDB:

   a. Issue a RESTORE DATABASE command with the REDIRECT option.

      ```
      db2 restore db mydb replace existing redirect
      ```

      After successful completion of step 1, and before completing step 3, the restore operation can be aborted by issuing:

      ```
      db2 restore db mydb abort
      ```

   b. Issue a SET TABLESPACE CONTAINERS command for each table space whose containers must be redefined. For example:

      ```
      db2 set tablespace containers for 5 using
         (file 'f:\ts3con1' 20000, file 'f:\ts3con2' 20000)
      ```

      To verify that the containers of the restored database are the ones specified in this step, issue the LIST TABLESPACE CONTAINERS command.

   c. After successful completion of steps 1 and 2, issue:

```
db2 restore db mydb continue
```

This is the final step of the redirected restore operation.

d. If step 3 fails, or if the restore operation has been aborted, the redirected restore can be restarted, beginning at step 1.

3. Following is a sample weekly incremental backup strategy for a recoverable database. It includes a weekly full database backup operation, a daily non-cumulative (delta) backup operation, and a mid-week cumulative (incremental) backup operation:

```
(Sun) backup db mydb use tsm
(Mon) backup db mydb online incremental delta use tsm
(Tue) backup db mydb online incremental delta use tsm
(Wed) backup db mydb online incremental use tsm
(Thu) backup db mydb online incremental delta use tsm
(Fri) backup db mydb online incremental delta use tsm
(Sat) backup db mydb online incremental use tsm
```

For an automatic database restore of the images created on Friday morning, issue:

```
restore db mydb incremental automatic taken at (Fri)
```

For a manual database restore of the images created on Friday morning, issue:

```
 restore db mydb incremental taken at (Fri)
 restore db mydb incremental taken at (Sun)
 restore db mydb incremental taken at (Wed)
 restore db mydb incremental taken at (Thu)
 restore db mydb incremental taken at (Fri)
```

4. To produce a backup image, which includes logs, for transportation to a remote site:

```
backup db sample online to /dev3/backup include logs
```

To restore that backup image, supply a LOGTARGET path and specify this path during ROLLFORWARD:

```
restore db sample from /dev3/backup logtarget /dev3/logs
rollforward db sample to end of logs and stop overflow log path /dev3/logs
```

5. To retreive only the log files from a backup image that includes logs:

```
restore db sample logs from /dev3/backup logtarget /dev3/logs
```

6. The USE TSM OPTIONS keywords can be used to specify the TSM information to use for the restore operation. On Windows platforms, omit the -fromowner option.

   - Specfying a delimited string:

   ```
   db2 restore db sample use TSM options "-fromnode bar -fromowner dmcinnis"
   ```

   - Specifying a fully qualified file:

   ```
   db2 restore db sample use TSM options @/u/dmcinnis/myoptions.txt
   ```

   The file `myoptions.txt` contains the following information: -fromnode=bar -fromowner=dmcinnis

**Usage notes:**

- A RESTORE DATABASE command of the form `db2 restore db <name>` will perform a full database restore with a database image and will perform a table space restore of the table spaces found in a table space image. Any RESTORE DATABASE command of the form `db2 restore db <name> tablespace` will perform a table space restore of the table spaces found in the image. Any

RESTORE DATABASE command in which a list of table spaces is provided will perform a restore of whatever table spaces are explicitly listed.

- Following the restore of an online backup, you must perform a roll forward recovery.
- If a backup image is compressed, DB2 will detect this and automatically decompress the data before restoring it. If a library is specified on the db2Restore API, it will be used for decompressing the data. Otherwise, if a library that is stored in the backup image will be used. Otherwise, the data cannot be decompressed, so the restore will fail.
- If the compression library is to be restored from a backup image (either explicitly by specifying the DB2RESTORE_COMPR_LIB restore type or implicitly by performing a normal restore of a compressed backup), the restore operation must be done on the same platform and operating system that the backup was taken on. If the platform the backup was taken on is not the same as the platform that the restore is being done on, the restore operation will fail, even if DB2 normally supports cross-platform restores involving the two systems.
- 
- To restore log files from a backup image which contains them, the LOGTARGET option must be specified, providing a fully qualified and valid path which exists on the DB2 server. If those conditions are satisfied, the restore utility will write the log files from the image to the target path. If a LOGTARGET is specified during a restore of a backup image which does not include logs, the restore will return an error before attempting to restore any table space data. A restore will also fail with an error if an invalid, or read-only, LOGTARGET path is specified.
- If any log files exist in the LOGTARGET path at the time the RESTORE DATABASE command is issued, a warning prompt will be returned to user. This warning will not be returned if WITHOUT PROMPTING is specified.
- During a restore operation where a LOGTARGET is specified, if any log file can not be extracted, for any reason, the restore will fail and return an error. If any of the log files being extracted from the backup image have the same name as an existing file already in the LOGTARGET path, the restore operation will fail and an error will be returned. The restore database utility will not overwrite existing log files in the LOGTARGET directory.
- It is also possible to restore only the saved log set from a backup image. To indicate that only the log files are to be restored, specify the LOGS option in addition to the LOGTARGET path. Specifying the LOGS option without a LOGTARGET path will result in an error. If any problem occurs while restoring log files in this mode of operation, the restore operation will terminate immediately and an error will be returned.
- During an automatic incremental restore operation, only the logs included in the target image of the restore operation will be retrived from the backup image. Any logs included in intermediate images referenced during the incremental restore process will not be extracted from those intermediate backup images. During a manual incremental restore operation, the LOGTARGET path should only be specified with the final restore command to be issued.

**Related reference:**

- “BACKUP DATABASE” on page 280
- “ROLLFORWARD DATABASE” on page 657
- “db2move - Database Movement Tool” on page 134

# REWIND TAPE

When running on Windows NT-based operating systems, DB2 supports backup and restore operations to streaming tape devices. Use this command for tape rewinding.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──REWIND TAPE──┬─────────────┬──────────────────────────────────────────►◄
                 └─ON──device──┘
```

**Command parameters:**

**ON device**
Specifies a valid tape device name. The default value is `\\.\TAPE0`.

**Related reference:**

- "INITIALIZE TAPE" on page 472
- "SET TAPE POSITION" on page 685

# ROLLFORWARD DATABASE

Recovers a database by applying transactions recorded in the database log files. Invoked after a database or a table space backup image has been restored, or if any table spaces have been taken offline by the database due to a media error. The database must be recoverable (that is, the *logarchmeth1* or *logarchmeth2* database configuration parameters must be set to a value other than OFF) before the database can be recovered with rollforward recovery.

**Scope:**

In a partitioned database environment, this command can only be invoked from the catalog partition. A database or table space rollforward operation to a specified point in time affects all partitions that are listed in the db2nodes.cfg file. A database or table space rollforward operation to the end of logs affects the partitions that are specified. If no partitions are specified, it affects all partitions that are listed in the db2nodes.cfg file; if rollforward recovery is not needed on a particular partition, that partition is ignored.

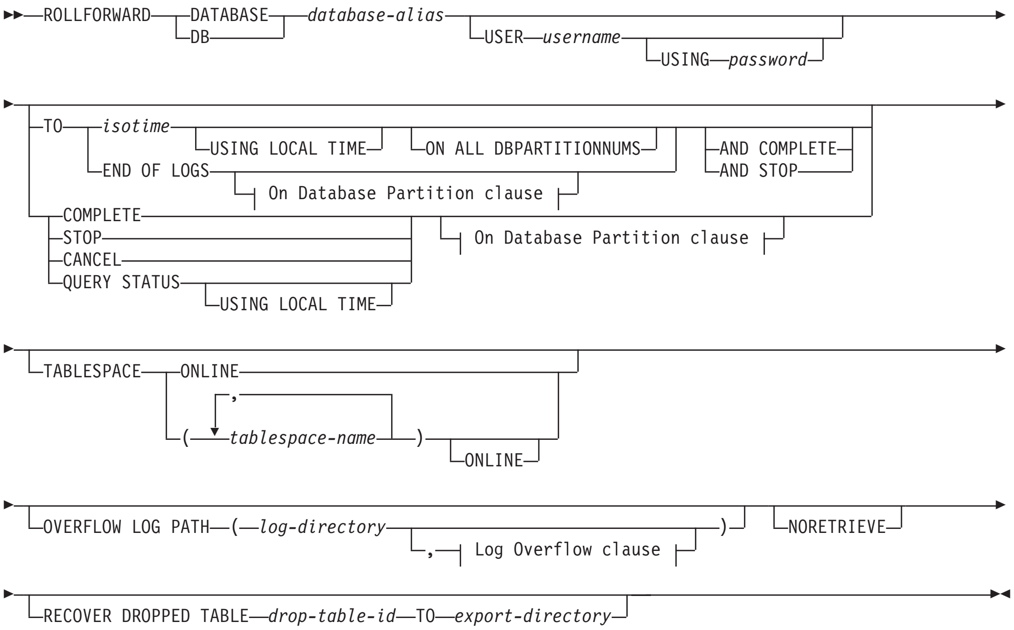
**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

None. This command establishes a database connection.

**Command syntax:**



### On Database Partition clause:

```
├──ON──┬─┤ Database Partition List clause ├──────────────────────┬─────────┤
       └─ALL DBPARTITIONNUMS──┬──────────────────────────────────┤
                              └─EXCEPT─┤ Database Partition List clause ├─┘
```

### Database Partition List clause:

```
├──┬─DBPARTITIONNUM──┬──(───────────────────────────────────────────────►
   └─DBPARTITIONNUMS─┘


   ┌─,──────────────────────────────────────────────┐
   ▼                                                 │
►────db-partition-number1──┬─────────────────────────┬┴──)──────────────┤
                           └─TO──db-partition-number2─┘
```

### Log Overflow clause:

```
   ┌─,────────────────────────────────────────────────────┐
   ▼                                                       │
├────log-directory──ON DBPARTITIONNUM──db-partition-number1─┴────────────┤
```

**Command parameters:**

**DATABASE database-alias**
The alias of the database that is to be rollforward recovered.

**USER username**
The user name under which the database is to be rollforward recovered.

**USING password**
The password used to authenticate the user name. If the password is omitted, the user is prompted to enter it.

**TO**

**isotime**
The point in time to which all committed transactions are to be rolled forward (including the transaction committed precisely at that time, as well as all transactions committed previously).

This value is specified as a time stamp, a 7-part character string that identifies a combined date and time. The format is *yyyy-mm-dd-hh.mm.ss.nnnnnn* (year, month, day, hour, minutes, seconds, microseconds), expressed in Coordinated Universal Time (UTC). UTC helps to avoid having the same time stamp associated with different logs (because of a change in time associated with daylight savings time, for example). The time stamp in a backup image is based on the local time at which the backup operation started. The CURRENT TIMEZONE special register specifies the difference between UTC and local time at the application server. The difference is represented by a time duration (a decimal number in which the first two digits represent the number of hours, the next two digits represent the number of minutes, and the last two digits represent the number of seconds). Subtracting CURRENT TIMEZONE from a local time converts that local time to UTC.

**USING LOCAL TIME**
Allows the user to rollforward to a point in time that is the user's local time rather than GMT time. This makes it easier for users to rollforward to a specific point in time on their local machines, and eliminates potential user errors due to the translation of local to GMT time.

**Notes:**

1. If the user specifies a local time for rollforward, all messages returned to the user will also be in local time. Note that all times are converted on the server, and in partitioned database environments, on the catalog database partition.
2. The timestamp string is converted to GMT on the server, so the time is local to the server's time zone, not the client's. If the client is in one time zone and the server in another, the server's local time should be used. This is different from the local time option from the Control Center, which is local to the client.
3. If the timestamp string is close to the time change of the clock due to daylight savings, it is important to know if the stop time is before or after the clock change, and specify it correctly.

**END OF LOGS**
Specifies that all committed transactions from all online archive log files listed in the database configuration parameter *logpath* are to be applied.

**ALL DBPARTITIONNUMS**
Specifies that transactions are to be rolled forward on all partitions specified in the `db2nodes.cfg` file. This is the default if a database partition clause is not specified.

**EXCEPT**
Specifies that transactions are to be rolled forward on all partitions specified in the `db2nodes.cfg` file, except those specified in the database partition list.

**ON DBPARTITIONNUM / ON DBPARTITIONNUMS**
Roll the database forward on a set of database partitions.

**db-partition-number1**
Specifies a database partition number in the database partition list.

**db-partition-number2**
Specifies the second database partition number, so that all partitions from *db-partition-number1* up to and including *db-partition-number2* are included in the database partition list.

**COMPLETE / STOP**
Stops the rolling forward of log records, and completes the rollforward recovery process by rolling back any incomplete transactions and turning off the rollforward pending state of the database. This allows access to the database or table spaces that are being rolled forward. These keywords are equivalent; specify one or the other, but not both. The keyword AND permits specification of multiple operations at once; for example, `db2 rollforward db sample to end of logs and complete`.

**Note:** When rolling table spaces forward to a point in time, the table spaces are placed in backup pending state.

**CANCEL**

Cancels the rollforward recovery operation. This puts the database or one or more table spaces on all partitions on which forward recovery has been started in restore pending state:

- If a *database* rollforward operation is not in progress (that is, the database is in rollforward pending state), this option puts the database in restore pending state.
- If a *table space* rollforward operation is not in progress (that is, the table spaces are in rollforward pending state), a table space list must be specified. All table spaces in the list are put in restore pending state.
- If a table space rollforward operation *is* in progress (that is, at least one table space is in rollforward in progress state), all table spaces that are in rollforward in progress state are put in restore pending state. If a table space list is specified, it must include all table spaces that are in rollforward in progress state. All table spaces on the list are put in restore pending state.
- If rolling forward to a point in time, any table space name that is passed in is ignored, and all table spaces that are in rollforward in progress state are put in restore pending state.
- If rolling forward to the end of the logs with a table space list, only the table spaces listed are put in restore pending state.

This option cannot be used to cancel a rollforward operation *that is actually running*. It can only be used to cancel a rollforward operation that is in progress but not actually running at the time. A rollforward operation can be in progress but not running if:

- It terminated abnormally.
- The STOP option was not specified.
- An error caused it to fail. Some errors, such as rolling forward through a non-recoverable load operation, can put a table space into restore pending state.

**Note:** Use this option with caution, and only if the rollforward operation that is in progress cannot be completed because some of the table spaces have been put in rollforward pending state or in restore pending state. When in doubt, use the LIST TABLESPACES command to identify the table spaces that are in rollforward in progress state, or in rollforward pending state.

**QUERY STATUS**

Lists the log files that the database manager has rolled forward, the next archive file required, and the time stamp (in CUT) of the last committed transaction since rollforward processing began. In a partitioned database environment, this status information is returned for each partition. The information returned contains the following fields:

**Database partition number**

**Rollforward status**

Status can be: database or table space rollforward pending, database or table space rollforward in progress, database or table space rollforward processing STOP, or not pending.

**Next log file to be read**

A string containing the name of the next required log file. In a partitioned database environment, use this information if the rollforward utility fails with a return code indicating that a log file is missing or that a log information mismatch has occurred.

**Log files processed**

A string containing the names of processed log files that are no longer needed for recovery, and that can be removed from the directory. If, for example, the oldest uncommitted transaction starts in log file *x*, the range of obsolete log files will not include *x*; the range ends at *x* - 1.

**Last committed transaction**

A string containing a time stamp in ISO format (*yyyy-mm-dd-hh.mm.ss*). This time stamp marks the last transaction committed after the completion of rollforward recovery. The time stamp applies to the database. For table space rollforward recovery, it is the time stamp of the last transaction committed to the database.

**Note:** QUERY STATUS is the default value if the TO, STOP, COMPLETE, or CANCEL clauses are omitted. If TO, STOP, or COMPLETE was specified, status information is displayed if the command has completed successfully. If individual table spaces are specified, they are ignored; the status request does not apply only to specified table spaces.

**TABLESPACE**

This keyword is specified for table space-level rollforward recovery.

**tablespace-name**

Mandatory for table space-level rollforward recovery to a point in time. Allows a subset of table spaces to be specified for rollforward recovery to the end of the logs. In a partitioned database environment, each table space in the list does not have to exist at each partition that is rolling forward. If it *does* exist, it must be in the correct state.

**ONLINE**

This keyword is specified to allow table space-level rollforward recovery to be done online. This means that other agents are allowed to connect while rollforward recovery is in progress.

**OVERFLOW LOG PATH log-directory**

Specifies an alternate log path to be searched for archived logs during recovery. Use this parameter if log files were moved to a location other than that specified by the *logpath* database configuration parameter. In a partitioned database environment, this is the (fully qualified) default overflow log path *for all partitions*. A relative overflow log path can be specified for single-partition databases.

**Note:** The OVERFLOW LOG PATH command parameter will overwrite the value (if any) of the database configuration parameter OVERFLOWLOGPATH.

**log-directory ON DBPARTITIONNUM**

In a partitioned database environment, allows a different log path to override the default overflow log path for a specific partition.

**NORETRIEVE**

Allows the user to control which log files to be rolled forward on the standby machine by allowing the user to disable the retrieval of archived logs. The benefits of this are:

- By controlling the logfiles to be rolled forward, one can ensure that the standby machine is X hours behind the production machine, to prevent the user affecting both systems.
- If the standby system does not have access to archive (eg. if TSM is the archive, it only allows the original machine to retrieve the files)
- It might also be possible that while the production system is archiving a file, the standby system is retrieving the same file, and it might then get an incomplete log file. Noretrieve would solve this problem.

**RECOVER DROPPED TABLE drop-table-id**
Recovers a dropped table during the rollforward operation. The table ID can be obtained using the LIST HISTORY command.

**TO export-directory**
Specifies a directory to which files containing the table data are to be written. The directory must be accessible to all database partitions.

**Examples:**

**Example 1**

The ROLLFORWARD DATABASE command permits specification of multiple operations at once, each being separated with the keyword AND. For example, to roll forward to the end of logs, and complete, the separate commands:

```
db2 rollforward db sample to end of logs
db2 rollforward db sample complete
```

can be combined as follows:

```
db2 rollforward db sample to end of logs and complete
```

Although the two are equivalent, it is recommended that such operations be done in two steps. It is important to verify that the rollforward operation has progressed as expected, before stopping it and possibly missing logs. This is especially important if a bad log is found during rollforward recovery, and the bad log is interpreted to mean the “end of logs”. In such cases, an undamaged backup copy of that log could be used to continue the rollforward operation through more logs.

**Example 2**

Roll forward to the end of the logs (two table spaces have been restored):

```
db2 rollforward db sample to end of logs
db2 rollforward db sample to end of logs and stop
```

These two statements are equivalent. Neither AND STOP or AND COMPLETE is needed for table space rollforward recovery to the end of the logs. Table space names are not required. If not specified, all table spaces requiring rollforward recovery will be included. If only a subset of these table spaces is to be rolled forward, their names must be specified.

**Example 3**

After three table spaces have been restored, roll one forward to the end of the logs, and the other two to a point in time, both to be done online:

```
db2 rollforward db sample to end of logs tablespace(TBS1) online

db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop
   tablespace(TBS2, TBS3) online
```

Note that two rollforward operations cannot be run concurrently. The second command can only be invoked after the first rollforward operation completes successfully.

**Example 4**

After restoring the database, roll forward to a point in time, using OVERFLOW LOG PATH to specify the directory where the user exit saves archived logs:

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop
   overflow log path (/logs)
```

**Example 5 (partitioned database environments)**

There are three database partitions: 0, 1, and 2. Table space TBS1 is defined on all partitions, and table space TBS2 is defined on partitions 0 and 2. After restoring the database on database partition 1, and TBS1 on database partitions 0 and 2, roll the database forward on database partition 1:

```
db2 rollforward db sample to end of logs and stop
```

This returns warning SQL1271 ("Database is recovered but one or more table spaces are off-line on database partition(s) 0 and 2.").

```
db2 rollforward db sample to end of logs
```

This rolls TBS1 forward on database partitions 0 and 2. The clause TABLESPACE(TBS1) is optional in this case.

**Example 6 (partitioned database environments)**

After restoring table space TBS1 on database partitions 0 and 2 only, roll TBS1 forward on database partitions 0 and 2:

```
db2 rollforward db sample to end of logs
```

Database partition 1 is ignored.

```
db2 rollforward db sample to end of logs tablespace(TBS1)
```

This fails, because TBS1 is not ready for rollforward recovery on database partition 1. Reports SQL4906N.

```
db2 rollforward db sample to end of logs on dbpartitionnums (0, 2)
   tablespace(TBS1)
```

This completes successfully.

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop
   tablespace(TBS1)
```

This fails, because TBS1 is not ready for rollforward recovery on database partition 1; all pieces must be rolled forward together.

**Note:** With table space rollforward to a point in time, the database partition clause is not accepted. The rollforward operation must take place on all the database partitions on which the table space resides.

After restoring TBS1 on database partition 1:

```
db2 rollforward db sample to 1998-04-03-14.21.56.245378 and stop
   tablespace(TBS1)
```

This completes successfully.

**Example 7 (partitioned database environment)**

After restoring a table space on all database partitions, roll forward to PIT2, but do not specify AND STOP. The rollforward operation is still in progress. Cancel and roll forward to PIT1:

```
   db2 rollforward db sample to pit2 tablespace(TBS1)
   db2 rollforward db sample cancel tablespace(TBS1)

 ** restore TBS1 on all database partitions **

   db2 rollforward db sample to pit1 tablespace(TBS1)
   db2 rollforward db sample stop tablespace(TBS1)
```

**Example 8 (partitioned database environments)**

Rollforward recover a table space that resides on eight database partitions (3 to 10) listed in the db2nodes.cfg file:

```
   db2 rollforward database dwtest to end of logs tablespace (tssprodt)
```

This operation to the end of logs (not point in time) completes successfully. The database partitions on which the table space resides do not have to be specified. The utility defaults to the db2nodes.cfg file.

**Example 9 (partitioned database environment)**

Rollforward recover six small table spaces that reside on a single-partition database partition group (on database partition 6):

```
   db2 rollforward database dwtest to end of logs on dbpartitionnum (6)
      tablespace(tsstore, tssbuyer, tsstime, tsswhse, tsslscat, tssvendor)
```

This operation to the end of logs (not point in time) completes successfully.

**Usage notes:**

If restoring from an image that was created during an online backup operation, the specified point in time for the rollforward operation must be later than the time at which the online backup operation completed. If the rollforward operation is stopped before it passes this point, the database is left in rollforward pending state. If a table space is in the process of being rolled forward, it is left in rollforward in progress state.

If one or more table spaces is being rolled forward to a point in time, the rollforward operation must continue at least to the minimum recovery time, which is the last update to the system catalogs for this table space or its tables. The minimum recovery time (in Coordinated Universal Time, or UTC) for a table space can be retrieved using the LIST TABLESPACES SHOW DETAIL command.

Rolling databases forward might require a load recovery using tape devices. If prompted for another tape, the user can respond with one of the following:

**c** Continue. Continue using the device that generated the warning message (for example, when a new tape has been mounted)

**d** Device terminate. Stop using the device that generated the warning message (for example, when there are no more tapes)

**t** Terminate. Terminate all devices.

If the rollforward utility cannot find the next log that it needs, the log name is returned in the SQLCA, and rollforward recovery stops. If no more logs are available, use the STOP option to terminate rollforward recovery. Incomplete transactions are rolled back to ensure that the database or table space is left in a consistent state.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.
- The keyword NODES can be substituted for DBPARTITIONNUMS.

**Related reference:**

- “BACKUP DATABASE” on page 280
- “RESTORE DATABASE” on page 647

## RUNCMD

Executes a specified command from the CLP interactive mode command history.

**Scope**

This command can only be run within CLP interactive mode. Specifically, it cannot be run from the CLP command mode or the CLP batch mode.

**Authorization:**

None

**Required connection:**

The required connection will depend on the command being executed.

**Command syntax:**

```
►►──┬─RUNCMD─┬──┬─────┬──────────────────────────────────────────────►◄
    └─R──────┘  └─num─┘
```

**Command parameters:**

**num**
If *num* is positive, executes the command corresponding to *num* in the command history. If *num* is negative, executes the command corresponding to *num*, counting backwards from the most recent command in the command history. Zero is not a valid value for *num*. If this parameter is not specified, executes the most recently run command. (This is equivalent to specifying a value of -1 for *num*).

**Usage notes:**

1. Typically, you would execute the HISTORY command to see a list of recently executed commands and then execute the RUNCMD to execute a command from this list.
2. The RUNCMD command is not recorded in the command history, but the command executed by the RUNCMD command is recorded in the command history.

**Related reference:**

- "EDIT" on page 361
- "HISTORY" on page 448

# RUNSTATS

Updates statistics about the physical characteristics of a table and the associated indexes. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

This utility should be called when a table has had many updates, or after reorganizing a table.

**Scope:**

This command can be issued from any database partition in the `db2nodes.cfg` file. It can be used to update the catalogs on the catalog database partition.

The command collects statistics for a table on the database partition from which it is invoked. If the table does not exist on that database partition, the first database partition in the database partition group is selected.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*
- CONTROL privilege on the table
- LOAD authority

A user does not need any explicit privilege to use this command on any declared global temporary table that exists within its connection.

**Required connection:**

Database

**Command syntax:**

```
>>--RUNSTATS--ON TABLE--table name--+-USE PROFILE--------------+-------------------->
                                    '-| Statistics Options |-'

>--+-------------------------------------+---------------------------------------><
   '-UTIL_IMPACT_PRIORITY--+----------+--'
                           '-priority-'
```

### Statistics Options:

```
                                      .-ALLOW WRITE ACCESS-.
|--+---------------------------+------+--------------------+------------------------>
   '-| Table Object Options |--'       '-ALLOW READ ACCESS--'
```

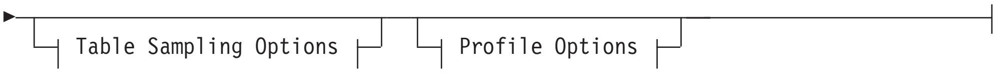


**Table Object Options:**

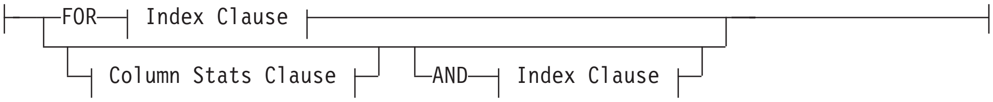
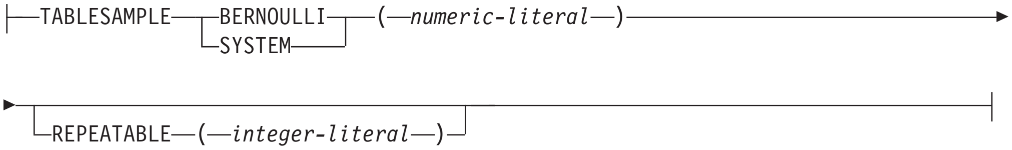
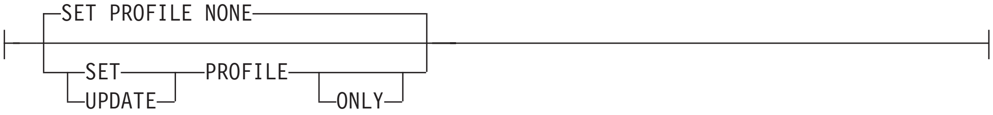


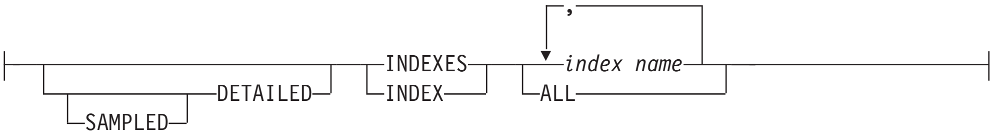
**Table Sampling Options:**



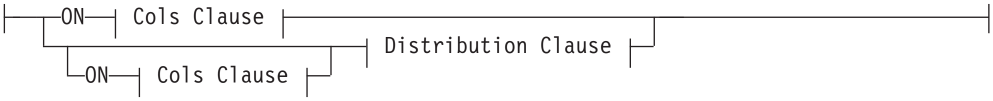
**Profile Options:**



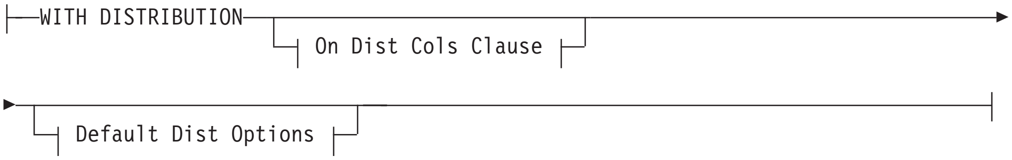
**Index Clause:**



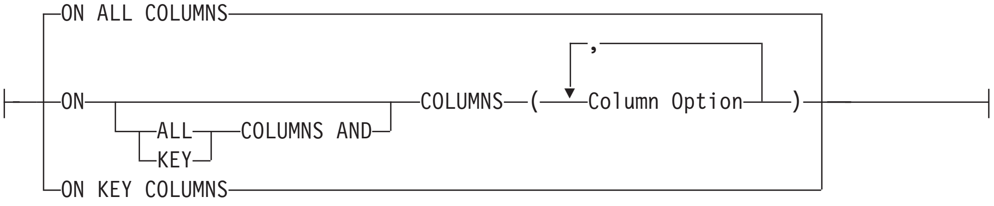
**Column Stats Clause:**



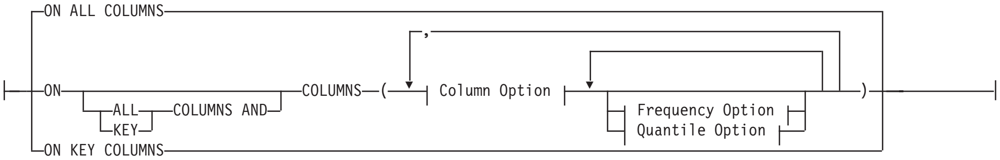
**Distribution Clause:**



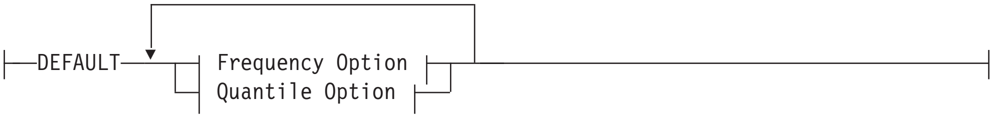
**On Cols Clause:**



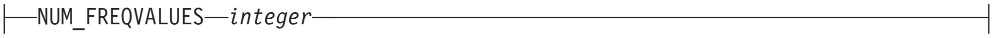
### On Dist Cols Clause:



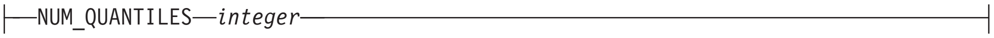
### Default Dist Option:



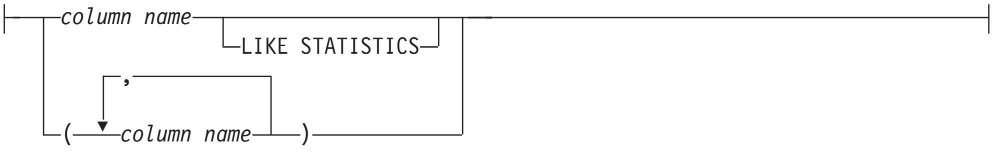
### Frequency Option:



### Quantile Option:



### Column Option:



**Command parameters:**

**table-name**

Identifies the table on which statistics are to be collected. It must be a table described in the catalog and must not be a view or a Hierarchy table. For typed tables, table-name must be the name of the root table of the table hierarchy. The fully qualified name or alias in the form: schema.table-name must be used. The schema is the user name under which the table was created.

**index-name**

Identifies an existing index defined on the table. The fully qualified name in the form *schema.index-name* must be used.

**USE PROFILE**

This option allows RUNSTATS to employ a previously stored statistics profile to gather statistics for a table. The statistics profile is created using the SET PROFILE options and is updated using the UPDATE PROFILE options.

**FOR INDEXES**

Collects and updates statistics for the indexes only. If no table statistics had been previously collected on the table, basic table statistics are also collected. These basic statistics do not include any distribution statistics.

**AND INDEXES**

Collects and updates statistics for both the table and the indexes.

**DETAILED**

Calculates extended index statistics. These are the CLUSTERFACTOR and PAGE_FETCH_PAIRS statistics that are gathered for relatively large indexes.

**SAMPLED**

This option, when used with the DETAILED option, allows RUNSTATS to employ a CPU sampling technique when compiling the extended index statistics. If the option is not specified, every entry in the index is examined to compute the extended index statistics.

**ON ALL COLUMNS**

Statistics collection can be done on some columns and not on others. Columns such as LONG VARCHAR or CLOB columns are ineligible. If it is desired to collect statistics on all eligible columns, one can use the ON ALL COLUMNS clause. Columns can be specified either for basic statistics collection (on-cols-clause) or in conjunction with the WITH DISTRIBUTION clause (on-dist-cols-clause). The ON ALL COLUMNS specification is the default option if neither of the column specific clauses are specified.

If it is specified in the on-cols-clause, all columns will have only basic column statistics collected unless specific columns are chosen as part of the WITH DISTRIBUTION clause. Those columns specified as part of the WITH DISTRIBUTION clause will also have basic and distribution statistics collected.

If the WITH DISTRIBUTION ON ALL COLUMNS is specified both basic statistics and distribution statistics are collected for all eligible columns. Anything specified in the on-cols-clause is redundant and therefore not necessary.

**ON COLUMNS**

This clause allows the user to specify a list of columns for which to collect statistics. If you specify group of columns, the number of distinct values for the group will be collected. Statistics for columns that you do not list will be cleared. This clause can be used in the on-cols-clause and the on-dist-cols-clause.

**Note:** Collecting distribution statistics for a group of columns is not currently supported.

**ON KEY COLUMNS**

Instead of listing specific columns, one can choose to collect statistics on columns that make up all the indexes defined on the table. It is assumed here that critical columns in queries are also those used to create indexes on the table. If there are no indexes on the table, it is as good as an empty list and no column statistics will be collected. It can be used in the on-cols-clause or the on-dist-cols-clause. It is redundant in the on-cols-clause if specified in both clauses since the WITH DISTRIBUTION clause is used to specify collection of both basic and distribution statistics.

**column-name**
: column-name must be the name of a column in the table. If you specify the name of an ineligible column for statistics collection, such as a non-existent column or a mistyped column name, error (-205) is returned. Two lists of columns can be specified, one without distribution and one with distribution. If the column is specified in the list that is not associated with the WITH DISTRIBUTION clause only basic column statistics will be collected. If the column appears in both lists, distribution statistics will be collected (unless NUM_FREQVALUES and NUM_QUANTILES are set to zero).

**NUM_FREQVALUES**
: Defines the maximum number of frequency values to collect. It can be specified for an individual column in the ON COLUMNS clause. If the value is not specified for an individual column, the frequency limit value will be picked up from that specified in the DEFAULT clause. If it is not specified there either, the maximum number of frequency values to be collected will be what is set in the NUM_FREQVALUES database configuration parameter.

**NUM_QUANTILES**
: Defines the maximum number of distribution quantile values to collect. It can be specified for an individual column in the ON COLUMNS clause. If the value is not specified for an individual column, the quantile limit value will be picked up from that specified in the DEFAULT clause. If it is not specified there either, the maximum number of quantile values to be collected will be what is set in the NUM_QUANTILES database configuration parameter.

**WITH DISTRIBUTION**
: This clause specifies that both basic statistics and distribution statistics are to be collected on the columns. If the ON COLUMNS clause is not specified, distribution statistics are collected on all the columns of the table (excluding columns that are ineligible such as CLOB and LONG VARCHAR). If the ON COLUMNS clause is specified, distribution statistics are collected only on the column list provided (excluding those ineligible for statistics collection). If the clause is not specified, only basic statistics are collected.

**Note:** Collection of distribution statistics on column groups is currently not supported; distribution statistics will not be collected when column groups are specified in the WITH DISTRIBUTION ON COLUMNS clause.

**DEFAULT**
: If NUM_FREQVALUES or NUM_QUANTILES are specified, these values will be used to determine the maximum number of frequency and quantile statistics to be collected for the columns, if these are not specified for individual columns in the ON COLUMNS clause. If the DEFAULT clause is not specified, the values used will be those in the corresponding database configuration parameters.

**LIKE STATISTICS**
When this option is specified additional column statistics are collected. These statistics are the SUB_COUNT and the SUB_DELIM_LENGTH statistics in SYSSTAT.COLUMNS. They are collected for string columns only and they are used by the query optimizer to improve the selectivity estimates for predicates of the type ″column LIKE ′%xyz′″ and ″column LIKE ′%xyz%′″

**ALLOW WRITE ACCESS**
Specifies that other users can read from and write to the table while statistics are calculated.

**ALLOW READ ACCESS**
Specifies that other users can have read-only access to the table while statistics are calculated.

**TABLESAMPLE BERNOULLI**
This option allows RUNSTATS to collect statistics on a sample of the rows from the table. BERNOULLI sampling considers each row individually, including that row with probability P/100 (where P is the value of numeric-literal) and excluding it with probability 1-P/100. Thus, if the numeric-literal were evaluated to be the value 10, representing a 10 percent sample, each row would be included with probability 0.1 and be excluded with probability 0.9. Unless the optional REPEATABLE clause is specified, each execution of RUNSTATS will usually yield a different such sample of the table. All data pages will be retrieved through a table scan but only the percentage of rows as specified through the numeric-literal parameter will be used for the statistics collection.

**TABLESAMPLE SYSTEM**
This option allows RUNSTATS to collect statistics on a sample of the data pages from the table. SYSTEM sampling considers each page individually, including that page with probability P/100 (where P is the value of numeric-literal) and excluding it with probability 1-P/100. Unless the optional REPEATABLE clause is specified, each execution of RUNSTATS will usually yield a different such sample of the table. The size of the sample is controlled by the numeric-literal parameter in parentheses, representing an approximate percentage P of the table to be returned. Only a percentage of the data pages as specified through the numeric-literal parameter will be retrieved and used for the statistics collection.

**REPEATABLE (*integer-literal*)**
Adding the REPEATABLE clause to the TABLESAMPLE clause ensures that repeated executions of RUNSTATS return the same sample. The *integer-literal* parameter is a non-negative integer representing the seed to be used in table sampling. Passing a negative seed will result in an error (SQL1197N). Note that the sample set might still vary between repeatable RUNSTATS invocations if activity against the table resulted in changes to the table data since the last time TABLESAMPLE REPEATABLE was run. Also, the method by which the sample was obtained as specified by the BERNOULLI or SYSTEM keyword, must also be the same to ensure consistent results.

*numeric-literal*
The numeric-literal parameter specifies the size of the sample to be obtained, as a percentage P. This value must be a positive number that is less than or equal to 100, and can be between 1 and 0. For example, a value of 0.01 represents one one-hundredth of a percent, such that 1 row in 10,000 would be sampled, on average. A value of 0 or 100 will be treated

by DB2 as if table sampling was not specified, regardless of whether TABLESAMPLE BERNOULLI or TABLESAMPLE SYSTEM is specified. A value greater than 100 or less than 0 will be treated by DB2 as an error (SQL1197N).

**SET PROFILE NONE**

Specifies that no statistics profile will be set for this RUNSTATS invocation.

**SET PROFILE**

Allows RUNSTATS to generate and store a specific statistics profile in the system catalog tables and executes the RUNSTATS command options to gather statistics.

**SET PROFILE ONLY**

Allows RUNSTATS to generate and store a specific statistics profile in the system catalog tables without running the RUNSTATS command options.

**UPDATE PROFILE**

Allows RUNSTATS to modify an existing statistics profile in the system catalog tables, and runs the RUNSTATS command options of the updated statistics profile to gather statistics.

**UPDATE PROFILE ONLY**

Allows RUNSTATS to modify an existing statistics profile in the system catalog tables without running the RUNSTATS command options of the updated statistics profile.

**UTIL_IMPACT_PRIORITY** *priority*

Specifies that RUNSTATS will be throttled at the level specified by *priority*. *priority* is a number in the range of 1 to 100, with 100 representing the highest priority and 1 representing the lowest. The priority specifies the amount of throttling to which the utility is subjected. All utilities at the same priority undergo the same amount of throttling, and utilities at lower priorities are throttled more than those at higher priorities. If *priority* is not specified, the RUNSTATS will have the default priority of 50. Omitting the UTIL_IMPACT_PRIORITY keyword will invoke the RUNSTATS utility without throttling support. If the UTIL_IMPACT_PRIORITY keyword is specified, but the *util_impact_lim* configuration parameter is set to 100, then the utility will run unthrottled.

**Note:** In a partitioned database, the RUNSTATS command collects the statistics on only a single database partition. If the database partition from which the RUNSTATS command is executed has a partition of the table, then the command executes on that database partition. Otherwise, the command executes on the first database partition in the database partition group across which the table is partitioned.

**Usage Notes:**

1. It is recommended to run the RUNSTATS command:
   - On tables that have been modified considerably (for example, if a large number of updates have been made, or if a significant amount of data has been inserted or deleted or if LOAD has been done without the statistics option during LOAD)
   - On tables that have been reorganized (using REORG, REDISTRIBUTE DATABASE PARTITION GROUP)
   - When a new index has been created.
   - Before binding applications whose performance is critical.

- When the prefetch quantity is changed

2. The options chosen must depend on the specific table and the application. In general:
   - If the table is a very critical table in critical queries, is relatively small, or does not change too much and there is not too much activity on the system itself, it might be worth spending the effort on collecting statistics in as much detail as possible.
   - If the time to collect statistics is limited, if the table is relatively large, or if the table is updated frequently, it might be beneficial to execute RUNSTATS limited to the set of columns that are used in predicates. This way, you will be able to execute the RUNSTATS command more often.
   - If time to collect statistics is very limited and the effort to tailor the RUNSTATS command on a table by table basis is a major issue, consider collecting statistics for the ″KEY″ columns only. It is assumed that the index contains the set of columns that are critical to the table and are most likely to appear in predicates.
   - If time to collect statistics is very limited and table statistics are to be gathered, consider using the TABLESAMPLE option to collect statistics on a subset of the table data.
   - If there are many indexes on the table and DETAILED (extended) information on the indexes might improve access plans, consider the SAMPLED option to reduce the time it takes to collect statistics. Regardless of whether you use the SAMPLED option, collecting detailed statistics on indexes is time consuming. Do not collect these statistics unless you are sure that they will be useful for your queries.
   - If there is skew in certain columns and predicates of the type ″column = constant″, it might be beneficial to specify a larger NUM_FREQVALUES value for that column
   - Collect distribution statistics for all columns that are used in equality predicates and for which the distribution of values might be skewed.
   - For columns that have range predicates (for example ″column >= constant″, ″column BETWEEN constant1 AND constant2″) or of the type ″column LIKE ′%xyz′″, it might be beneficial to specify a larger NUM_QUANTILES value.
   - If storage space is a concern and one cannot afford too much time on collecting statistics, do not specify high NUM_FREQVALUES or NUM_QUANTILES values for columns that are not used in predicates.
   - Note that if index statistics are requested, and statistics have never been run on the table containing the index, statistics on both the table and indexes are calculated.
3. After the command is run note the following:
   - A COMMIT should be issued to release the locks.
   - To allow new access plans to be generated, the packages that reference the target table must be rebound.
   - Executing the command on portions of the table could result in inconsistencies as a result of activity on the table since the command was last issued. In this case a warning message is returned. Issuing RUNSTATS on the table only might make table and index level statistics inconsistent. For example, you might collect index level statistics on a table and later delete a significant number of rows from the table. If you then issue RUNSTATS on the table only, the table cardinality might be less than FIRSTKEYCARD, which is an inconsistency. In the same way, if you collect statistics on a new index when you create it, the table level statistics might be inconsistent.

4. The RUNSTATS command will drop previously collected distribution statistics if table statistics are requested. For example, “RUNSTATS ON TABLE”, or “RUNSTATS ON TABLE ... AND INDEXES ALL” will cause previously collected distribution statistics to be dropped. If the command is run on indexes only then previously collected distribution statistics are retained. For example, “RUNSTATS ON TABLE ... FOR INDEXES ALL” will cause the previously collected distribution statistics to be retained.
5. For range-clustered tables, there is a special system-generated index in the catalog tables which represents the range ordering property of range-clustered tables. When statistics are collected on this type of table, if the table is to be included as part of the statistics collection, statistics will also be collected for the system-generated index. The statistics reflect the fast access of the range lookups by representing the index as a two-level index with as many pages as the base data table, and having the base data clustered perfectly along the index order.
6. In the ″On Dist Cols″ clause of the command syntax, the ″Frequency Option″ and ″Quantile Option″ parameters are currently not supported for Column GROUPS. These options are supported for single columns.
7. There are three prefetch statistics that cannot be computed when working in DMS mode. When looking at the index statistics in the index catalogs, you will see a “-1” value for the following statistics:
   - AVERAGE_SEQUENCE_FETCH_PAGES
   - AVERAGE_SEQUENCE_FETCH_GAP
   - AVERAGE_RANDOM_FETCH_PAGES
8. Runstats sampling through TABLESAMPLE only occurs with table data pages and not index pages. When index statistics as well as sampling is requested, all the index pages are scanned for statistics collection. It is only in the collection of table statistics where TABLESAMPLE is applicable. However, a more efficient collection of detailed index statistics is available through the SAMPLED DETAILED option. This is a different method of sampling than that employed by TABLESAMPLE and only applies to the detailed set of index statistics.
9. A statistics profile can be set or updated for the table specified in the RUNSTATS command, by using the set profile or update profile options. The statistics profile is stored in a visible string format, which represents the RUNSTATS command, in the STATISTICS_PROFILE column of the SYSIBM.SYSTABLES system catalog table.

**Examples:**

1. Collect statistics on the table only, on all columns without distribution statistics:

   ```
   RUNSTATS ON TABLE db2user.employee
   ```
2. Collect statistics on the table only, on columns empid and empname with distribution statistics:

   ```
   RUNSTATS ON TABLE  db2user.employee
        WITH DISTRIBUTION ON COLUMNS (empid, empname)
   ```
3. Collect statistics on the table only, on all columns with distribution statistics using a specified number of frequency limit for the table while picking the NUM_QUANTILES from the configuration setting:

   ```
   RUNSTATS ON TABLE db2user.employee WITH DISTRIBUTION DEFAULT
        NUM_FREQVALUES 50
   ```
4. Collect statistics on a set of indexes:

```
RUNSTATS ON TABLE db2user.employee for indexes
               db2user.empl1, db2user.empl2
```

5. Collect basic statistics on all indexes only:

```
RUNSTATS ON TABLE db2user.employee FOR INDEXES ALL
```

6. Collect basic statistics on the table and all indexes using sampling for the detailed index statistics collection:

```
RUNSTATS ON TABLE db2user.employee AND SAMPLED DETAILED INDEXES ALL
```

7. Collect statistics on table, with distribution statistics on columns empid, empname and empdept and the two indexes Xempid and Xempname. Distribution statistics limits are set individually for empdept, while the other two columns use a common default:

```
RUNSTATS ON TABLE db2user.employee
   WITH DISTRIBUTION ON COLUMNS (empid, empname, empdept NUM_FREQVALUES
       50 NUM_QUANTILES 100)
       DEFAULT NUM_FREQVALUES 5 NUM_QUANTILES 10
   AND INDEXES db2user.Xempid, db2user.Xempname
```

8. Collect statistics on all columns used in indexes and on all indexes:

```
RUNSTATS ON TABLE db2user.employee ON KEY COLUMNS AND INDEXES ALL
```

9. Collect statistics on all indexes and all columns without distribution except for one column. Consider T1 containing columns c1, c2, ...., c8

```
RUNSTATS ON TABLE db2user.T1
       WITH DISTRIBUTION  ON COLUMNS (c1, c2, c3 NUM_FREQVALUES 20
       NUM_QUANTILES 40, c4, c5, c6, c7, c8)
       DEFAULT NUM_FREQVALUES 0, NUM_QUANTILES 0
       AND INDEXES ALL
```

```
RUNSTATS ON TABLE db2user.T1
       WITH DISTRIBUTION ON COLUMNS (c3 NUM_FREQVALUES 20 NUM_QUANTILES 40)
       AND INDEXES ALL
```

10. Collect statistics on table T1 for the individual columns c1 and c5 as well as on the column combinations (c2, c3) and (c2, c4). Multi-column cardinality is very useful to the query optimizer when it estimates filter factors for predicates on columns in which the data is correlated.

```
RUNSTATS ON TABLE db2user.T1  ON COLUMNS (c1,  (c2,  c3),
(c2, c4),  c5)
```

11. Collect statistics on table T1 for the individual columns c1 and c2. For column c1 also collect the LIKE predicate statistics.

```
RUNSTATS ON TABLE db2user.T1  ON COLUMNS (c1 LIKE STATISTICS,  c2)
```

12. Register a statistics profile to collect statistics on the table only, on all columns with distribution statistics using a specified number of frequency limit for the table while picking the NUM_QUANTILES from the configuration setting. The command also updates the statistics as specified.

```
RUNSTATS ON TABLE db2user.employee WITH DISTRIBUTION DEFAULT
NUM_FREQVALUES 50 SET PROFILE
```

13. Register a statistics profile to collect statistics on the table only, on all columns with distribution statistics using a specified number of frequency limit for the table while picking the NUM_QUANTILES from the configuration setting. Statistics are not collected.

```
RUNSTATS ON TABLE db2user.employee WITH DISTRIBUTION
     DEFAULT NUM_FREQVALUES 50 SET PROFILE ONLY
```

14. Modify the previously registered statistics profile by changing the NUM_FREQVALUES value from 50 to 30. The command also updates the statistics as specified.

```
RUNSTATS ON TABLE db2user.employee WITH DISTRIBUTION
     DEFAULT NUM_FREQVALUES 30 UPDATE PROFILE
```

15. Modify the previously registered statistics profile by changing the NUM_FREQVALUES value from 50 to 30. Statistics are not collected.

    ```
    RUNSTATS ON TABLE db2user.employee WITH DISTRIBUTION
         DEFAULT NUM_FREQVALUES 30 UPDATE PROFILE ONLY
    ```

16. Modify the previously registered statistics profile by adding column empl_address and column group (empl_title, empl_salary) options. The command also updates the statistics as specified.

    ```
    RUNSTATS ON TABLE db2user.employee
         ON COLUMNS (empl_address, (empl_title, empl_salary)) UPDATE
         PROFILE
    ```

17. Modify the previously registered statistics profile by adding column empl_address and column group (empl_title, empl_salary) options. Statistics are not collected.

    ```
    RUNSTATS ON TABLE db2user.employee
         ON COLUMNS (empl_address, (empl_title, empl_salary)) UPDATE
         PROFILE ONLY
    ```

18. Collect statistics on a table using the options recorded in the statistics profile for that table:

    ```
    RUNSTATS ON TABLE db2user.employee USE PROFILE
    ```

19. Query the RUNSTATS command options corresponding to the previously registered statistics profile stored in the catalogs of the table:

    ```
    SELECT STATISTICS_PROFILE FROM SYSIBM.SYSTABLES WHERE NAME =
    'EMPLOYEE'
    ```

20. Collect statistics, including distribution statistics, on 30 percent of the rows:

    ```
    RUNSTATS ON TABLE db2user.employee  WITH DISTRIBUTION
         TABLESAMPLE BERNOULLI(30)
    ```

21. To control the sample set on which statistics will be collected and to be able to repeatedly use the same sample set, you can do so as follows:

    ```
    RUNSTATS ON TABLE db2user.employee  WITH DISTRIBUTION
         TABLESAMPLE BERNOULLI(30) REPEATABLE(4196)
    ```

    Issuing the same statement as above will result in the same set of statistics as long as the data has not changed in the interm.

22. Collect index statistics as well as table statistics on 1.5 percent of the data pages. Note that only table data pages and not index pages are sampled. In this example 1.5 percent of table data pages are used for the collection of table statistics, while for index statistics all the index pages will be used:

    ```
    RUNSTATS ON TABLE db2user.employee TABLESAMPLE SYSTEM(1.5)
    AND INDEXES ALL
    ```

**Related concepts:**

- "Collecting statistics using a statistics profile" in the *Administration Guide: Performance*
- "Collecting statistics on a sample of the table data" in the *Administration Guide: Performance*
- "Automatic statistics collection" in the *Administration Guide: Performance*

**Related tasks:**

- "Collecting catalog statistics" in the *Administration Guide: Performance*

# SET CLIENT

Specifies connection settings for the back-end process.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►─SET CLIENT─┬────────────────┬─┬───────────────────────────────┬──────────────►
              └─CONNECT──┬─1─┬─┘ └─DISCONNECT──┬─EXPLICIT────┬─┘
                         └─2─┘                 ├─CONDITIONAL─┤
                                               └─AUTOMATIC───┘

►─┬──────────────────────────────────┬─┬─────────────────────┬───────────────────►
  └─MAX_NETBIOS_CONNECTIONS──value─┘ └─SQLRULES──┬─DB2─┬─┘
                                                  └─STD─┘

►─┬──────────────────────────┬───────────────────────────────────────────────────►
  └─SYNCPOINT──┬─ONEPHASE─┬─┘
               ├─TWOPHASE─┤
               └─NONE─────┘

►─┬──────────────────────────────────────────────────────────────────┬───────────►
  └─CONNECT_DBPARTITIONNUM──┬─db-partition-number─────┬─┘
                            └─CATALOG_DBPARTITIONNUM─┘

►─┬─────────────────────────────────────────────┬───────────────────────────────►◄
  └─ATTACH_DBPARTITIONNUM──db-partition-number─┘
```

**Command parameters:**

**CONNECT**

**1** Specifies that a CONNECT statement is to be processed as a type 1 CONNECT.

**2** Specifies that a CONNECT statement is to be processed as a type 2 CONNECT.

**DISCONNECT**

**EXPLICIT**
Specifies that only database connections that have been explicitly marked for release by the RELEASE statement are to be disconnected at commit.

**CONDITIONAL**
Specifies that the database connections that have been marked RELEASE or have no open WITH HOLD cursors are to be disconnected at commit.

**AUTOMATIC**
Specifies that all database connections are to be disconnected at commit.

**MAX_NETBIOS_CONNECTIONS value**
Specifies the maximum number of concurrent connections that can be made in an application using a NetBIOS adapter. Maximum value is 254. This parameter must be set before the first NetBIOS connection is made. Changes subsequent to the first connection are ignored.

**SQLRULES**

**DB2** Specifies that a type 2 CONNECT is to be processed according to the DB2 rules.

**STD** Specifies that a type 2 CONNECT is to be processed according to the Standard (STD) rules based on ISO/ANS SQL92.

**SYNCPOINT**
Specifies how commits or rollbacks are to be coordinated among multiple database connections.

**ONEPHASE**
Specifies that no transaction manager (TM) is to be used to perform a two-phase commit. A one-phase commit is to be used to commit the work done by each database in multiple database transactions.

**TWOPHASE**
Specifies that the TM is required to coordinate two-phase commits among those databases that support this protocol.

**NONE**
Specifies that no TM is to be used to perform a two-phase commit, and does not enforce single updater, multiple reader. A COMMIT is sent to each participating database. The application is responsible for recovery if any of the commits fail.

**CONNECT_DBPARTITIONNUM (partitioned database environment only)**

**db-partition-number**
Specifies the database partition to which a connect is to be made. A value between zero and 999, inclusive. Overrides the value of the environment variable **DB2NODE**.

**CATALOG_DBPARTITIONNUM**
Specifying this value permits the client to connect to the catalog database partition of the database without knowing the identity of that database partition in advance.

**ATTACH_DBPARTITIONNUM db-partition-number (partitioned database environment only)**
Specifies the database partition to which an attach is to be made. A value between zero and 999, inclusive. Overrides the value of the environment variable **DB2NODE**.

For example, if database partitions 1, 2, and 3 are defined, the client only needs to be able to access one of these database partitions. If only database partition 1 containing databases has been cataloged, and this parameter is set to 3, then the next attach attempt will result in an attachment at database partition 3, after an initial attachment at database partition 1.

**Examples:**

To set specific values:

```
db2 set client connect 2 disconnect automatic sqlrules std
   syncpoint twophase
```

To change SQLRULES back to DB2, but keep the other settings:

```
db2 set client sqlrules db2
```

**Note:** The connection settings revert to default values after the TERMINATE command is issued.

**Usage notes:**

SET CLIENT cannot be issued if one or more connections are active.

If SET CLIENT is successful, the connections in the subsequent units of work will use the connection settings specified. If SET CLIENT is unsuccessful, the connection settings of the back-end process are unchanged.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword CONNECT_NODE can be substituted for CONNECT_DBPARTITIONNUM.
- The keyword CATALOG_NODE can be substituted for CATALOG_DBPARTITIONNUM.
- The keyword ATTACH_NODE can be substituted for ATTACH_DBPARTITIONNUM.

**Related reference:**

- "TERMINATE" on page 705
- "QUERY CLIENT" on page 587

# SET RUNTIME DEGREE

Sets the maximum run time degree of intra-partition parallelism for SQL statements for specified active applications.

**Scope:**

This command affects all database partitions that are listed in the `$HOME/sqllib/db2nodes.cfg` file.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

Instance. To change the maximum run time degree of intra-partition parallelism on a remote server, it is first necessary to attach to that server. If no attachment exists, the SET RUNTIME DEGREE command fails.

**Command syntax:**

```
►►──SET RUNTIME DEGREE FOR──┬─ALL──────────────────────────┬──TO──degree──────────►◄
                            │      ┌─,─────────────────┐   │
                            │      ▼                   │   │
                            └─(────application-handle──┴──)┘
```

**Command parameters:**

**FOR**

**ALL** The specified degree will apply to all applications.

**application-handle**
Specifies the agent to which the new degree applies. List the values using the LIST APPLICATIONS command.

**TO degree**
The maximum run time degree of intra-partition parallelism.

**Examples:**

The following example sets the maximum run time degree of parallelism for two users, with *application-handle* values of 41408 and 55458, to 4:

```
db2 SET RUNTIME DEGREE FOR ( 41408, 55458 ) TO 4
```

**Usage notes:**

This command provides a mechanism to modify the maximum degree of parallelism for active applications. It can be used to override the value that was determined at SQL statement compilation time.

The run time degree of intra-partition parallelism specifies the maximum number of parallel operations that will be used when the statement is executed. The degree of intra-partition parallelism for an SQL statement can be specified at statement compilation time using the CURRENT DEGREE special register or the **degree** bind option. The maximum run time degree of intra-partition parallelism for an active application can be specified using the SET RUNTIME DEGREE command. The *max_querydegree* database manager configuration parameter specifies the maximum run time degree for any SQL statement executing on this instance of the database manager.

The actual run time degree will be the lowest of:

- the *max_querydegree* configuration parameter
- the application run time degree
- the SQL statement compilation degree.

**Related reference:**

- "LIST APPLICATIONS" on page 480

## SET TABLESPACE CONTAINERS

A *redirected restore* is a restore in which the set of table space containers for the restored database is different from the set of containers for the original database at the time the backup was done. This command permits the addition, change, or removal of table space containers for a database that is to be restored. If, for example, one or more containers become inaccessible for any reason, the restore fails if it is not redirected to different containers.

**Note:** A redirected restore is not allowed when a user exit program is used to perform the restore.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

Database

**Command syntax:**

```
►►──SET TABLESPACE CONTAINERS FOR──tablespace-id─────────────────────────►

►─┬──────────────────────────────────────────────┬─USING──────────────────►
  │ ┌─REPLAY─┐                                   │
  └─┴─IGNORE─┴──ROLLFORWARD CONTAINER OPERATIONS─┘

        ┌─,──────────────────────────┐
        ▼                            │
►─┬─(───PATH──"container-string"─────┴─)──────────────────┬─────────────►◄
  │     ┌─,──────────────────────────────────────────────┐ │
  │     ▼                                                │ │
  └─(───┬─FILE───┬──"container-string"──number-of-pages──┴─)─┘
        └─DEVICE─┘
```

**Command parameters:**

**FOR tablespace-id**
An integer that uniquely represents a table space used by the database being restored.

**REPLAY ROLLFORWARD CONTAINER OPERATIONS**
Specifies that any ALTER TABLESPACE operation issued against this table space since the database was backed up is to be redone during a subsequent roll forward of the database.

**IGNORE ROLLFORWARD CONTAINER OPERATIONS**
Specifies that ALTER TABLESPACE operations in the log are to be ignored when performing a roll forward.

**USING PATH ″container-string″**
For an SMS table space, identifies one or more containers that will belong to the table space and into which the table space data will be stored. It is an absolute or relative directory name. If the directory name is not absolute, it is relative to the database directory. The string cannot exceed 240 bytes in length.

**USING FILE/DEVICE ″container-string″ number-of-pages**
For a DMS table space, identifies one or more containers that will belong to the table space and into which the table space data will be stored. The container type (either FILE or DEVICE) and its size (in 4KB pages) are specified. A mixture of file and device containers can be specified. The string cannot exceed 254 bytes in length.

For a file container, the string must be an absolute or relative file name. If the file name is not absolute, it is relative to the database directory.

For a device container, the string must be a device name. The device must already exist.

**Examples:**

See the example in RESTORE DATABASE.

**Usage notes:**

A backup of a database, or one or more table spaces, keeps a record of all the table space containers in use by the table spaces being backed up. During a restore, all containers listed in the backup are checked to see if they currently exist and are accessible. If one or more of the containers is inaccessible for any reason, the restore will fail. In order to allow a restore in such a case, the redirecting of table space containers is supported during the restore. This support includes adding, changing, or removing of table space containers. It is this command that allows the user to add, change or remove those containers.

**Related reference:**

- “BACKUP DATABASE” on page 280
- “RESTORE DATABASE” on page 647
- “ROLLFORWARD DATABASE” on page 657

## SET TAPE POSITION

When running on Windows NT-based operating systems, DB2 supports backup and restore operations to streaming tape devices. Use this command for tape positioning.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──SET TAPE POSITION──┬─────────────┬──TO──position──────────────────────►◄
                       └─ON──device──┘
```

**Command parameters:**

**ON device**
Specifies a valid tape device name. The default value is `\\.\TAPE0`.

**TO position**
Specifies the mark at which the tape is to be positioned. DB2 for Windows writes a tape mark after every backup image. A value of 1 specifies the first position, 2 specifies the second position, and so on. If the tape is positioned at tape mark 1, for example, archive 2 is positioned to be restored.

**Related reference:**

- "INITIALIZE TAPE" on page 472
- "REWIND TAPE" on page 656

# SET UTIL_IMPACT_PRIORITY

Changes the impact setting for a running utility. Using this command, you can:

- throttle a utility that was invoked in unthrottled mode
- unthrottle a throttled utility (disable throttling)
- reprioritize a throttled utility (useful if running multiple simultaneous throttled utilities)

**Scope:**

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance.

**Command syntax:**

```
►►──SET UTIL_IMPACT_PRIORITY FOR──utility-id──TO──priority──────────────────────►◄
```

**Command parameters:**

*utility-id*
: ID of the utility whose impact setting will be updated. IDs of running utilities can be obtained with the LIST UTILITIES command.

**TO** *priority*
: Specifies an instance-level limit on the impact associated with running a utility. A value of 100 represents the highest priority and 1 represents the lowest priority. Setting *priority* to 0 will force a throttled utility to continue unthrottled. Setting *priority* to a non-zero value will force an unthrottled utility to continue in throttled mode.

**Examples:**

The following example unthrottles the utility with ID 2.

```
SET UTIL_IMPACT_PRIORITY FOR 2 TO 0
```

The following example throttles the utility with ID 3 to priority 10. If the priority was 0 before the change then a previously unthrottled utility is now throttled. If the utility was previously throttled (priority had been set to a value greater than zero), then the utility has been reprioritized.

```
SET UTIL_IMPACT_PRIORITY FOR 3 TO 10
```

**Usage notes:**

Throttling requires having an impact policy defined by setting the *util_impact_lim* configuration parameter.

**Related reference:**

- “util_impact_lim - Instance impact policy configuration parameter” in the *Administration Guide: Performance*
- “LIST UTILITIES” on page 518

# SET WRITE

The SET WRITE command allows a user to suspend I/O writes or to resume I/O writes for a database. Typical use of this command is for splitting a mirrored database. This type of mirroring is achieved through a disk storage system.

This new state, SUSPEND_WRITE, is visible from the Snapshot Monitor. All table spaces must be in a NORMAL state for the command to execute successfully. If any one table space is in a state other than NORMAL, the command will fail.

**Authorization:**

This command only affect the node on which it is executed. The authorization of this command requires the issuer to have one of the following privileges:

- sysadm
- sysctrl
- sysmaint

**Required Connection:**

Database

**Command Syntax:**

```
>>──SET──WRITE──┬─SUSPEND─┬──FOR──┬─DATABASE─┬─────────────────────────────><
                └─RESUME──┘       └─DB───────┘
```

**Command Parameters:**

**SUSPEND**
: Suspending I/O writes will put all table spaces into a new state SUSPEND_WRITE state. Writes to the logs are also suspended by this command. All database operations, apart from online backup and restore, should function normally while database writes are suspended. However, some operations can wait while attempting to flush dirty pages from the buffer pool or log buffers to the logs. These operations will resume normally once the database writes are resumed.

**RESUME**
: Resuming I/O writes will remove the SUSPEND_WRITE state from all of the table spaces and make the table spaces available for update.

**Usage Notes:**

It is suggested that I/O writes be resumed from the same connection from which they were suspended. Ensuring that this connection is available to resume I/O writes involves not performing any operations from this connection until database writes are resumed. Otherwise, some operations can wait for I/O writes to be resumed if dirty pages must be flushed from the buffer pool or from log buffers to the logs. Furthermore, subsequent connection attempts might hang if they require flushing dirty pages from the buffer pool to disk. Subsequent connections will complete successfully once database I/O resumes. If your connection attempts are hanging, and it has become impossible to resume I/O from the connection that you used to suspend I/O, then you will have to run the RESTART DATABASE command with the WRITE RESUME option. When used in this circumstance, the RESTART DATABASE command will resume I/O writes without performing crash recovery. The RESTART DATABASE command with the WRITE RESUME option will only perform crash recovery when you use it after a database crash.

**Related concepts:**

- "High availability through online split mirror and suspended I/O support" in the *Data Recovery and High Availability Guide and Reference*

# START DATABASE MANAGER

Starts the current database manager instance background processes on a single database partition or on all the database partitions defined in a multi-partitioned database environment.

**Scope:**

In a multi-partitioned database environment, this command affects all database partitions that are listed in the `$HOME/sqllib/db2nodes.cfg` file, unless the *dbpartitionnum* parameter is used.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Note:** The ADD DBPARTITIONNUM start option requires either *sysadm* or *sysctrl* authority.

**Required connection:**

None

**Command syntax:**

```
>>──┬─START──┬─DATABASE MANAGER─┬─┬──────────────────────────────────────────────►
    │        ├─DB MANAGER───────┤ │
    │        └─DBM──────────────┘ │
    └─db2start────────────────────┘

►──┬───────────────────────────────────────────────────────────────┬─────────►
   └─REMOTE──┬──────────┬──instancename──┤ remote options ├────────┘
             └─INSTANCE─┘

►──┬────────────────────────────────┬──┬──────────────────┬──────────────────►
   └─ADMIN MODE──┬───────────────────┬┘  └─PROFILE──profile─┘
                 ├─USER──username────┤
                 └─GROUP──groupname──┘

►──┬────────────────────────────────────────────────────────────┬───────────►◄
   └─DBPARTITIONNUM──db-partition-number──┤ start options ├─────┘
```

### remote options:

```
├──┬─ADMINNODE──nodename─┬──USER──username──USING──password──────────────────┤
   └─HOSTNAME──hostname──┘
```

### start options:

```
├──┬─────────────────────────────────────────────────────────┬──────────────────────┤
   ├─ADD DBPARTITIONNUM──┤ add dbpartitionnum options ├──────┤
   ├─STANDALONE──────────────────────────────────────────────┤
   └─RESTART──┤ restart options ├────────────────────────────┘
```

**add dbpartitionnum options:**

```
├──HOSTNAME──hostname──PORT──logical-port──┬──────────────────────────────┬──────────►
                                           └─COMPUTER──computer-name──────┘

►──┬───────────────────┬──┬───────────────────────┬──┬─────────────────────┬─────────►
   └─USER──username────┘  └─PASSWORD──password────┘  └─NETNAME──netname────┘

►──┬─────────────────────────────────────────────┬───────────────────────────────────┤
   ├─LIKE DBPARTITIONNUM──db-partition-number────┤
   └─WITHOUT TABLESPACES─────────────────────────┘
```

**restart options:**

```
├──┬───────────────────────┬──┬───────────────────────┬──┬──────────────────────────────┬──►
   └─HOSTNAME──hostname────┘  └─PORT──logical-port────┘  └─COMPUTER──computername──────┘

►──┬───────────────────┬──┬───────────────────────┬──┬─────────────────────┬─────────────┤
   └─USER──username────┘  └─PASSWORD──password────┘  └─NETNAME──netname────┘
```

**Command parameters:**

**REMOTE [INSTANCE] instancename**
Specifies the name of the remote instance you wish to start.

**ADMINNODE nodename**
With REMOTE, or REMOTE INSTANCE, specifies the name of the administration node.

**HOSTNAME hostname**
With REMOTE, or REMOTE INSTANCE, specifies the name of the host node.

**USER username**
With REMOTE, or REMOTE INSTANCE, specifies the name of the user.

**USING password**
With REMOTE, or REMOTE INSTANCE, and the USER, specifies the password of the user.

**ADMIN MODE**
Starts the instance in quiesced mode for administration purposes. This is equivalent to the QUIESCE INSTANCE command except in this case the instance is not already "up", and therefore there is no need to force the connections off.

**USER username**
With ADMIN MODE, specifies the name of the user.

**GROUP groupname**
With ADMIN MODE, specifies the name of the group.

**Note:** All of the following parameters are valid in an Enterprise Server Edition (ESE) environment only.

**PROFILE profile**

Specifies the name of the profile file to be executed at each database partition to define the DB2 environment. This file is executed before the database partitions are started. The profile file must reside in the `sqllib` directory of the instance owner.

**Note:** The environment variables in the profile file are not necessarily all defined in the user session.

**DBPARTITIONNUM db-partition-number**

Specifies the database partition to be started. If no other options are specified, a normal startup is done at this database partition.

Valid values are from 0 to 999 inclusive. If ADD DBPARTITIONNUM is not specified, the value must already exist in the `db2nodes.cfg` file of the instance owner. If no database partition number is specified, all database partitions defined in the configuration file are started.

**ADD DBPARTITIONNUM**

Specifies that the new database partition is added to the `db2nodes.cfg` file of the instance owner with the *hostname* and *logical-port* values.

Ensure that the combination of *hostname* and *logical-port* is unique.

The add database partition utility is executed internally to create all existing databases on the database partition being added. After a database partition is added, the `db2nodes.cfg` file is not updated with the new database partition until a **db2stop** is issued. The database partition is not part of the MPP system until the next **db2start** following the **db2stop**.

**Note:** When the database partitions are created on the new node, their configuration parameters are set to the default.

**HOSTNAME hostname**

With ADD DBPARTITIONNUM, specifies the host name to be added to the `db2nodes.cfg` file.

**PORT logical-port**

With ADD DBPARTITIONNUM, specifies the logical port to be added to the `db2nodes.cfg` file. Valid values are from 0 to 999.

**COMPUTER computername**

The computer name for the machine on which the new database partition is created. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

**USER username**

The user name for the account on the new database partition. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

**PASSWORD password**

The password for the account on the new database partition. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

**NETNAME netname**

Specifies the *netname* to be added to the `db2nodes.cfg` file. If not specified, this parameter defaults to the value specified for *hostname*.

**LIKE DBPARTITIONNUM db-partition-number**
Specifies that the containers for the system temporary table spaces will be the same as the containers on the specified *db-partition-number* for each database in the instance. The database partition specified must be a database partition that is already in the `db2nodes.cfg` file.

**WITHOUT TABLESPACES**
Specifies that containers for the system temporary table spaces are not created for any of the databases. The ALTER TABLESPACE statement must be used to add system temporary table space containers to each database before the database can be used.

**STANDALONE**
Specifies that the database partition is to be started in stand-alone mode. FCM does not attempt to establish a connection to any other database partition. This option is used when adding a database partition.

**RESTART**
Starts the database manager after a failure. Other database partitions are still operating, and this database partition attempts to connect to the others. If neither the *hostname* nor the *logical-port* parameter is specified, the database manager is restarted using the *hostname* and *logical-port* values specified in `db2nodes.cfg`. If either parameter is specified, the new values are sent to the other database partitions when a connection is established. The `db2nodes.cfg` file is updated with this information.

**HOSTNAME hostname**
With RESTART, specifies the host name to be used to override that in the database partition configuration file.

**PORT logical-port**
With RESTART, specifies the logical port number to be used to override that in the database partition configuration file. If not specified, this parameter defaults to the *logical-port* value that corresponds to the *num* value in the `db2nodes.cfg` file. Valid values are from 0 to 999.

**COMPUTER computername**
The computer name for the machine on which the new database partition is created. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

**USER username**
The user name for the account on the new database partition. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

**PASSWORD password**
The password for the account on the new database partition. This parameter is mandatory on Windows NT, but is ignored on other operating systems.

**NETNAME netname**
Specifies the *netname* to override that specified in the `db2nodes.cfg` file. If not specified, this parameter defaults to the *netname* value that corresponds to the *db-partition-number* value in the `db2nodes.cfg` file.

**Examples:**

The following is sample output from **db2start** issued on a three-database partition system with database partitions 10, 20, and 30:

```
04-07-1997 10:33:05    10   0   SQL1063N  DB2START processing was successful.
04-07-1997 10:33:07    20   0   SQL1063N  DB2START processing was successful.
04-07-1997 10:33:07    30   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

**Usage notes:**

It is not necessary to issue this command on a client node. It is provided for compatibility with older clients, but it has no effect on the database manager.

Once started, the database manager instance runs until the user stops it, even if all application programs that were using it have ended.

If the database manager starts successfully, a successful completion message is sent to the standard output device. If an error occurs, processing stops, and an error message is sent to the standard output device. In a partitioned database environment, messages are returned on the database partition that issued the START DATABASE MANAGER command.

If no parameters are specified in a partitioned database environment, the database manager is started on all parallel nodes using the parameters specified in the database partition configuration file.

If a START DATABASE MANAGER command is in progress, ensure that the applicable database partitions have started *before* issuing a request to the database.

The db2cshrc file is not supported and cannot be used to define the environment.

You can start an instance in a quiesced state. You can do this by using one of the following choices:

```
db2start admin mode
```

or

```
db2start admin mode user username
```

or

```
db2start admin mode group groupname
```

On UNIX platforms, the START DATABASE MANAGER command supports the SIGINT and SIGALRM signals. The SIGINT signal is issued if CTRL+C is pressed. The SIGALRM signal is issued if the value specified for the *start_stop_time* database manager configuration parameter is reached. If either signal occurs, all in-progress startups are interrupted and a message (SQL1044N for SIGINT and SQL6037N for SIGALRM) is returned from each interrupted database partition to the $HOME/sqllib/log/db2start. *timestamp*.log error log file. Database partitions that are already started are not affected. If CTRL+C is pressed on a database partition that is starting, **db2stop** must be issued on that database partition before an attempt is made to start it again.

On the Windows NT operating system, neither the **db2start** command nor the **NET START** command returns warnings if any communication subsystem failed to start. The database manager in a Windows NT environment is implemented as an NT service, and does not return an error if the service is started successfully. Be sure to examine the NT Event Log or the `DB2DIAG.LOG` file for any errors that might have occurred during the running of **db2start**.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keywords LIKE NODE can be substituted for LIKE DBPARTITIONNUM.
- The keyword ADDNODE can be substituted for ADD DBPARTITIONNUM.
- The keyword NODENUM can be substituted for DBPARTITIONNUM.

**Related reference:**

- "STOP DATABASE MANAGER" on page 698
- "ADD DBPARTITIONNUM" on page 271

# START HADR

Starts HADR operations for a database.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance. The command establishes a database connection if one does not exist, and closes the database connection when the command completes.

**Command syntax:**

```
>>──START HADR ON──┬─DATABASE─┬──database-alias──────────────────────────────────>
                   └─DB───────┘

>──┬─────────────────────────────────────────┬──AS──┬─PRIMARY──┬──────────┬─┬───────────><
   └─USER──user-name──┬────────────────────┬─┘      │          └─BY FORCE─┘ │
                      └─USING──password────┘        └─STANDBY───────────────┘
```

**Command parameters:**

**DATABASE** *database-alias*
: Identifies the database on which HADR operations are to start.

**USER** *user-name*
: Identifies the user name under which the HADR operations are to be started.

  **USING** *password*
  : The password used to authenticate *user-name*.

**AS PRIMARY**
: Specifies that HADR primary operations are to be started on the database.

  **BY FORCE**
  : Specifies that the HADR primary database will not wait for the standby database to connect to it. After a start BY FORCE, the primary database will still accept valid connections from the standby database whenever the standby later becomes available.

  **Caution:** Use the START HADR command with the AS PRIMARY BY FORCE option with caution. If the standby database has been changed to a primary and the original primary database is restarted by issuing the START HADR command with the AS PRIMARY BY FORCE option, both copies of your database will be operating independently as primaries. (This is sometimes referred to as *split brain* or *dual primary*.) In this case, each primary database can accept connections and perform transactions, and neither receives and replays the updates made by the other. As a result, the two copies of the database will become inconsistent with each other.

**AS STANDBY**

Specifies that HADR standby operations are to be started on the database. The standby database will attempt to connect to the HADR primary database until a connection is successfully established, or until the connection attempt is explicitly rejected by the primary. (The connection might be rejected by the primary database if an HADR configuration parameter is set incorrectly or if the database copies are inconsistent, both conditions for which continuing to retry the connection is not appropriate.)

**Usage notes:**

The following table shows database behavior in various conditions:

| Database status | Behavior upon START HADR command with the AS PRIMARY option | Behavior upon START HADR command with the AS STANDBY option |
|---|---|---|
| Inactive standard database | Activated as HADR primary database. | Database starts as an standby database if it is in rollforward-pending mode (which can be the result of a restore or a split mirror) or in rollforward in-progress mode. Otherwise, an error is returned. |
| Active standard database | Database enters HADR primary role. | Error message returned. |
| Inactive primary database | Activated as HADR primary database. | After a failover, this reintegrates the failed primary into the HADR pair as the new standby database. Some restrictions apply. |
| Active primary database | Warning message issued. | Error message returned. |
| Inactive standby database | Error message returned. | Starts the database as the standby database. |
| Active standby database | Error message returned. | Warning message issued. |

# STOP DATABASE MANAGER

Stops the current database manager instance. Unless explicitly stopped, the database manager continues to be active. This command does not stop the database manager instance if any applications are connected to databases. If there are no database connections, but there are instance attachments, it forces the instance attachments and stops the database manager. This command also deactivates any outstanding database activations before stopping the database manager.

On partitioned database system, this command stops the current database manager instance on a database partition or on all database partitions. When it stops the database manager on all database partitions, it uses the `db2nodes.cfg` configuration file to obtain information about each database partition.

This command can also be used to drop a database partition from the `db2nodes.cfg` file (partitioned database systems only).

This command is not valid on a client.

**Scope:**

By default, and in a partitioned database environment, this command affects all database partitions that are listed in the `db2nodes.cfg` file.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

None

**Command syntax:**

```
►►──┬─STOP──┬─DATABASE MANAGER─┬─┬──┬──────────────────┬─────────────────────►
    │       ├─DB MANAGER───────┤ │  └─PROFILE──profile─┘
    │       └─DBM──────────────┘ │
    └─db2stop────────────────────┘

►──┬──────────────────────────────────────────────┬─────────────────────────►◄
   ├─DBPARTITIONNUM──db-partition-number──────────┤
   ├─DROP DBPARTITIONNUM──db-partition-number─────┤
   └─FORCE──┬───────────────────────────────────┬─┘
            └─DBPARTITIONNUM──db-partition-number─┘
```

**Command parameters:**

**PROFILE profile**
partitioned database systems only. Specifies the name of the profile file that was executed at startup to define the DB2 environment for those database partitions that were started. If a profile for the START DATABASE

MANAGER command was specified, the same profile must be specified here. The profile file must reside in the sqllib directory of the instance owner.

**DBPARTITIONNUM db-partition-number**

partitioned database systems only. Specifies the database partition to be stopped.

Valid values are from 0 to 999 inclusive, and must be in the db2nodes.cfg file. If no database partition number is specified, all database partitions defined in the configuration file are stopped.

**DROP DBPARTITIONNUM db-partition-number**

partitioned database systems only. Specifies the database partition to be dropped from the db2nodes.cfg file.

Before using this parameter, run the DROP DBPARTITIONNUM VERIFY command to ensure that there is no user data on this database partition.

When this option is specified, all database partitions in the db2nodes.cfg file are stopped.

**FORCE**

Specifies to use FORCE APPLICATION ALL when stopping the database manager at each database partition.

**DBPARTITIONNUM db-partition-number**

partitioned database systems only. Specifies the database partition to be stopped after all applications on that database partition have been forced to stop. If the FORCE option is used without this parameter, all applications on all database partitions are forced before all the database partitions are stopped.

**Examples:**

The following is sample output from **db2stop** issued on a three-partition system with database partitions 10, 20, and 30:

```
04-07-1997 10:32:53    10   0   SQL1064N  DB2STOP processing was successful.
04-07-1997 10:32:54    20   0   SQL1064N  DB2STOP processing was successful.
04-07-1997 10:32:55    30   0   SQL1064N  DB2STOP processing was successful.
SQL1064N  DB2STOP processing was successful.
```

**Usage notes:**

It is not necessary to issue this command on a client node. It is provided for compatibility with older clients, but it has no effect on the database manager.

Once started, the database manager instance runs until the user stops it, even if all application programs that were using it have ended.

If the database manager is stopped, a successful completion message is sent to the standard output device. If an error occurs, processing stops, and an error message is sent to the standard output device.

If the database manager cannot be stopped because application programs are still connected to databases, use the FORCE APPLICATION command to disconnect all users first, or reissue the STOP DATABASE MANAGER command with the FORCE option.

The following information applies to partitioned database environments only:

- If no parameters are specified, the database manager is stopped on each database partition listed in the configuration file. The administration notification log might contain messages to indicate that other database partitions are shutting down.
- Any database partitions added to the partitioned database system since the previous STOP DATABASE MANAGER command was issued will be updated in the `db2nodes.cfg` file.
- On UNIX platforms, this command supports the SIGALRM signal, which is issued if the value specified for the *start_stop_time* database manager configuration parameter is reached. If this signal occurs, all in-progress stops are interrupted, and message SQL6037N is returned from each interrupted database partition to the `$HOME/sqllib/log/db2stop.` *timestamp*`.log` error log file. Database partitions that are already stopped are not affected.
- The `db2cshrc` file is not supported and cannot be specified as the value for the PROFILE parameter.

**Attention:** The UNIX **kill** command should *not* be used to terminate the database manager because it will abruptly end database manager processes without controlled termination and cleanup processing.

**Related reference:**

- "FORCE APPLICATION" on page 372
- "START DATABASE MANAGER" on page 690
- "DEACTIVATE DATABASE" on page 342
- "DROP DBPARTITIONNUM VERIFY" on page 358

# STOP HADR

Stops HADR operations for a database.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance. The command establishes a database connection if one does not exist, and closes the database connection when the command completes.

**Command syntax:**

```
►►──STOP HADR ON──┬─DATABASE─┬──database-alias──────────────────────────────►
                  └─DB───────┘

►──┬──────────────────────────────────────┬──────────────────────────────►◄
   └─USER──user-name──┬─────────────────┬─┘
                      └─USING──password─┘
```

**Command parameters:**

**DATABASE** *database-alias*
Identifies the database on which HADR operations are to stop.

**USER** *user-name*
Identifies the user name under which the HADR operations are to be stopped.

**USING** *password*
The password used to authenticate *user-name*.

**Usage notes:**

The following table shows database behavior in various conditions:

| Database status | Behavior upon STOP HADR command |
|---|---|
| Inactive standard database | Error message returned. |
| Active standard database | Error message returned. |
| Inactive primary database | Database role changes to standard. Database configuration parameter *hadr_db_role* is updated to STANDARD. Database remains offline. At the next restart, enters standard role. |

| Database status | Behavior upon STOP HADR command |
|---|---|
| Active primary database | Stops shipping logs to the HADR standby database and shuts down all HADR EDUs on the HADR primary database. Database role changes to standard and database remains online. Database remains in standard role until an explicit START HADR command with the AS PRIMARY option is issued. Open sessions and transactions are not affected by the STOP HADR command. You can repeatedly issue STOP HADR and START HADR commands while the database remains online. These commands take effect dynamically. |
| Inactive standby database | Database role changes to standard. Database configuration parameter *hadr_db_role* is updated to STANDARD. Database remains offline. Database is put into rollforward pending mode. |
| Active standby database | Error message returned: Deactivate the standby database before attempting to convert it to a standard database. |

# TAKEOVER HADR

Instructs an HADR standby database to take over as the new HADR primary database for the HADR pair.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance. The command establishes a database connection if one does not exist, and closes the database connection when the command completes.

**Command syntax:**

```
►►──TAKEOVER HADR ON──┬─DATABASE─┬──database-alias──────────────────────────►
                      └─DB───────┘

►──┬──────────────────────────────────────┬──┬──────────┬─────────────────►◄
   └─USER──user-name──┬────────────────┬──┘  └─BY FORCE─┘
                      └─USING──password─┘
```

**Command parameters:**

**DATABASE** *database-alias*
: Identifies the current HADR standby database that should take over as the HADR primary database.

**USER** *user-name*
: Identifies the user name under which the takeover operation is to be started.

  **USING** *password*
  : The password used to authenticate *user-name*.

**BY FORCE**
: Specifies that the database will not wait for confirmation that the original HADR primary database has been shut down. This option is required if the HADR pair is not in peer state.

**Usage notes:**

The following table shows the behavior of the TAKEOVER HADR command when issued on an active standby for each possible state and option combination. An error message is returned if this command is issued on an inactive standby database.

| Standby state | BY FORCE option used | Takeover behavior |
|---|---|---|
| Local catchup or remote catchup | No | Error message returned |

| Standby state | BY FORCE option used | Takeover behavior |
|---|---|---|
| Local catchup or remote catchup | Yes | Error message returned |
| Peer | No | Primary database and standby database switch roles.<br>If no failure is encountered during takeover, there will be no data loss. However, if failures are encountered during takeover, data loss might occur and the roles of the primary and standby might or might not have been changed. The following is a guideline for handling failures during a takeover in which the primary and standby switch roles:<br>1. If a failure occurs during a takeover operation, the roles of the HADR systems might or might not have been changed. If possible, make sure both databases are online. Check the HADR role of the available database or databases using the Snapshot Monitor, or by checking the value of the database configuration parameter *hadr_db_role*.<br>2. If the intended new primary is still in standby role, and takeover is still desired, re-issue the TAKEOVER HADR command (see the next guideline regarding the BY FORCE option).<br>3. It is possible to end up with both databases in standby role. In that case, the BY FORCE option can be issued at whichever node should now become the primary. The BY FORCE option is required in this case because the two standbys cannot establish the usual HADR primary-standby connection. |
| Peer | Yes | The standby notifies the primary to shut itself (the primary) down. The standby stops receiving logs from the primary, finishes replaying the logs it has already received, and then becomes a primary. The standby does *not* wait for any acknowledgement from the primary to confirm that it has received the takeover notification or that it has shut down. |
| Remote catchup pending | No | Error message returned. |
| Remote catchup pending | Yes | The standby becomes a primary. |

# TERMINATE

Explicitly terminates the command line processor's back-end process.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──TERMINATE──────────────────────────────────────────────────────►◄
```

**Command parameters:**
None

**Usage notes:**
If an application is connected to a database, or a process is in the middle of a unit of work, TERMINATE causes the database connection to be lost. An internal commit is then performed.

Although TERMINATE and CONNECT RESET both break the connection to a database, only TERMINATE results in termination of the back-end process.

It is recommended that TERMINATE be issued prior to executing the **db2stop** command. This prevents the back-end process from maintaining an attachment to a database manager instance that is no longer available.

Back-end processes in MPP systems must also be terminated when the **DB2NODE** environment variable is updated in the session. This environment variable is used to specify the coordinator database partition number within an MPP multiple logical node configuration.

## UNCATALOG DATABASE

Deletes a database entry from the system database directory.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──UNCATALOG──┬─DATABASE─┬──database-alias──────────────────────────────►◄
               └─DB───────┘
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the database to uncatalog.

**Usage notes:**

Only entries in the system database directory can be uncataloged. Entries in the local database directory can be deleted using the DROP DATABASE command.

To recatalog the database on the instance, use the CATALOG DATABASE command. To list the databases that are cataloged on a node, use the LIST DATABASE DIRECTORY command.

The authentication type of a database, used when communicating with a down-level server, can be changed by first uncataloging the database, and then cataloging it again with a different type.

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. See the information for the configuration parameter *dir_cache* in the GET DATABASE MANAGER CONFIGURATION command. An application's directory cache is created during its first directory lookup. Because the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**

- "CATALOG DATABASE" on page 307
- "DROP DATABASE" on page 352
- "GET DATABASE MANAGER CONFIGURATION" on page 395

- “LIST DATABASE DIRECTORY” on page 483
- “TERMINATE” on page 705

# UNCATALOG DCS DATABASE

Deletes an entry from the Database Connection Services (DCS) directory.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──UNCATALOG DCS──┬─DATABASE─┬──database-alias──────────────────────────────►◄
                   └─DB───────┘
```

**Command parameters:**

**DATABASE database-alias**
Specifies the alias of the DCS database to uncatalog.

**Usage notes:**

DCS databases are also cataloged in the system database directory as remote databases and can be uncataloged using the UNCATALOG DATABASE command.

To recatalog a database in the DCS directory, use the CATALOG DCS DATABASE command. To list the DCS databases that are cataloged on a node, use the LIST DCS DIRECTORY command.

**Note:** If directory caching is enabled. database, node, and DCS directory files are cached in memory. See the information provided for the configuration parameter *dir_cache* in the output of the GET DATABASE MANAGER CONFIGURATION command. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use the TERMINATE command. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**
- "CATALOG DCS DATABASE" on page 311
- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "TERMINATE" on page 705
- "UNCATALOG DATABASE" on page 706
- "LIST DCS DIRECTORY" on page 493

# UNCATALOG LDAP DATABASE

Used to deregister the database from Lightweight Directory Access Protocol (LDAP).

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──UNCATALOG LDAP──┬─DATABASE─┬──dbalias──────────────────────────────────►
                    └─DB───────┘

►──┬──────────────────────────────────────────┬───────────────────────────►◄
   └─USER──username──┬──────────────────────┬─┘
                     └─PASSWORD──password───┘
```

**Command parameters:**

**DATABASE dbalias**
Specifies the alias of the LDAP database to uncatalog.

**USER username**
Specifies the user’s LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to delete the object from the LDAP directory. If the user’s LDAP DN is not specified, the credentials of the current logon user will be used.

**PASSWORD password**
Account password.

**Usage notes:**

When a database is dropped, the database object is removed from LDAP. The database is also automatically deregistered from LDAP when the database server that manages the database is deregistered from LDAP. It might, however, be necessary to manually uncatalog the database from LDAP if:

- The database server does not support LDAP. The administrator must manually uncatalog each database from LDAP after the database is dropped.
- During DROP DATABASE the database object cannot be removed from LDAP (because LDAP cannot be accessed). In this case, the database is still removed from the local machine, but the existing entry in LDAP is not deleted.

**Related reference:**

- “CATALOG LDAP DATABASE” on page 313
- “CATALOG LDAP NODE” on page 316
- “UNCATALOG LDAP NODE” on page 710

# UNCATALOG LDAP NODE

Uncatalogs a node entry in Lightweight Directory Access Protocol (LDAP).

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──UNCATALOG LDAP──NODE──nodename──┬──────────────────────────────────────────┬──►◄
                                    └─USER──username──┬──────────────────────┬─┘
                                                      └─PASSWORD──password───┘
```

**Command parameters:**

**NODE nodename**
Specifies the name of the node to uncatalog.

**USER username**
Specifies the user's LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to delete the object from the LDAP directory. If the user's LDAP DN is not specified, the credentials of the current logon user will be used.

**PASSWORD password**
Account password.

**Usage notes:**

The LDAP node is automatically uncataloged when the DB2 server is deregistered from LDAP.

**Related reference:**

- "CATALOG LDAP DATABASE" on page 313
- "UNCATALOG LDAP DATABASE" on page 709
- "CATALOG LDAP NODE" on page 316

# UNCATALOG NODE

Deletes an entry from the node directory.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

None. Directory operations affect the local directory only.

**Command syntax:**

```
►►──UNCATALOG NODE──nodename──────────────────────────────────────────►◄
```

**Command parameters:**

**NODE nodename**
Specifies the node entry being uncataloged.

**Usage notes:**

UNCATALOG NODE can be executed on any type of node, but only the local directory is affected, even if there is an attachment to a remote instance, or a different local instance.

**Note:** If directory caching is enabled, database, node, and DCS directory files are cached in memory. An application's directory cache is created during its first directory lookup. Since the cache is only refreshed when the application modifies any of the directory files, directory changes made by other applications might not be effective until the application has restarted.

To refresh the CLP's directory cache, use TERMINATE. To refresh DB2's shared cache, stop (**db2stop**) and then restart (**db2start**) the database. To refresh the directory cache for another application, stop and then restart that application.

**Related reference:**
- "CATALOG APPC NODE" on page 303
- "CATALOG TCPIP NODE" on page 324
- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "TERMINATE" on page 705
- "CATALOG NETBIOS NODE" on page 321
- "CATALOG LOCAL NODE" on page 317
- "CATALOG APPN NODE" on page 305
- "CATALOG NAMED PIPE NODE" on page 319

# UNCATALOG ODBC DATA SOURCE

Uncatalogs a user or system ODBC data source.

A *data source*, in ODBC (Open Database Connectivity) terminology, is a user-defined name for a specific database. That name is used to access the database through ODBC APIs. On Windows, either user or system data sources can be uncataloged. A user data source is only visible to the user who cataloged it, whereas a system data source is visible to and can be used by all other users.

This command is available on Windows only.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
               ┌─USER───┐
►►──UNCATALOG──┤        ├──ODBC DATA SOURCE──data-source-name──────────────────►◄
               └─SYSTEM─┘
```

**Command parameters:**

**USER** Uncatalog a user data source. This is the default if no keyword is specified.

**SYSTEM**
Uncatalog a system data source.

**ODBC DATA SOURCE data-source-name**
Specifies the name of the data source to be uncataloged. Maximum length is 32 characters.

**Related reference:**

- "CATALOG ODBC DATA SOURCE" on page 323
- "LIST ODBC DATA SOURCES" on page 507

# UNQUIESCE

Restores user access to instances or databases which have been quiesced for maintenance or other reasons. UNQUIESCE restores user access without necessitating a shutdown and database restart.

Unless specifically designated, no user except those with *sysadm*, *sysmaint*, or *sysctrl* has access to a database while it is quiesced. Therefore an UNQUIESCE is required to restore general access to a quiesced database.

**Scope:**

UNQUIESCE DB restores user access to all objects in the quiesced database.

UNQUIESCE INSTANCE *instance-name* restores user access to the instance and the databases in the instance *instance-name*.

To stop the instance and unquiesce it and all its databases, issue the `db2stop` command. Stopping and restarting DB2 will unquiesce all instances and databases.

**Authorization:**

One of the following:

For database level unquiesce:
- *sysadm*
- *dbadm*

For instance level unquiesce:
- *sysadm*
- *sysctrl*

**Command syntax:**

```
►►──UNQUIESCE──┬─DB─────────────────────┬───────────────────────────────────►◄
               └─INSTANCE──instance-name─┘
```

**Required connection:**
Database

(Database connection is not required for an instance unquiesce.)

**Command parameters:**

**DB** Unquiesce the database. User access will be restored to all objects in the database.

**INSTANCE** *instance-name*
Access is restored to the instance *instance-name* and the databases in the instance.

**Examples:**

**Unquiescing a Database**

```
db2 unquiesce db
```

This command will unquiesce the database that had previously been quiesced.

# UPDATE ADMIN CONFIGURATION

Modifies specified entries in the DB2 Administration Server (DAS) configuration file. The DAS is a special administrative tool that enables remote administration of DB2 servers.

When you install the DAS, a blank copy of the configuration file is stored on each physical partition. You must create entries in each copy. You can specify the following DAS configuration parameters to be used the next time you start the DAS:

- Name of the DB2 Server System - db2system
- DAS Administration Authority Group Name - dasadm_group
- Scheduler Mode - sched_enable
- Tools Catalog Database Instance - toolscat_inst
- Tools Catalog Database - toolscat_db
- Tools Catalog Database Schema - toolscat_schema
- Execute Expired Tasks - exec_exp_task
- Scheduler User ID - sched_userid
- Authentication Type DAS - authentication

The following DAS configuration parameters can be specified originally and then later changed while the DAS is online:

- DAS Discovery Mode - discover
- SMTP Server - smtp_server
- Java Development Kit Installation Path DAS - jdk_path
- Location of Contact List - contact_host
- DAS Code Page - das_codepage
- DAS Territory - das_territory

For more information about these parameters, see individual parameter descriptions.

**Scope:**

Issue this command from each administration node to specify or change parameter settings for that node.

**Authorization:**

*dasadm*

**Required connection:**

Node. To update the DAS configuration for a remote system, use the FOR NODE option with the administrator node name.

**Command syntax:**

```
                                                     ┌──────────────────────┐
                                                     ▼                      │
►►──UPDATE ADMIN──┬─CONFIGURATION─┬──USING────config-keyword value─┴──────────►
                  ├─CONFIG────────┤
                  └─CFG───────────┘
```

```
►─┬──────────────────────────────────────────────────────────┬──────────────►◄
  └─FOR NODE──node-name──┬─────────────────────────────────┬─┘
                         └─USER──username──USING──password─┘
```

**Command parameters:**

**USING config-keyword value**
Specifies the admin configuration parameter to be updated.

**FOR NODE**
Enter the name of an administration node to update the DAS configuration parameters there.

**USER username USING** *password*
If connection to the administration node requires user name and password authorization, enter this information.

**Usage notes:**

To view or print a list of the DAS configuration parameters, use GET ADMIN CONFIGURATION. To reset the DAS configuration parameters to the recommended DAS defaults, use RESET ADMIN CONFIGURATION.

When configuration parameters take effect depends on whether you change a standard configuration parameter or one of the parameters that can be reset online. Standard configuration parameter values are reset when you execute the **db2admin** command.

If an error occurs, the DAS configuration file is not changed.

In order to update the DAS configuration using UPDATE ADMIN CONFIGURATION, you must use the command line processor from an instance that is at the same installed level as the DAS.

The DAS configuration file cannot be updated if the checksum is invalid. This might occur if you change the DAS configuration file manually, without using the appropriate command. If this happens, you must drop and re-create the DAS to reset its configuration file.

**Related reference:**

- "GET ADMIN CONFIGURATION" on page 374
- "RESET ADMIN CONFIGURATION" on page 635
- "Configuration parameters summary" in the *Administration Guide: Performance*

# UPDATE ALERT CONFIGURATION

Updates the alert configuration settings for health indicators.

**Authorization:**

One of the following:

- sysadm
- sysmaint
- sysctrl

**Required Connection:**

Instance. An explicit attachment is not required.

**Command Syntax:**

```
>>-UPDATE ALERT--+-CONFIGURATION-+--FOR--+-+-+-DATABASE MANAGER-+-+------------------------------------+--USING--health indicator name-->
                 +-CONFIG--------+       | | +-DB MANAGER-------+ |                                    |
                 '-CFG-----------'       | | '-DBM--------------' |                                    |
                                         | +-DATABASES----------+                                    |
                                         | +-CONTAINERS---------+                                    |
                                         | '-TABLESPACES--------'                                    |
                                         '-+-DATABASE----------------------------+--ON--database alias-'
                                           +-TABLESPACE--name--------------------+
                                           '-CONTAINER--name--FOR--tablespace-id-'
```

```
          .-,---------------------.
          V                       |
>--+-SET----parameter name--value-+----------------------------------------------------------------------------------------------------------+-><
   |              .-,----------------------------------------------------------------------------.                                           |
   |              |                                                .-,-------------------------. |                                           |
   |              V                                                V                           | |                                           |
   +-UPDATE ACTION-+-SCRIPT--pathname-+--ON--+-WARNING----------+---SET--parameter name--value-+-+-------------------------------------------+
   |               '-TASK--name-------'      +-ALARM------------+                                                                            |
   |                                         '-ATTENTION--state-'                                                                            |
   |              .-,---------------------------------------------.                                                                          |
   |              V                                               |                                                                          |
   +-DELETE ACTION-+-SCRIPT--pathname-+--ON--+-WARNING----------+-+--------------------------------------------------------------------------+
   |               '-TASK--name-------'      +-ALARM------------+                                                                            |
   |                                         '-ATTENTION--state-'                                                                            |
   |             .-,-----------------------------------------------------------------------------------------------------------------------. |
   |             V                                                                                                                         | |
   '-ADD ACTION----+-SCRIPT--pathname--<Add Script Details>-+--ON--+-WARNING----------+--+--------------+--USER--username--USING--password-+-'
                   '-TASK--name-----------------------------'      +-ALARM------------+  '-ON--hostname-'
                                                                   '-ATTENTION--state-'
```

**Add Script Details:**

```
|--TYPE--+-DB2--+------------------------------------------------+------+-------------------->
         |      '-+-STATEMENT TERMINATION CHARACTER-+--character-'      |
         |        +-STMT TERM CHAR------------------+                   |
         |        '-TERM CHAR-----------------------'                   |
         '-+-OPERATING SYSTEM-+--+------------------------------------+-'
           '-OS---------------'  '-+-COMMAND LINE PARAMETERS-+--parms-'
                                   '-PARMS-------------------'

>--WORKING DIRECTORY--pathname---------------------------------------------------------------|
```

**Command Parameters:**

**DATABASE MANAGER**
Updates alert settings for the database manager.

**DATABASES**
Updates alert settings for all databases managed by the database manager. These are the settings that apply to all databases that do not have custom settings. Custom settings are defined using the "DATABASE ON database alias" clause.

**CONTAINERS**
Updates alert settings for all table space containers managed by the database manager. These are the settings that apply to all table space containers that do not have custom settings. Custom settings are defined using the ″CONTAINER name ON database alias″ clause.

**TABLESPACES**
Updates alert settings for all table spaces managed by the database manager. These are the settings that apply to all table spaces that do not have custom settings. Custom settings are defined using the ″TABLESPACE name ON database alias″ clause.

**DATABASE ON** *database alias*
Updates the alert settings for the database specified using the ″ON database alias″ clause. If this database has custom settings, then they override the settings for all databases for the instance, which is specified using the DATABASES parameter.

**CONTAINER** *name* **FOR** *tablespace-id* **ON** *database alias*
Updates the alert settings for the table space container called *name*, for the table space specified using the ″FOR tablespace-id″ clause, on the database specified using the ″ON database alias″ clause. If this table space container has custom settings, then they override the settings for all table space containers for the database, which is specified using the CONTAINERS parameter.

**TABLESPACE** *name* **ON** *database alias*
Updates the alert settings for the table space called name, on the database specified using the ″ON database alias″ clause. If this table space has custom settings, then they override the settings for all table spaces for the database, which is specified using the TABLESPACES parameter.

**USING** *health indicator name*
Specifies the set of health indicators for which alert configuration will be updated. Health indicator names consist of a two-letter object identifier followed by a name which describes what the indicator measures. For example:

```
db.sort_privmem_util
```

**SET** *parameter-name value*
Updates the alert configuration element, parameter-name, of the health indicator to the specified value. parameter-name must be one of the following:

- ALARM
- WARNING
- SENSITIVITY
- ACTIONSENABLED
- THRESHOLDSCHECKED

**UPDATE ACTION SCRIPT** *pathname* **ON [WARNING | ALARM | ATTENTION state]**
Specifies that the script attributes of the predefined script with absolute pathname pathname will be updated according to the following clause:

**SET parameter-name value**
Updates the script attribute, parameter-name, to the specified value. parameter-name must be one of the following:

- SCRIPTTYPE
- WORKINGDIR

- TERMCHAR
- CMDLINEPARMS
- USERID
- PASSWORD
- SYSTEM

**UPDATE ACTION TASK** *name* **ON [WARNING | ALARM | ATTENTION state]**
Specifies that the task attributes of the task with name name will be updated according to the following clause:

**SET** *parameter-name value*
Updates the task attribute, parameter-name, to the specified value. *parameter-name* must be one of the following:

- USERID
- PASSWORD
- SYSTEM

**DELETE ACTION SCRIPT** *pathname* **ON [WARNING | ALARM | ATTENTION state]**
Removes the action script with absolute pathname pathname from the list of alert action scripts.

**DELETE ACTION TASK** *name* **ON [WARNING | ALARM | ATTENTION state]**
Removes the action task called name from the list of alert action tasks.

**ADD ACTION SCRIPT** *pathname* **ON [WARNING | ALARM | ATTENTION state]**
Specifies that a new action script with absolute pathname pathname is to be added, the attributes of which are given by the following:

**TYPE** An action script must be either a DB2 Command script or an operating system script:

- DB2
- OPERATING SYSTEM

If it is a DB2 Command script, then the following clause allows one to optionally specify the character, *character*, that is used in the script to terminate statements:

```
STATEMENT TERMINATION CHARACTER ;
```

If it is an operating system script, then the following clause allows one to optionally specify the command-line parameters, parms, that would be passed to the script upon invocation: COMMAND LINE PARAMETERS *parms*

**WORKING DIRECTORY** *pathname*
Specifies the absolute pathname, pathname, of the directory in which the script will be executed.

**USER** *username* **USING** *password*
Specifies the user account, username, and associated password, password, under which the script will be executed.

**ADD ACTION TASK** *name* **ON [WARNING | ALARM | ATTENTION** *state***]**
Specifies that a new task, called name, is to be added to be run ON the specified condition

**ON [WARNING | ALARM | ATTENTION** *state***]**
Specifies the condition on which the action will run. For threshold-based

HIs, this is WARNING or ALARM. For state-based HIs, this will be a numeric state as documented in a table to be provided for each state-based HI (for example, table space states).

# UPDATE ALTERNATE SERVER FOR DATABASE

Updates the alternate server for a database alias in the system database directory.

**Scope:**

This command only affects the database partition on which it is executed.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*

**Required connection:**

None.

**Command syntax:**

```
►►──UPDATE ALTERNATE SERVER FOR──┬─DATABASE─┬──database-alias──USING──────────────►
                                 └─DB───────┘

►─HOSTNAME──hostname──PORT──port-number─────────────────────────────────────────►◄
```

**Command parameters:**

**DATABASE** *database-alias*
: Specifies the alias of the database where the alternate server is to be updated.

**HOSTNAME** *hostname*
: Specifies a fully qualified host name or the IP address of the node where the alternate server for the database resides.

**PORT** *port-number*
: Specifies the port number of the alternate server of the database manager instance.

**Examples:**

The following example updates the alternate server for the SAMPLE database using host name montero and port 20396:

```
db2 update alternate server for database sample using hostname montero port 20396
```

The following two examples reset the alternate server for the SAMPLE database:

```
db2 update alternate server for database sample using hostname NULL port NULL
```

or

```
db2 update alternate server for database sample using hostname "" port NULL
```

**Usage notes:**
- This command is only applied to the system database directory.
- This command should only be used on a server instance. If it is issued on a client instance, it is ignored and message SQL1889W is returned.

- If Lightweight Directory Access Protocol (LDAP) support is enabled on the machine on which the command is issued, the alternate server for the database will be automatically registered in the LDAP directory.

**Related reference:**

- "CREATE DATABASE" on page 331

# UPDATE ALTERNATE SERVER FOR LDAP DATABASE

Updates the alternate server for a database in Lightweight Directory Access Protocol (LDAP).

**Authorization:**

Read/write access to the LDAP server.

**Required connection:**

None.

**Command syntax:**

```
►►──UPDATE ALTERNATE SERVER FOR LDAP──┬─DATABASE─┬──database-alias───────────────►
                                      └─DB───────┘

►─USING──┬─NODE──node───────┬──┬──────────────────────────────────────────┬─────►◄
         └─GWNODE──gwnode───┘  └─USER──username──┬──────────────────────┬─┘
                                                 └─PASSWORD──password───┘
```

**Command parameters:**

**DATABASE** *database-alias*
: Specifies the alias of the database to be updated.

**NODE** *node*
: Specifies the node name where the alternate server for the database resides.

**GWNODE** *gwnode*
: Specifies the node name where the alternate gateway for the database resides.

**USER** *username*
: Specifies the user's LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to create the object in the LDAP directory. If the user's LDAP DN is not specified, the credentials of the current logon user will be used.

If the user's LDAP DN and password have been specified using **db2ldcfg**, the user name and password do not have to be specified here.

**PASSWORD** *password*
: Account password.

If the user's LDAP DN and password have been specified using **db2ldcfg**, the user name and password do not have to be specified here.

**Related reference:**

- "CATALOG LDAP DATABASE" on page 313
- "db2ldcfg - Configure LDAP Environment" on page 120

# UPDATE CLI CONFIGURATION

Updates the contents of a specified section in the `db2cli.ini` file.

The `db2cli.ini` file is used as the DB2 call level interface (CLI) configuration file. It contains various keywords and values that can be used to modify the behavior of the DB2 CLI and the applications using it. The file is divided into sections, each section corresponding to a database alias name.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
►►──UPDATE CLI──┬─CONFIGURATION─┬──┬───────────────────────────┬──────────────────►
                ├─CONFIG────────┤  └─AT──┬─GLOBAL─┬──LEVEL─┘
                └─CFG───────────┘        └─USER───┘

                                   ┌──────────────┐
                                   ▼              │
►─FOR SECTION──section-name──USING───keyword value┴───────────────────────────►◄
```

**Command parameters:**

**FOR SECTION section-name**
Name of the section whose keywords are to be updated. If the specified section does not exist, a new section is created.

**AT GLOBAL LEVEL**
Specifies that the CLI configuration parameter is to be updated at the global level.

**Note:** This parameter is only applicable when LDAP support is enabled.

**AT USER LEVEL**
Specifies that the CLI configuration parameter is to be updated at the user level.

**Note:** If LDAP support is enabled, this setting will be consistent when logging on to different machines with the same LDAP user ID. If LDAP support is disabled, this setting will be consistent only when logging on to the same machine with the same operating system user ID.

**USING keyword value**
Specifies the CLI/ODBC parameter to be updated.

**Usage notes:**

The section name and the keywords specified on this command are not case sensitive. However, the keyword values *are* case sensitive.

If a keyword value is a string containing single quotation marks or imbedded blanks, the entire string must be delimited by double quotation marks. For example:

```
db2 update cli cfg for section tstcli1x
   using TableType "'TABLE','VIEW','SYSTEM TABLE'"
```

When the AT USER LEVEL keywords are specified, the CLI configuration parameters for the specified section are updated only for the current user; otherwise, they are updated for all users on the local machine. The CLI configuration at the user level is maintained in the LDAP directory and cached on the local machine. When reading the CLI configuration, DB2 always reads from the cache. The cache is refreshed when:

- The user updates the CLI configuration.
- The user explicitly forces a refresh of the CLI configuration using the REFRESH LDAP command.

In an LDAP environment, users can configure a set of default CLI settings for a database catalogued in the LDAP directory. When an LDAP cataloged database is added as a DSN (Data Source Name), either by using the CCA (Client Configuration Assistant) or the ODBC configuration utility, any default CLI settings, if they exist in the LDAP directory, will be configured for that DSN on the local machine. The AT GLOBAL LEVEL clause must be specified to configure a CLI parameter as a default setting.

**Related reference:**

- "GET CLI CONFIGURATION" on page 383
- "REFRESH LDAP" on page 612

## UPDATE COMMAND OPTIONS

Sets one or more command options during an interactive session, or from a batch input file. The settings revert to system defaults (or the system default overrides in **DB2OPTIONS**) when the interactive session or batch input file ends.

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
                                     ┌──────────────────────────────────┐
                                     ▼                                  │
►►──UPDATE COMMAND OPTIONS USING───option-letter──┬─ON──value─┬─┴─────────────►◄
                                                  └─OFF───────┘
```

**Command parameters:**

**USING option-letter**
The following option-letters can be set:

| | |
|---|---|
| **a** | Display SQLCA |
| **c** | Auto-commit SQL statements |
| **e** | Display SQLCODE/SQLSTATE |
| **l** | Log commands in a history file |
| **n** | Remove new line character |
| **o** | Display to standard output |
| **p** | Display DB2 interactive prompt |
| **r** | Save output report to a file |
| **s** | Stop execution on command error |
| **v** | Echo current command |
| **w** | Show SQL statement warning messages |
| **z** | Redirect all output to a file. |

**ON value**
The `e`, `l`, `r`, and `z` options require a value if they are turned on. For the `e` option, *value* can be `c` to display the SQLCODE, or `s` to display the SQLSTATE. For the `l`, `r`, and `z` options, *value* represents the name to be used for the history file or the report file. No other options accept a value.

**Usage notes:**

These settings override system defaults, settings in **DB2OPTIONS**, and options specified using the command line option flags.

The file input option (`-f`) and the statement termination option (`-t`) cannot be updated using this command.

To view the current option settings, use the LIST COMMAND OPTIONS command.

**Related reference:**

- "LIST COMMAND OPTIONS" on page 482

# UPDATE CONTACT

Updates the attributes of a contact that is defined on the local system. A contact is a user to whom the Scheduler and Health Monitor send messages. To create a contact, use the ADD CONTACT command.

**Authorization:**

None.

**Required connection:**

None.

**Command syntax:**

```
                               ┌─,─────────────┐
                               ▼               │
►►──UPDATE CONTACT──name──USING───keyword──value─┴──────────────────────────►◄
```

**Command parameters:**

**CONTACT name**
The name of the contact that will be updated.

**USING keyword value**
Specifies the contact parameter to be updated (*keyword*) and the value to which it will be set (*value*). The valid set of keywords is:

**ADDRESS**
The email address that is used by the SMTP server to send the notification.

**TYPE**
Whether the address is for an email address or a pager.

**MAXPAGELEN**
The maximum number of characters that the pager can accept.

**DESCRIPTION**
A textual description of the contact. This has a maximum length of 128 characters.

# UPDATE CONTACTGROUP

Updates the attributes of a contact group that is defined on the local system. A contact group is a list of users who should be notified by the Scheduler and the Health Monitor.

**Authorization:**

None

**Required Connection:**

None

**Command Syntax:**

```
►►──UPDATE CONTACTGROUP──name──────────────────────────────────────────────────►


     ┌─,────────────────────────────────────────────────────────┐
     ▼                                                          │
►─────┬─ADD──┬───┬─CONTACT─┬──name──┬──────────────────────────────────┬─┴──────────►◄
      └─DROP─┘   └─GROUP───┘        └─DESCRIPTION──new description─┘
```

**Command Parameters:**

**CONTACTGROUP** *name*
: Name of the contact group which will be updated.

**ADD CONTACT** *name*
: Specifies the name of the new contact to be added to the group. You do not need to define a contact with the ADD CONTACT command before you add it to a group.

**DROP CONTACT** *name*
: Specifies the name of a contact in the group that will be dropped from the group.

**ADD GROUP** *name*
: Specifies the name of the new contact group to be added to the group.

**DROP GROUP** *name*
: Specifies the name of a contact group that will be dropped from the group.

**DESCRIPTION** *new description*
: Optional. A new textual description for the contact group.

# UPDATE DATABASE CONFIGURATION

Modifies individual entries in a specific database configuration file.

A database configuration file resides on every node on which the database has been created.

**Scope:**

This command only affects the node on which it is executed.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*

**Required connection:**

Instance. An explicit attachment is not required, but a database connection is recommended when the database is active. If the database is listed as remote, an instance attachment to the remote node is established for the duration of the command. To change a parameter online, you must be connected to the database.

**Command syntax:**

```
►►──UPDATE──┬─DATABASE─┬──┬─CONFIGURATION─┬──┬───────────────────────┬──────────────►
            └─DB───────┘  ├─CONFIG────────┤  └─FOR──database-alias─┘
                          └─CFG───────────┘


           ┌────────────────────────┐  ┌─IMMEDIATE─┐
           ▼                        │  │           │
►─USING─────config-keyword value────┴──┼───────────┼──────────────────────────────►◄
                                       └─DEFERRED──┘
```

**Command parameters:**

**DEFERRED**
Make the changes only in the configuration file, so that the changes take effect the next time you reactivate the database.

**FOR database-alias**
Specifies the alias of the database whose configuration is to be updated. Specifying the database alias is not required when a database connection has already been established.

**IMMEDIATE**
Make the changes immediately, while the database is running. IMMEDIATE is the default action, but it requires a database connection to be effective.

**USING** *config-keyword value*
*config-keyword* specifies the database configuration parameter to be updated. *value* specifies the value to be assigned to the parameter.

**Usage notes:**

To view or print a list of the database configuration parameters, use the GET DATABASE CONFIGURATION command.

To reset all the database configuration parameters to the recommended defaults, use the RESET DATABASE CONFIGURATION command.

To change a database configuration parameter, use the UPDATE DATABASE CONFIGURATION command. For example, to change the logging mode to “archival logging” on a single-partition database environment containing a database called ZELLMART, use:

```
db2 update db cfg for zellmart using logretain recovery
```

To check that the *logretain* configuration parameter has changed, use:

```
db2 get db cfg for zellmart
```

When changing configuration parameters in a multiple-partitioned database environment, the **db2_all** command should be used. Using the **db2_all** command results in the update being issued against all partitions.

For example, to change the logging mode to “archival logging” in a multiple-partitioned database environment containing a database called “zellmart”, use:

```
db2_all ";db2 update db cfg for zellmart using logretain recovery"
```

To check that the *logretain* configuration parameter has changed on all database partitions, use:

```
db2_all ";db2 get db cfg for zellmart"
```

If you are working on a UNIX operating system, and you have the “grep” command, you can use the following command to view only the *logretain* values:

```
db2_all ";db2 get db cfg for zellmart | grep -i logretain"
```

For more information about DB2 configuration parameters and the values available for each type of database node, see the individual configuration parameter descriptions. The values of these parameters differ for each type of database node configured (server, client, or server with remote clients).

Not all parameters can be updated.

Some changes to the database configuration file become effective only after they are loaded into memory. All applications must disconnect from the database before this can occur. For more information on which parameters are configurable on-line and which ones are not, see summary list of configuration parameters.

For example, to change the *sortheap* database configuration parameter online for the SALES database, enter the following commands:

```
db2 connect to sales
db2 update db cfg using sortheap 1000
db2 connect reset
```

If an error occurs, the database configuration file does not change. The database configuration file cannot be updated if the checksum is invalid. This might occur if the database configuration file is changed without using the appropriate command. If this happens, the database must be restored to reset the database configuration file.

**Related concepts:**

- "rah and db2_all commands overview" in the *Administration Guide: Implementation*

**Related tasks:**

- "Configuring DB2 with configuration parameters" in the *Administration Guide: Performance*

**Related reference:**

- "GET DATABASE CONFIGURATION" on page 389
- "RESET DATABASE CONFIGURATION" on page 639
- "Configuration parameters summary" in the *Administration Guide: Performance*

# UPDATE DATABASE MANAGER CONFIGURATION

Modifies individual entries in the database manager configuration file.

**Authorization:**

*sysadm*

**Required connection:**

None or instance. An instance attachment is not required to perform local DBM configuration operations, but is required to perform remote DBM configuration operations. To update the database manager configuration for a remote instance, it is necessary to first attach to that instance. To update a configuration parameter on-line, it is also necessary to first attach to the instance.

**Command syntax:**

```
►►──UPDATE──┬─DATABASE MANAGER─┬──┬─CONFIGURATION─┬─────────────────────────►
            ├─DB MANAGER───────┤  ├─CONFIG────────┤
            └─DBM──────────────┘  └─CFG───────────┘


            ┌──────────────────────┐
            ▼                      │  ┌─IMMEDIATE─┐
►──USING────config-keyword value───┴──┼───────────┼──────────────────────►◄
                                      └─DEFERRED──┘
```

**Command parameters:**

**DEFERRED**
: Make the changes only in the configuration file, so that the changes take effect when the instance is restarted.

**IMMEDIATE**
: Make the changes right now, dynamically, while the instance is running. IMMEDIATE is the default, but it requires an instance attachment to be effective.

**USING config-keyword value**
: Specifies the database manager configuration parameter to be updated. For a list of configuration parameters, refer to the configuration parameters summary.

**Usage notes:**

To view or print a list of the database manager configuration parameters, use the GET DATABASE MANAGER CONFIGURATION command. To reset the database manager configuration parameters to the recommended database manager defaults, use the RESET DATABASE MANAGER CONFIGURATION command. For more information about database manager configuration parameters and the values of these parameters appropriate for each type of database node configured (server, client, or server with remote clients), see individual configuration parameter descriptions.

Not all parameters can be updated.

Some changes to the database manager configuration file become effective only after they are loaded into memory. For more information on which parameters are configurable on-line and which ones are not, see the configuration parameter summary. Server configuration parameters that are not reset immediately are reset during execution of **db2start**. For a client configuration parameter, parameters are reset the next time you restart the application. If the client is the command line processor, it is necessary to invoke TERMINATE.

For example, to change the *DIAGLEVEL* database manager configuration parameter on-line for the **eastern** instance of the database manager, enter the following command:

```
db2 attach to eastern
db2 update dbm cfg using DIAGLEVEL 1
db2 detach
```

If an error occurs, the database manager configuration file does not change.

The database manager configuration file cannot be updated if the checksum is invalid. This can occur if you edit database manager configuration file and do not use the appropriate command. If the checksum is invalid, you must reinstall the database manager to reset the database manager configuration file.

When you update the SVCENAME, NNAME, or TPNAME database manager configuration parameters for the current instance, if LDAP support is enabled and there is an LDAP server registered for this instance, the LDAP server is updated with the new value or values.

**Related tasks:**

- "Configuring DB2 with configuration parameters" in the *Administration Guide: Performance*

**Related reference:**

- "GET DATABASE MANAGER CONFIGURATION" on page 395
- "RESET DATABASE MANAGER CONFIGURATION" on page 641
- "TERMINATE" on page 705
- "Configuration parameters summary" in the *Administration Guide: Performance*

# UPDATE HEALTH NOTIFICATION CONTACT LIST

Updates the contact list for notification about health alerts issued by an instance.

**Authorization:**

One of the following:
- sysadm
- sysctrl
- sysmaint

**Required Connection:**

Instance. An explicit attachment is not required.

**Command Syntax:**

```
►►──UPDATE──┬─HEALTH NOTIFICATION CONTACT─┬──LIST──────────────────────────►
            └─NOTIFICATION────────────────┘


    ┌─,──────────────────────────┐
    ▼                            │
►──┬─ADD──┬──┬─CONTACT─┬──name───┴──────────────────────────────────────►◄
   └─DROP─┘  └─GROUP───┘
```

**Command Parameters:**

**ADD GROUP** *name*
: Add a new contact group that will notified of the health of the instance.

**ADD CONTACT** *name*
: Add a new contact that will notified of the health of the instance.

**DROP GROUP** *name*
: Removes the contact group from the list of contacts that will notified of the health of the instance.

**DROP CONTACT** *name*
: Removes the contact from the list of contacts that will notified of the health of the instance.

# UPDATE HISTORY FILE

Updates the location, device type, or comment in a history file entry.

**Authorization:**

One of the following:
- *sysadm*
- *sysctrl*
- *sysmaint*
- *dbadm*

**Required connection:**

Database

**Command syntax:**

```
►►──UPDATE HISTORY FOR──┬─object-part─┬──WITH──────────────────────────────────►
                        └─EID──eid────┘

►──┬─LOCATION──new-location──DEVICE TYPE──new-device-type─┬──────────────────►◄
   ├─COMMENT──new-comment─────────────────────────────────┤
   └─STATUS──new-status───────────────────────────────────┘
```

**Command parameters:**

**EID** *eid*
Specifies the history entry ID.

**FOR** *object-part*
Specifies the identifier for the backup or copy image. It is a time stamp with an optional sequence number from `001` to `999`.

**LOCATION** *new-location*
Specifies the new physical location of a backup image. The interpretation of this parameter depends on the device type.

**DEVICE TYPE** *new-device-type*
Specifies a new device type for storing the backup image. Valid device types are:

| | |
|---|---|
| **D** | Disk |
| **K** | Diskette |
| **T** | Tape |
| **A** | TSM |
| **U** | User exit |
| **P** | Pipe |
| **N** | Null device |
| **X** | XBSA |
| **Q** | SQL statement |
| **O** | Other |

**COMMENT** *new-comment*
Specifies a new comment to describe the entry.

**STATUS** *new-status*
Specifies a new status for an entry. Valid values are:

**A** Marks the entry as active.

**I** Marks the entry as inactive.

**Examples:**

To update the history file entry for a full database backup taken on April 13, 1997 at 10:00 a.m., enter:

```
db2 update history for 19970413100000001 with
   location /backup/dbbackup.1 device type d
```

**Usage notes:**

The primary purpose of the database history file is to record information, but the data contained in the history is used directly by automatic restore operations. During any restore where the AUTOMATIC option is specified, the history of backup images and their locations will be referenced and used by the restore utility to fulfill the automatic restore request. If the automatic restore function is to be used and backup images have been relocated since they were created, it is recommended that the database history record for those images be updated to reflect the current location. If the backup image location in the database history is not updated, automatic restore will not be able to locate the backup images, but manual restore commands can still be used successfully.

**Related reference:**

- "PRUNE HISTORY/LOGFILE" on page 584

# UPDATE LDAP NODE

Updates the protocol information associated with a node entry that represents the DB2 server in Lightweight Directory Access Protocol (LDAP).

**Authorization:**

None

**Required connection:**

None

**Command syntax:**

```
>>──UPDATE LDAP──NODE──nodename──┬─────────────────────────┬────────────────────────►
                                 └─HOSTNAME──hostname──┘

►──┬──────────────────────────┬──┬────────────────┬──┬────────────────────┬────────►
   └─SVCENAME──svcename─┘     └─NNAME──nname─┘    └─NETWORK──net_id─┘

►──┬──────────────────────────┬──┬────────────────┬──┬────────────────┬────────────►
   └─PARTNERLU──partner_lu─┘     └─TPNAME──tpname─┘  └─MODE──mode─┘

►──┬──────────────────────────┬──┬─────────────────────────────┬──────────────────►
   └─SECURITY──┬─NONE────┬─┘     └─LANADDRESS──lan_address─┘
               ├─SAME────┤
               └─PROGRAM─┘

►──┬─────────────────────────────────────┬──┬──────────────────────┬──────────────►
   └─CHGPWDLU──change_password_lu─┘      └─WITH──"comments"─┘

►──┬─────────────────────────────────────────────┬─────────────────────────────────►◄
   └─USER──username──┬──────────────────────┬─┘
                     └─PASSWORD──password─┘
```

**Command parameters:**

**NODE nodename**
: Specifies the node name when updating a remote DB2 server. The node name is the value specified when registering the DB2 server in LDAP.

**HOSTNAME hostname**
: Specifies the TCP/IP host name (or IP address).

**SVCENAME svcename**
: Specifies the TCP/IP service name or port number.

**NNAME nname**
: Specifies the NETBIOS workstation name.

**Note:** NETBIOS is only supported on Windows operating systems; however, this protocol can be updated from a work station using a non-Windows operating system.

**NETWORK net_id**
: Specifies the APPN network ID.

**PARTNERLU partner_lu**
: Specifies the APPN partner LU name for the DB2 server machine.

**TPNAME tpname**
Specifies the APPN transaction program name.

**MODE mode**
Specifies the APPN mode name.

**SECURITY**
Specifies the APPN security level. Valid values are:

**NONE**
Specifies that no security information is to be included in the allocation request sent to the server. This is the default security type for DB2 UDB server.

**SAME** Specifies that a user name is to be included in the allocation request sent to the server, together with an indicator that the user name has been ″already verified″. The server must be configured to accept ″already verified″ security.

**PROGRAM**
Specifies that both a user name and a password are to be included in the allocation request sent to the server. This is the default security type for host database servers such as DB2 for OS/390 or z/OS, DB2 for iSeries.

**LANADDRESS lan_address**
Specifies the APPN network adaptor address.

**CHGPWDLU change_password_lu**
Specifies the name of the partner LU that is to be used when changing the password for a host database server.

**WITH** ″**comments**″
Describes the DB2 server. Any comment that helps to describe the server registered in the network directory can be entered. Maximum length is 30 characters. A carriage return or a line feed character is not permitted. The comment text must be enclosed by double quotation marks.

**USER username**
Specifies the user's LDAP distinguished name (DN). The LDAP user DN must have sufficient authority to create and update the object in the LDAP directory. If the user's LDAP DN is not specified, the credentials of the current logon user will be used.

**PASSWORD password**
Account password.

**Related reference:**

- "REGISTER" on page 613
- "DEREGISTER" on page 344

# UPDATE MONITOR SWITCHES

Turns one or more database monitor recording switches on or off. When the database manager starts, the settings of the six switches are determined by the *dft_mon* database manager configuration parameter.

The database monitor records a base set of information at all times. Users who require more than this basic information can turn on the appropriate switches, but at a cost to system performance. The amount of information available in output from the GET SNAPSHOT command reflects which, if any, switches are on.

**Authorization:**

One of the following:

- *sysadm*
- *sysctrl*
- *sysmaint*
- *sysmon*

**Required connection:**

Instance or database:

- If there is neither an attachment to an instance, nor a connection to a database, a default instance attachment is created.
- If there is both an attachment to an instance, and a database connection, the instance attachment is used.

To update the monitor switches at a remote instance (or a different local instance), it is necessary to first attach to that instance.

**Command syntax:**

```
                                    ┌──────────────────────────┐
                                    ▼                          │
►►──UPDATE MONITOR SWITCHES USING───switch-name──┬─ON──┬───────┴─────────────►
                                                 └─OFF─┘

►──┬─────────────────────────────────────────┬──────────────────────────────►◄
   ├─AT DBPARTITIONNUM──db-partition-number─┤
   └─GLOBAL──────────────────────────────────┘
```

**Command parameters:**

**USING switch-name**
The following switch names are available:

| | |
|---|---|
| **BUFFERPOOL** | Buffer pool activity information |
| **LOCK** | Lock information |
| **SORT** | Sorting information |
| **STATEMENT** | SQL statement information |

| | |
|---|---|
| **TABLE** | Table activity information |
| **TIMESTAMP** | Monitoring timestamp information |
| **UOW** | Unit of work information. |

**AT DBPARTITIONNUM db-partition-number**
Specifies the database partition for which the status of the monitor switches is to be displayed.

**GLOBAL**
Returns an aggregate result for all database partitions in a partitioned database system.

**Usage notes:**

Information is collected by the database manager only after a switch is turned on. The switches remain set until **db2stop** is issued, or the application that issued the UPDATE MONITOR SWITCHES command terminates. To clear the information related to a particular switch, set the switch off, then on.

Updating switches in one application does not affect other applications.

To view the switch settings, use the GET MONITOR SWITCHES command.

**Compatibilities:**

For compatibility with versions earlier than Version 8:

- The keyword NODE can be substituted for DBPARTITIONNUM.

**Related reference:**

- "GET SNAPSHOT" on page 419
- "GET MONITOR SWITCHES" on page 410

# Chapter 4. Using command line SQL statements

This section provides information about using Structured Query Language (SQL) statements from the command line. These statements can be executed directly from an operating system command prompt, and can be used to define and manipulate information stored in a database table, index, or view in much the same way as if the commands were written into an application program. Information can be added, deleted, or updated, and reports can be generated from the contents of tables.

All SQL statements that can be executed through the command line processor are listed in the CLP column of Table 23 on page 746. The syntax of all the SQL statements, whether executed from the command line or embedded in a source program, is described in the SQL Reference. The syntax of many embedded SQL statements and CLP SQL statements is identical. However, host variables, parameter markers, descriptor names, and statement names are applicable only to embedded SQL. The syntax of CALL, CLOSE, CONNECT, DECLARE CURSOR, FETCH, and OPEN *does* depend on whether these statements are embedded or executed through the CLP. The CLP syntax of these statements is provided below:

**CALL**

```
►►──CALL──procedure-name──(──┬──────────────────────┬──)──────────────►◄
                             │ ┌─,──────────────┐   │
                             │ ▼                │   │
                             └───┬─expression─┬─┴───┘
                                 ├─?──────────┤
                                 └─null───────┘
```

**CLOSE**

```
►►──CLOSE──cursor-name───────────────────────────────────────────────►◄
```

**CONNECT**

```
►►──CONNECT──┬────────────────────────────────────────────────────┬──►◄
             ├─TO──server-name──┬────────────────┬──┬───────────────────┬─┤
             │                  └─┤ lock-block ├─┘  └─┤ authorization ├─┘ │
             ├─RESET──────────────────────────────────────────────────────┤
             │                  (1)                                       │
             └─┤ authorization ├──────────────────────────────────────────┘
```

### authorization:

```
├──USER──authorization-name──────────────────────────────────────────►
```

```
►─┬────────────────────────────────────────────────────┬──────────────────────────┤
  ├─USING─password─┬─────────────────────────────────┬─┤
  │                └─NEW─password─CONFIRM─password─┘ │
  └─CHANGE PASSWORD──────────────────────────────────┘
```

**lock-block:**

```
  ┌─IN SHARE MODE──────────────────────────┐
├─┼────────────────────────────────────────┼───────────────────────────────────┤
  └─IN EXCLUSIVE MODE─┬──────────────────┬─┘
                      └─ON SINGLE NODE─┘
```

**Notes:**

1 This form is only valid if implicit connect is enabled.

### DECLARE CURSOR

```
►►─DECLARE─cursor-name─CURSOR─┬───────────┬─FOR─select-statement──────────►◄
                              └─WITH HOLD─┘
```

### FETCH

```
►►─FETCH─┬──────┬─cursor-name─────────────────────────────────────────────►
         └─FROM─┘

►─┬──────────────────────────────────────────────────────────┬────────────►◄
  ├─FOR─┬─ALL─┬─┬─ROW──┬──────────────────────────────────────┤
  │     └─n───┘ └─ROWS─┘                                      │
  └─LOB─┬─COLUMN──┬─ALL─INTO─filename─┬─APPEND────┬───────────┘
        └─COLUMNS─┘                   ├─NEW───────┤
                                      └─OVERWRITE─┘
```

### OPEN

```
►►─OPEN─cursor-name───────────────────────────────────────────────────────►◄
```

**Notes:**

1. When CALL is issued:
   - An expression must be used for each IN or INOUT parameter of the procedure. For an INOUT parameter, the expression must be a single literal value.
   - A question mark (?) must be used for each OUT parameter of the procedure.
   - The stored procedure must be cataloged. If an uncataloged procedure is called, a SQL0440N error message is returned.
2. The CLP version of CONNECT permits the user to change the password, using the following parameters:

   **NEW password**

   Specifies the new password that is to be assigned to the user name. Passwords can be up to 18 characters in length. The system on which the password will be changed depends on how user authentication has been set up.

**CONFIRM password**
A string that must be identical to the new password. This parameter is used to catch entry errors.

**CHANGE PASSWORD**
If this option is specified, the user is prompted for the current password, a new password, and for confirmation of the new password. Passwords are not displayed at entry.

3. When FETCH is issued through the command line processor, decimal and floating-point numbers are displayed with the territory's decimal delimiter, that is, a period (.) in the U.S., Canada, and the U.K.; a comma (,) in most other countries. However, when INSERT, UPDATE, CALL, and other SQL statements are issued through the command line processor to update tables, a period must be used as the decimal delimiter, even in countries that use a comma for that purpose.
4. When FETCH is issued through the command line processor, null values are typically displayed as a hyphen (-). For databases configured with DFT_SQLMATHWARN YES, expressions that result in an arithmetic error are processed as null values. Such arithmetic error nulls are displayed as a plus (+).

   For example, create and populate table t1 as follows:

   ```
   create table t1 (i1 int , i2 int);
   insert into t1 values (1,1),(2,0),(3,null);
   ```

   The statement: `select i1/i2 from t1` generates the following result:

   ```
   1
   ---
     1
     +
     -
   3 records selected
   ```

5. A new LOB option has been added to FETCH. If the LOB clause is specified, only the next row is fetched:
   - Each LOB column value is fetched into a file with the name *filename.xxx*, where *filename* is specified in the LOB clause, and *xxx* is a file extension from `001` to `999` (`001` is the first LOB column in the select list of the corresponding DECLARE CURSOR statement, `002` is the second LOB column, and `999` is the 999th column). The maximum number of LOB columns that can be fetched into files is 999.
   - Names of the files containing the data are displayed in the LOB columns.
6. The command line processor displays BLOB columns in hexadecimal representation.
7. SQL statements that contain references to structured type columns cannot be issued if an appropriate transform function is not available.

To change the way that the CLP displays data (when querying databases using SQL statements through the CLP), rebind the CLP bind files against the database being queried. For example, to display date and time in ISO format, do the following:

1. Create a text file containing the names of the CLP bind files. This file is used as the list file for binding multiple files with one BIND command. In this example the file is named `clp.lst`, and its contents are:

```
db2clpcs.bnd +
db2clprr.bnd +
db2clpur.bnd +
db2clprs.bnd +
db2clpns.bnd
```

2. Connect to the database.
3. Issue the following command:

```
db2 bind @clp.lst collection nullid datetime iso
```

*Table 23. SQL Statements (DB2 Universal Database)*

| SQL Statement | Dynamic[1] | Command Line Processor (CLP) | Call Level Interface[3] (CLI) | SQL Procedure |
|---|---|---|---|---|
| ALLOCATE CURSOR | | | | X |
| assignment statement | | | | X |
| ASSOCIATE LOCATORS | | | | X |
| ALTER { BUFFERPOOL, NICKNAME,[9] NODEGROUP, SERVER,[9] TABLE, TABLESPACE, USER MAPPING,[9] TYPE, VIEW } | X | X | X | |
| BEGIN DECLARE SECTION[2] | | | | |
| CALL | X | X | X | X |
| CASE statement | | | | X |
| CLOSE | | X | `SQLCloseCursor()`, `SQLFreeStmt()` | X |
| COMMENT ON | X | X | X | X |
| COMMIT | X | X | `SQLEndTran()`, `SQLTransact()` | X |
| Compound SQL (Embedded) | | | X[4] | |
| compound statement | | | | X |
| CONNECT (Type 1) | | X | `SQLBrowseConnect()`, `SQLConnect()`, `SQLDriverConnect()` | |
| CONNECT (Type 2) | | X | `SQLBrowseConnect()`, `SQLConnect()`, `SQLDriverConnect()` | |
| CREATE { ALIAS, BUFFERPOOL, DISTINCT TYPE, EVENT MONITOR, FUNCTION, FUNCTION MAPPING,[9] INDEX, INDEX EXTENSION, METHOD, NICKNAME,[9] NODEGROUP, PROCEDURE, SCHEMA, SERVER, TABLE, TABLESPACE, TRANSFORM, TYPE MAPPING,[9] TRIGGER, USER MAPPING,[9] TYPE, VIEW, WRAPPER[9] } | X | X | X | X[10] |
| DECLARE CURSOR[2] | | X | `SQLAllocStmt()` | X |
| DECLARE GLOBAL TEMPORARY TABLE | X | X | X | X |
| DELETE | X | X | X | X |
| DESCRIBE[8] | | X | `SQLColAttributes()`, `SQLDescribeCol()`, `SQLDescribeParam()`[6] | |

*Table 23. SQL Statements (DB2 Universal Database) (continued)*

| SQL Statement | Dynamic[1] | Command Line Processor (CLP) | Call Level Interface[3] (CLI) | SQL Procedure |
|---|---|---|---|---|
| DISCONNECT | | X | `SQLDisconnect()` | |
| DROP | X | X | X | X[10] |
| END DECLARE SECTION[2] | | | | |
| EXECUTE | | | `SQLExecute()` | X |
| EXECUTE IMMEDIATE | | | `SQLExecDirect()` | X |
| EXPLAIN | X | X | X | X |
| FETCH | | X | `SQLExtendedFetch()`, `SQLFetch()`, `SQLFetchScroll()` | X |
| FLUSH EVENT MONITOR | X | X | X | |
| FOR statement | | | | X |
| FREE LOCATOR | | | X[4] | X |
| GET DIAGNOSTICS | | | | X |
| GOTO statement | | | | X |
| GRANT | X | X | X | X |
| IF statement | | | | X |
| INCLUDE[2] | | | | |
| INSERT | X | X | X | X |
| ITERATE | | | | X |
| LEAVE statement | | | | X |
| LOCK TABLE | X | X | X | X |
| LOOP statement | | | | X |
| OPEN | | X | `SQLExecute()`, `SQLExecDirect()` | X |
| PREPARE | | | `SQLPrepare()` | X |
| REFRESH TABLE | X | X | X | |
| RELEASE | | X | | X |
| RELEASE SAVEPOINT | X | X | X | X |
| RENAME TABLE | X | X | X | |
| RENAME TABLESPACE | X | X | X | |
| REPEAT statement | | | | X |
| RESIGNAL statement | | | | X |
| RETURN statement | | | | X |
| REVOKE | X | X | X | |
| ROLLBACK | X | X | `SQLEndTran()`, `SQLTransact()` | X |
| SAVEPOINT | X | X | X | X |
| select-statement | X | X | X | X |
| SELECT INTO | | | | X |
| SET CONNECTION | | X | `SQLSetConnection()` | |

*Table 23. SQL Statements (DB2 Universal Database) (continued)*

| SQL Statement | Dynamic[1] | Command Line Processor (CLP) | Call Level Interface[3] (CLI) | SQL Procedure |
|---|---|---|---|---|
| SET CURRENT DEFAULT TRANSFORM GROUP | X | X | X | X |
| SET CURRENT DEGREE | X | X | X | X |
| SET CURRENT EXPLAIN MODE | X | X | X, `SQLSetConnectAttr()` | X |
| SET CURRENT EXPLAIN SNAPSHOT | X | X | X, `SQLSetConnectAttr()` | X |
| SET CURRENT PACKAGESET | | | | |
| SET CURRENT QUERY OPTIMIZATION | X | X | X | X |
| SET CURRENT REFRESH AGE | X | X | X | X |
| SET EVENT MONITOR STATE | X | X | X | X |
| SET INTEGRITY | X | X | X | |
| SET PASSTHRU[9] | X | X | X | X |
| SET PATH | X | X | X | X |
| SET SCHEMA | X | X | X | X |
| SET SERVER OPTION[9] | X | X | X | X |
| SET transition-variable[5] | X | X | X | X |
| SIGNAL statement | | | | X |
| SIGNAL SQLSTATE[5] | X | X | X | |
| UPDATE | X | X | X | X |
| VALUES INTO | | | | X |
| WHENEVER[2] | | | | |
| WHILE statement | | | | X |

**Notes:**

1. You can code all statements in this list as static SQL, but only those marked with X as dynamic SQL.
2. You cannot execute this statement.
3. An X indicates that you can execute this statement using either `SQLExecDirect()` or `SQLPrepare()` and `SQLExecute()`. If there is an equivalent DB2 CLI function, the function name is listed.
4. Although this statement is not dynamic, with DB2 CLI you can specify this statement when calling either `SQLExecDirect()`, or `SQLPrepare()` and `SQLExecute()`.
5. You can only use this within CREATE TRIGGER statements.
6. You can only use the SQL DESCRIBE statement to describe output, whereas with DB2 CLI you can also describe input (using the `SQLDescribeParam()` function).
7. You can only use the SQL FETCH statement to fetch one row at a time in one direction, whereas with the DB2 CLI `SQLExtendedFetch()` and `SQLFetchScroll()` functions, you can fetch into arrays. Furthermore, you can fetch in any direction, and at any position in the result set.
8. The DESCRIBE SQL statement has a different syntax than that of the CLP DESCRIBE command.
9. Statement is supported only for federated database servers.
10. SQL procedures can only issue CREATE and DROP statements for indexes, tables, and views.

**Related reference:**

- "CLOSE statement" in the *SQL Reference, Volume 2*
- "CONNECT (Type 1) statement" in the *SQL Reference, Volume 2*
- "CONNECT (Type 2) statement" in the *SQL Reference, Volume 2*

- “DECLARE CURSOR statement” in the *SQL Reference, Volume 2*
- “FETCH statement” in the *SQL Reference, Volume 2*
- “OPEN statement” in the *SQL Reference, Volume 2*
- “SELECT statement” in the *SQL Reference, Volume 2*

# Appendix A. How to read the syntax diagrams

A syntax diagram shows how a command should be specified so that the operating system can correctly interpret what is typed.

Read a syntax diagram from left to right, and from top to bottom, following the horizontal line (the main path). If the line ends with an arrowhead, the command syntax is continued, and the next line starts with an arrowhead. A vertical bar marks the end of the command syntax.

When typing information from a syntax diagram, be sure to include punctuation, such as quotation marks and equal signs.

Parameters are classified as keywords or variables:

- Keywords represent constants, and are shown in uppercase letters; at the command prompt, however, keywords can be entered in upper, lower, or mixed case. A command name is an example of a keyword.
- Variables represent names or values that are supplied by the user, and are shown in lowercase letters; at the command prompt, however, variables can be entered in upper, lower, or mixed case, unless case restrictions are explicitly stated. A file name is an example of a variable.

A parameter can be a combination of a keyword and a variable.

Required parameters are displayed on the main path:

```
►►──COMMAND──required parameter──────────────────────────────────────►◄
```

Optional parameters are displayed below the main path:

```
►►──COMMAND──┬────────────────────┬──────────────────────────────────►◄
             └─optional parameter─┘
```

A parameter's default value is displayed above the path:

```
►►──COMMAND──┬──────────────────────┬─────────────────────────────────►◄
             │          ┌─VALUE1─┐  │
             └─OPTPARM──┼─VALUE2─┼──┘
                        ├─VALUE3─┤
                        └─VALUE4─┘
```

A stack of parameters, with the first parameter displayed on the main path, indicates that one of the parameters must be selected:

```
►►──COMMAND──┬─required choice1─┬─────────────────────────────────────►◄
             └─required choice2─┘
```

A stack of parameters, with the first parameter displayed below the main path, indicates that one of the parameters can be selected:

```
►►──COMMAND──┬──────────────────┬─────────────────────────────────────►◄
             ├─optional_choice1─┤
             └─optional_choice2─┘
```

An arrow returning to the left, above the path, indicates that items can be repeated in accordance with the following conventions:

- If the arrow is uninterrupted, the item can be repeated in a list with the items separated by blank spaces:

```
             ┌──────────────────────┐
             ▼                      │
►►──COMMAND────repeatable parameter─┴─────────────────────────────────►◄
```

- If the arrow contains a comma, the item can be repeated in a list with the items separated by commas:

```
             ┌─,────────────────────┐
             ▼                      │
►►──COMMAND────repeatable_parameter─┴─────────────────────────────────►◄
```

Items from parameter stacks can be repeated in accordance with the stack conventions for required and optional parameters discussed previously.

Some syntax diagrams contain parameter stacks within other parameter stacks. Items from stacks can only be repeated in accordance with the conventions discussed previously. That is, if an inner stack does not have a repeat arrow above it, but an outer stack does, only one parameter from the inner stack can be chosen and combined with any parameter from the outer stack, and that combination can be repeated. For example, the following diagram shows that one could combine parameter *choice2a* with parameter *choice2*, and then repeat that combination again (*choice2* plus *choice2a*):

```
              ┌──────────────────────────────────────────────┐
              ▼                                              │
►►──COMMAND────┬────────────────────────┬─parameter choice3──┴──────────────►◄
               ├─parameter choice1──────┤
               ├─parameter choice2──────┤
               │ ┌─parameter choice2a─┐ │
               └─┼─parameter choice2b─┼─┘
                 └─parameter choice2c─┘
```

Some commands are preceded by an optional path parameter:

```
►►──┬──────┬─COMMAND──────────────────────────────────────────────────────►◄
    └─path─┘
```

If this parameter is not supplied, the system searches the current directory for the command. If it cannot find the command, the system continues searching for the command in all the directories on the paths listed in the `.profile`.

Some commands have syntactical variants that are functionally equivalent:

```
►►──┬─COMMAND FORM1─┬─────────────────────────────────────────────────────►◄
    └─COMMAND FORM2─┘
```

# Appendix B. Naming Conventions

This section provides information about the conventions that apply when naming database manager objects, such as databases and tables, and authentication IDs.

- Character strings that represent names of database manager objects can contain any of the following: a-z, A-Z, 0-9, @, #, and $.
- ’User IDs and groups may also contain any of the following additional characters when supported by the security plug-in: _, !, %, (, ), {, }, –, ., ^.
- ’User IDs and groups containing any of the following characters must be delimited with quotations when entered through the command line processor: !, %, (, ), {, }, –, ., ^,
- The first character in the string must be an alphabetic character, @, #, or $; it cannot be a number or the letter sequences SYS, DBM, or IBM.
- Unless otherwise noted, names can be entered in lowercase letters; however, the database manager processes them as if they were uppercase.

  The exception to this is character strings that represent names under the systems network architecture (SNA). Many values, such as logical unit names (partner_lu and local_lu), are case sensitive. The name must be entered exactly as it appears in the SNA definitions that correspond to those terms.
- A database name or database alias is a unique character string containing from one to eight letters, numbers, or keyboard characters from the set described above.

  Databases are cataloged in the system and local database directories by their aliases in one field, and their original name in another. For most functions, the database manager uses the name entered in the alias field of the database directories. (The exceptions are CHANGE DATABASE COMMENT and CREATE DATABASE, where a directory path must be specified.)
- The name or the alias name of a table or a view is an SQL identifier that is a unique character string 1 to 128 characters in length. Column names can be 1 to 30 characters in length.

  A fully qualified table name consists of the *schema.tablename*. The schema is the unique user ID under which the table was created. The schema name for a declared temporary table must be SESSION.
- Authentication IDs cannot exceed 30 characters on Windows 32-bit operating systems and 8 characters on all other operating systems.
- Group IDs cannot exceed 30 characters in length.
- Local aliases for remote nodes that are to be cataloged in the node directory cannot exceed eight characters in length.

# Appendix C. DB2 Universal Database technical information

## DB2 documentation and help

DB2® technical information is available through the following tools and methods:

- DB2 Information Center
  - Topics
  - Help for DB2 tools
  - Sample programs
  - Tutorials
- Downloadable PDF files, PDF files on CD, and printed books
  - Guides
  - Reference manuals
- Command line help
  - Command help
  - Message help
  - SQL state help
- Installed source code
  - Sample programs

You can access additional DB2 Universal Database™ technical information such as technotes, white papers, and Redbooks™ online at ibm.com®. Access the DB2 Information Management software library site at www.ibm.com/software/data/pubs/.

### DB2 documentation updates

IBM® may periodically make documentation FixPaks and other documentation updates to the DB2 Information Center available. If you access the DB2 Information Center at http://publib.boulder.ibm.com/infocenter/db2help/, you will always be viewing the most up-to-date information. If you have installed the DB2 Information Center locally, then you need to install any updates manually before you can view them. Documentation updates allow you to update the information that you installed from the *DB2 Information Center CD* when new information becomes available.

The Information Center is updated more frequently than either the PDF or the hardcopy books. To get the most current DB2 technical information, install the documentation updates as they become available or go to the DB2 Information Center at the www.ibm.com site.

**Related concepts:**

- “CLI sample programs” in the *CLI Guide and Reference, Volume 1*
- “Java sample programs” in the *Application Development Guide: Building and Running Applications*
- “DB2 Information Center” on page 758

**Related tasks:**

- “Invoking contextual help from a DB2 tool” on page 775

- “Updating the DB2 Information Center installed on your computer or intranet server” on page 767
- “Invoking message help from the command line processor” on page 776
- “Invoking command help from the command line processor” on page 776
- “Invoking SQL state help from the command line processor” on page 777

**Related reference:**

- “DB2 PDF and printed documentation” on page 769

## DB2 Information Center

The DB2® Information Center gives you access to all of the information you need to take full advantage of DB2 family products, including DB2 Universal Database™, DB2 Connect™, DB2 Information Integrator and DB2 Query Patroller™. The DB2 Information Center also contains information for major DB2 features and components including replication, data warehousing, and the DB2 extenders.

The DB2 Information Center has the following features if you view it in Mozilla 1.0 or later or Microsoft® Internet Explorer 5.5 or later. Some features require you to enable support for JavaScript™:

**Flexible installation options**

You can choose to view the DB2 documentation using the option that best meets your needs:

- To effortlessly ensure that your documentation is always up to date, you can access all of your documentation directly from the DB2 Information Center hosted on the IBM® Web site at http://publib.boulder.ibm.com/infocenter/db2help/
- To minimize your update efforts and keep your network traffic within your intranet, you can install the DB2 documentation on a single server on your intranet
- To maximize your flexibility and reduce your dependence on network connections, you can install the DB2 documentation on your own computer

**Search**

You can search all of the topics in the DB2 Information Center by entering a search term in the **Search** text field. You can retrieve exact matches by enclosing terms in quotation marks, and you can refine your search with wildcard operators (*, ?) and Boolean operators (AND, NOT, OR).

**Task-oriented table of contents**

You can locate topics in the DB2 documentation from a single table of contents. The table of contents is organized primarily by the kind of tasks you may want to perform, but also includes entries for product overviews, goals, reference information, an index, and a glossary.

- Product overviews describe the relationship between the available products in the DB2 family, the features offered by each of those products, and up to date release information for each of these products.
- Goal categories such as installing, administering, and developing include topics that enable you to quickly complete tasks and develop a deeper understanding of the background information for completing those tasks.

- Reference topics provide detailed information about a subject, including statement and command syntax, message help, and configuration parameters.

**Show current topic in table of contents**
You can show where the current topic fits into the table of contents by clicking the **Refresh / Show Current Topic** button in the table of contents frame or by clicking the **Show in Table of Contents** button in the content frame. This feature is helpful if you have followed several links to related topics in several files or arrived at a topic from search results.

**Index** You can access all of the documentation from the index. The index is organized in alphabetical order by index term.

**Glossary**
You can use the glossary to look up definitions of terms used in the DB2 documentation. The glossary is organized in alphabetical order by glossary term.

**Integrated localized information**
The DB2 Information Center displays information in the preferred language set in your browser preferences. If a topic is not available in your preferred language, the DB2 Information Center displays the English version of that topic.

For iSeries™ technical information, refer to the IBM eServer™ iSeries information center at www.ibm.com/eserver/iseries/infocenter/.

**Related concepts:**

- "DB2 Information Center installation scenarios" on page 759

**Related tasks:**

- "Updating the DB2 Information Center installed on your computer or intranet server" on page 767
- "Displaying topics in your preferred language in the DB2 Information Center" on page 768
- "Invoking the DB2 Information Center" on page 766
- "Installing the DB2 Information Center using the DB2 Setup wizard (UNIX)" on page 762
- "Installing the DB2 Information Center using the DB2 Setup wizard (Windows)" on page 764

## DB2 Information Center installation scenarios

Different working environments can pose different requirements for how to access DB2® information. The DB2 Information Center can be accessed on the IBM® Web site, on a server on your organization's network, or on a version installed on your computer. In all three cases, the documentation is contained in the DB2 Information Center, which is an architected web of topic-based information that you view with a browser. By default, DB2 products access the DB2 Information Center on the IBM Web site. However, if you want to access the DB2 Information Center on an intranet server or on your own computer, you must install the DB2 Information Center using the DB2 Information Center CD found in your product Media Pack. Refer to the summary of options for accessing DB2 documentation which follows, along with the three installation scenarios, to help determine which method of accessing the DB2 Information Center works best for you and your work environment, and what installation issues you might need to consider.

**Summary of options for accessing DB2 documentation:**

The following table provides recommendations on which options are possible in your work environment for accessing the DB2 product documentation in the DB2 Information Center.

| Internet access | Intranet access | Recommendation |
|---|---|---|
| Yes | Yes | Access the DB2 Information Center on the IBM Web site, or access the DB2 Information Center installed on an intranet server. |
| Yes | No | Access the DB2 Information Center on the IBM Web site. |
| No | Yes | Access the DB2 Information Center installed on an intranet server. |
| No | No | Access the DB2 Information Center on a local computer. |

**Scenario: Accessing the DB2 Information Center on your computer:**

Tsu-Chen owns a factory in a small town that does not have a local ISP to provide him with Internet access. He purchased DB2 Universal Database™ to manage his inventory, his product orders, his banking account information, and his business expenses. Never having used a DB2 product before, Tsu-Chen needs to learn how to do so from the DB2 product documentation.

After installing DB2 Universal Database on his computer using the typical installation option, Tsu-Chen tries to access the DB2 documentation. However, his browser gives him an error message that the page he tried to open cannot be found. Tsu-Chen checks the installation manual for his DB2 product and discovers that he has to install the DB2 Information Center if he wants to access DB2 documentation on his computer. He finds the *DB2 Information Center CD* in the media pack and installs it.

From the application launcher for his operating system, Tsu-Chen now has access to the DB2 Information Center and can learn how to use his DB2 product to increase the success of his business.

**Scenario: Accessing the DB2 Information Center on the IBM Web site:**

Colin is an information technology consultant with a training firm. He specializes in database technology and SQL and gives seminars on these subjects to businesses all over North America using DB2 Universal Database. Part of Colin’s seminars includes using DB2 documentation as a teaching tool. For example, while teaching courses on SQL, Colin uses the DB2 documentation on SQL as a way to teach basic and advanced syntax for database queries.

Most of the businesses at which Colin teaches have Internet access. This situation influenced Colin’s decision to configure his mobile computer to access the DB2 Information Center on the IBM Web site when he installed the latest version of DB2 Universal Database. This configuration allows Colin to have online access to the latest DB2 documentation during his seminars.

However, sometimes while travelling Colin does not have Internet access. This posed a problem for him, especially when he needed to access to DB2 documentation to prepare for seminars. To avoid situations like this, Colin installed a copy of the DB2 Information Center on his mobile computer.

Colin enjoys the flexibility of always having a copy of DB2 documentation at his disposal. Using the **db2set** command, he can easily configure the registry variables on his mobile computer to access the DB2 Information Center on either the IBM Web site, or his mobile computer, depending on his situation.

**Scenario: Accessing the DB2 Information Center on an intranet server:**

Eva works as a senior database administrator for a life insurance company. Her administration responsibilities include installing and configuring the latest version of DB2 Universal Database on the company's UNIX® database servers. Her company recently informed its employees that, for security reasons, it would not provide them with Internet access at work. Because her company has a networked environment, Eva decides to install a copy of the DB2 Information Center on an intranet server so that all employees in the company who use the company's data warehouse on a regular basis (sales representatives, sales managers, and business analysts) have access to DB2 documentation.

Eva instructs her database team to install the latest version of DB2 Universal Database on all of the employee's computers using a response file, to ensure that each computer is configured to access the DB2 Information Center using the host name and the port number of the intranet server.

However, through a misunderstanding Migual, a junior database administrator on Eva's team, installs a copy of the DB2 Information Center on several of the employee computers, rather than configuring DB2 Universal Database to access the DB2 Information Center on the intranet server. To correct this situation Eva tells Migual to use the **db2set** command to change the DB2 Information Center registry variables (DB2_DOCHOST for the host name, and DB2_DOCPORT for the port number) on each of these computers. Now all of the appropriate computers on the network have access to the DB2 Information Center, and employees can find answers to their DB2 questions in the DB2 documentation.

**Related concepts:**

- "DB2 Information Center" on page 758

**Related tasks:**

- "Updating the DB2 Information Center installed on your computer or intranet server" on page 767
- "Installing the DB2 Information Center using the DB2 Setup wizard (UNIX)" on page 762
- "Installing the DB2 Information Center using the DB2 Setup wizard (Windows)" on page 764
- "Setting the location for accessing the DB2 Information Center: Common GUI help"

**Related reference:**

- "db2set - DB2 Profile Registry" on page 203

## Installing the DB2 Information Center using the DB2 Setup wizard (UNIX)

DB2 product documentation can be accessed in three ways: on the IBM Web site, on an intranet server, or on a version installed on your computer. By default, DB2 products access DB2 documentation on the IBM Web site. If you want to access the DB2 documentation on an intranet server or on your own computer, you must install the documentation from the *DB2 Information Center CD*. Using the DB2 Setup wizard, you can define your installation preferences and install the DB2 Information Center on a computer that uses a UNIX operating system.

**Prerequisites:**

This section lists the hardware, operating system, software, and communication requirements for installing the DB2 Information Center on UNIX computers.

- **Hardware requirements**

  You require one of the following processors:
  - PowerPC (AIX)
  - HP 9000 (HP-UX)
  - Intel 32–bit (Linux)
  - Solaris UltraSPARC computers (Solaris Operating Environment)
- **Operating system requirements**

  You require one of the following operating systems:
  - IBM AIX 5.1 (on PowerPC)
  - HP-UX 11i (on HP 9000)
  - Red Hat Linux 8.0 (on Intel 32–bit)
  - SuSE Linux 8.1 (on Intel 32–bit)
  - Sun Solaris Version 8 (on Solaris Operating Environment UltraSPARC computers)

  **Note:** The DB2 Information Center runs on a subset of the UNIX operating systems on which DB2 clients are supported. It is therefore recommended that you either access the DB2 Information Center from the IBM Web site, or that you install and access the DB2 Information Center on an intranet server.
- **Software requirements**
  - The following browser is supported:
    - Mozilla Version 1.0 or greater
- The DB2 Setup wizard is a graphical installer. You must have an implementation of the X Window System software capable of rendering a graphical user interface for the DB2 Setup wizard to run on your computer. Before you can run the DB2 Setup wizard you must ensure that you have properly exported your display. For example, enter the following command at the command prompt:

  ```
  export DISPLAY=9.26.163.144:0.
  ```
- **Communication requirements**
  - TCP/IP

**Procedure:**

To install the DB2 Information Center using the DB2 Setup wizard:

1. Log on to the system.
2. Insert and mount the DB2 Information Center product CD on your system.
3. Change to the directory where the CD is mounted by entering the following command:

   ```
   cd /cd
   ```

   where */cd* represents the mount point of the CD.
4. Enter the **./db2setup** command to start the DB2 Setup wizard.
5. The IBM DB2 Setup Launchpad opens. To proceed directly to the installation of the DB2 Information Center, click **Install Product**. Online help is available to guide you through the remaining steps. To invoke the online help, click **Help**. You can click **Cancel** at any time to end the installation.
6. On the **Select the product you would like to install** page, click **Next**.
7. Click **Next** on the **Welcome to the DB2 Setup wizard** page. The DB2 Setup wizard will guide you through the program setup process.
8. To proceed with the installation, you must accept the license agreement. On the **License Agreement** page, select **I accept the terms in the license agreement** and click **Next**.
9. Select **Install DB2 Information Center on this computer** on the **Select the installation action** page. If you want to use a response file to install the DB2 Information Center on this or other computers at a later time, select **Save your settings in a response file**. Click **Next**.
10. Select the languages in which the DB2 Information Center will be installed on **Select the languages to install** page. Click **Next**.
11. Configure the DB2 Information Center for incoming communication on the **Specify the DB2 Information Center port** page. Click **Next** to continue the installation.
12. Review the installation choices you have made in the **Start copying files** page. To change any settings, click **Back**. Click **Install** to copy the DB2 Information Center files onto your computer.

You can also install the DB2 Information Center using a response file.

The installation logs `db2setup.his`, `db2setup.log`, and `db2setup.err` are located, by default, in the `/tmp` directory.

The `db2setup.log` file captures all DB2 product installation information, including errors. The `db2setup.his` file records all DB2 product installations on your computer. DB2 appends the `db2setup.log` file to the `db2setup.his` file. The `db2setup.err` file captures any error output that is returned by Java, for example, exceptions and trap information.

When the installation is complete, the DB2 Information Center will be installed in one of the following directories, depending upon your UNIX operating system:

- AIX: /usr/opt/db2_08_01
- HP-UX: /opt/IBM/db2/V8.1
- Linux: /opt/IBM/db2/V8.1
- Solaris Operating Environment: /opt/IBM/db2/V8.1

**Related concepts:**

- “DB2 Information Center” on page 758
- “DB2 Information Center installation scenarios” on page 759

**Related tasks:**

- "Installing DB2 using a response file (UNIX)" in the *Installation and Configuration Supplement*
- "Updating the DB2 Information Center installed on your computer or intranet server" on page 767
- "Displaying topics in your preferred language in the DB2 Information Center" on page 768
- "Invoking the DB2 Information Center" on page 766
- "Installing the DB2 Information Center using the DB2 Setup wizard (Windows)" on page 764

## Installing the DB2 Information Center using the DB2 Setup wizard (Windows)

DB2 product documentation can be accessed in three ways: on the IBM Web site, on an intranet server, or on a version installed on your computer. By default, DB2 products access DB2 documentation on the IBM Web site. If you want to access the DB2 documentation on an intranet server or on your own computer, you must install the DB2 documentation from the *DB2 Information Center CD*. Using the DB2 Setup wizard, you can define your installation preferences and install the DB2 Information Center on a computer that uses a Windows operating system.

**Prerequisites:**

This section lists the hardware, operating system, software, and communication requirements for installing the DB2 Information Center on Windows.

- **Hardware requirements**

  You require one of the following processors:
  - 32-bit computers: a Pentium or Pentium compatible CPU
- **Operating system requirements**

  You require one of the following operating systems:
  - Windows 2000
  - Windows XP

  **Note:** The DB2 Information Center runs on a subset of the Windows operating systems on which DB2 clients are supported. It is therefore recommended that you either access the DB2 Information Center on the IBM Web site, or that you install and access the DB2 Information Center on an intranet server.
- **Software requirements**
  - The following browsers are supported:
    - Mozilla 1.0 or greater
    - Internet Explorer Version 5.5 or 6.0 (Version 6.0 for Windows XP)
- **Communication requirements**
  - TCP/IP

**Restrictions:**

- You require an account with administrative privileges to install the DB2 Information Center.

**Procedure:**

To install the DB2 Information Center using the DB2 Setup wizard:

1. Log on to the system with the account that you have defined for the DB2 Information Center installation.
2. Insert the CD into the drive. If enabled, the auto-run feature starts the IBM DB2 Setup Launchpad.
3. The DB2 Setup wizard determines the system language and launches the setup program for that language. If you want to run the setup program in a language other than English, or the setup program fails to auto-start, you can start the DB2 Setup wizard manually.

   To start the DB2 Setup wizard manually:

   a. Click **Start** and select **Run**.

   b. In the **Open** field, type the following command:

      ```
      x:\setup.exe /i 2-letter language identifier
      ```

      where *x:* represents your CD drive, and *2-letter language identifier* represents the language in which the setup program will be run.

   c. Click **OK**.
4. The IBM DB2 Setup Launchpad opens. To proceed directly to the installation of the DB2 Information Center, click **Install Product**. Online help is available to guide you through the remaining steps. To invoke the online help, click **Help**. You can click **Cancel** at any time to end the installation.
5. On the **Select the product you would like to install** page, click **Next**.
6. Click **Next** on the **Welcome to the DB2 Setup wizard** page. The DB2 Setup wizard will guide you through the program setup process.
7. To proceed with the installation, you must accept the license agreement. On the **License Agreement** page, select **I accept the terms in the license agreement** and click **Next**.
8. Select **Install DB2 Information Center on this computer** on the **Select the installation action** page. If you want to use a response file to install the DB2 Information Center on this or other computers at a later time, select **Save your settings in a response file**. Click **Next**.
9. Select the languages in which the DB2 Information Center will be installed on **Select the languages to install** page. Click **Next**.
10. Configure the DB2 Information Center for incoming communication on the **Specify the DB2 Information Center port** page. Click **Next** to continue the installation.
11. Review the installation choices you have made in the **Start copying files** page. To change any settings, click **Back**. Click **Install** to copy the DB2 Information Center files onto your computer.

You can install the DB2 Information Center using a response file. You can also use the **db2rspgn** command to generate a response file based on an existing installation.

For information on errors encountered during installation, see the `db2.log` and `db2wi.log` files located in the `'My Documents'\DB2LOG\` directory. The location of the `'My Documents'` directory will depend on the settings on your computer.

The `db2wi.log` file captures the most recent DB2 installation information. The `db2.log` captures the history of DB2 product installations.

**Related concepts:**

- "DB2 Information Center" on page 758
- "DB2 Information Center installation scenarios" on page 759

**Related tasks:**

- "Installing a DB2 product using a response file (Windows)" in the *Installation and Configuration Supplement*
- "Updating the DB2 Information Center installed on your computer or intranet server" on page 767
- "Displaying topics in your preferred language in the DB2 Information Center" on page 768
- "Invoking the DB2 Information Center" on page 766
- "Installing the DB2 Information Center using the DB2 Setup wizard (UNIX)" on page 762

**Related reference:**

- "db2rspgn - Response File Generator (Windows)" on page 200

## Invoking the DB2 Information Center

The DB2 Information Center gives you access to all of the information that you need to use DB2 products for Linux, UNIX, and Windows operating systems such as DB2 Universal Database, DB2 Connect, DB2 Information Integrator, and DB2 Query Patroller.

You can invoke the DB2 Information Center from one of the following places:

- Computers on which a DB2 UDB client or server is installed
- An intranet server or local computer on which the DB2 Information Center installed
- The IBM Web site

**Prerequisites:**

Before you invoke the DB2 Information Center:

- *Optional*: Configure your browser to display topics in your preferred language
- *Optional*: Configure your DB2 client to use the DB2 Information Center installed on your computer or intranet server

**Procedure:**

To invoke the DB2 Information Center on a computer on which a DB2 UDB client or server is installed:

- From the Start Menu (Windows operating system): Click **Start** ⟶ **Programs** ⟶ **IBM DB2** ⟶ **Information** ⟶ **Information Center**.
- From the command line prompt:
  - For Linux and UNIX operating systems, issue the **db2icdocs** command.
  - For the Windows operating system, issue the **db2icdocs.exe** command.

To open the DB2 Information Center installed on an intranet server or local computer in a Web browser:

- Open the Web page at http://<host-name>:<port-number>/, where <host-name> represents the host name and <port-number> represents the port number on which the DB2 Information Center is available.

To open the DB2 Information Center on the IBM Web site in a Web browser:
- Open the Web page at publib.boulder.ibm.com/infocenter/db2help/.

**Related concepts:**
- "DB2 Information Center" on page 758
- "DB2 Information Center installation scenarios" on page 759

**Related tasks:**
- "Invoking contextual help from a DB2 tool" on page 775
- "Updating the DB2 Information Center installed on your computer or intranet server" on page 767
- "Invoking command help from the command line processor" on page 776
- "Setting the location for accessing the DB2 Information Center: Common GUI help"

**Related reference:**
- "HELP" on page 447

## Updating the DB2 Information Center installed on your computer or intranet server

The DB2 Information Center available from http://publib.boulder.ibm.com/infocenter/db2help/ will be periodically updated with new or changed documentation. IBM may also make DB2 Information Center updates available to download and install on your computer or intranet server. Updating the DB2 Information Center does not update DB2 client or server products.

**Prerequisites:**

You must have access to a computer that is connected to the Internet.

**Procedure:**

To update the DB2 Information Center installed on your computer or intranet server:

1. Open the DB2 Information Center hosted on the IBM Web site at: http://publib.boulder.ibm.com/infocenter/db2help/
2. In the Downloads section of the welcome page under the Service and Support heading, click the **DB2 Universal Database documentation** link.
3. Determine if the version of your DB2 Information Center is out of date by comparing the latest refreshed documentation image level to the documentation level you have installed. The documentation level you have installed is listed on the DB2 Information Center welcome page.
4. If a more recent version of the DB2 Information Center is available, download the latest refreshed *DB2 Information Center* image applicable to your operating system.

5. To install the refreshed *DB2 Information Center* image, follow the instructions provided on the Web page.

**Related concepts:**

- "DB2 Information Center installation scenarios" on page 759

**Related tasks:**

- "Invoking the DB2 Information Center" on page 766
- "Installing the DB2 Information Center using the DB2 Setup wizard (UNIX)" on page 762
- "Installing the DB2 Information Center using the DB2 Setup wizard (Windows)" on page 764

## Displaying topics in your preferred language in the DB2 Information Center

The DB2 Information Center attempts to display topics in the language specified in your browser preferences. If a topic has not been translated into your preferred language, the DB2 Information Center displays the topic in English.

**Procedure:**

To display topics in your preferred language in the Internet Explorer browser:

1. In Internet Explorer, click the **Tools** —> **Internet Options** —> **Languages...** button. The Language Preferences window opens.
2. Ensure your preferred language is specified as the first entry in the list of languages.
   - To add a new language to the list, click the **Add...** button.

     **Note:** Adding a language does not guarantee that the computer has the fonts required to display the topics in the preferred language.
   - To move a language to the top of the list, select the language and click the **Move Up** button until the language is first in the list of languages.
3. Refresh the page to display the DB2 Information Center in your preferred language.

To display topics in your preferred language in the Mozilla browser:

1. In Mozilla, select the **Edit** —> **Preferences** —> **Languages** button. The Languages panel is displayed in the Preferences window.
2. Ensure your preferred language is specified as the first entry in the list of languages.
   - To add a new language to the list, click the **Add...** button to select a language from the Add Languages window.
   - To move a language to the top of the list, select the language and click the **Move Up** button until the language is first in the list of languages.
3. Refresh the page to display the DB2 Information Center in your preferred language.

**Related concepts:**

- "DB2 Information Center" on page 758

## DB2 PDF and printed documentation

The following tables provide official book names, form numbers, and PDF file names. To order hardcopy books, you must know the official book name. To print a PDF file, you must know the PDF file name.

The DB2 documentation is categorized by the following headings:

- Core DB2 information
- Administration information
- Application development information
- Business intelligence information
- DB2 Connect information
- Getting started information
- Tutorial information
- Optional component information
- Release notes

The following tables describe, for each book in the DB2 library, the information needed to order the hard copy, or to print or view the PDF for that book. A full description of each of the books in the DB2 library is available from the IBM Publications Center at www.ibm.com/shop/publications/order

### Core DB2 information

The information in these books is fundamental to all DB2 users; you will find this information useful whether you are a programmer, a database administrator, or someone who works with DB2 Connect, DB2 Warehouse Manager, or other DB2 products.

*Table 24. Core DB2 information*

| Name | Form Number | PDF File Name |
|---|---|---|
| *IBM DB2 Universal Database Command Reference* | SC09-4828 | db2n0x81 |
| *IBM DB2 Universal Database Glossary* | No form number | db2t0x81 |
| *IBM DB2 Universal Database Message Reference, Volume 1* | GC09-4840, not available in hardcopy | db2m1x81 |
| *IBM DB2 Universal Database Message Reference, Volume 2* | GC09-4841, not available in hardcopy | db2m2x81 |
| *IBM DB2 Universal Database What's New* | SC09-4848 | db2q0x81 |

### Administration information

The information in these books covers those topics required to effectively design, implement, and maintain DB2 databases, data warehouses, and federated systems.

*Table 25. Administration information*

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Universal Database Administration Guide: Planning* | SC09-4822 | db2d1x81 |

Table 25. Administration information (continued)

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Universal Database Administration Guide: Implementation* | SC09-4820 | db2d2x81 |
| *IBM DB2 Universal Database Administration Guide: Performance* | SC09-4821 | db2d3x81 |
| *IBM DB2 Universal Database Administrative API Reference* | SC09-4824 | db2b0x81 |
| *IBM DB2 Universal Database Data Movement Utilities Guide and Reference* | SC09-4830 | db2dmx81 |
| *IBM DB2 Universal Database Data Recovery and High Availability Guide and Reference* | SC09-4831 | db2hax81 |
| *IBM DB2 Universal Database Data Warehouse Center Administration Guide* | SC27-1123 | db2ddx81 |
| *IBM DB2 Universal Database SQL Reference, Volume 1* | SC09-4844 | db2s1x81 |
| *IBM DB2 Universal Database SQL Reference, Volume 2* | SC09-4845 | db2s2x81 |
| *IBM DB2 Universal Database System Monitor Guide and Reference* | SC09-4847 | db2f0x81 |

## Application development information

The information in these books is of special interest to application developers or programmers working with DB2 Universal Database (DB2 UDB). You will find information about supported languages and compilers, as well as the documentation required to access DB2 UDB using the various supported programming interfaces, such as embedded SQL, ODBC, JDBC, SQLJ, and CLI. If you are using the DB2 Information Center, you can also access HTML versions of the source code for the sample programs.

Table 26. Application development information

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Universal Database Application Development Guide: Building and Running Applications* | SC09-4825 | db2axx81 |
| *IBM DB2 Universal Database Application Development Guide: Programming Client Applications* | SC09-4826 | db2a1x81 |
| *IBM DB2 Universal Database Application Development Guide: Programming Server Applications* | SC09-4827 | db2a2x81 |
| *IBM DB2 Universal Database Call Level Interface Guide and Reference, Volume 1* | SC09-4849 | db2l1x81 |

Table 26. Application development information (continued)

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Universal Database Call Level Interface Guide and Reference, Volume 2* | SC09-4850 | db2l2x81 |
| *IBM DB2 Universal Database Data Warehouse Center Application Integration Guide* | SC27-1124 | db2adx81 |
| *IBM DB2 XML Extender Administration and Programming* | SC27-1234 | db2sxx81 |

## Business intelligence information

The information in these books describes how to use components that enhance the data warehousing and analytical capabilities of DB2 Universal Database.

Table 27. Business intelligence information

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Warehouse Manager Standard Edition Information Catalog Center Administration Guide* | SC27-1125 | db2dix81 |
| *IBM DB2 Warehouse Manager Standard Edition Installation Guide* | GC27-1122 | db2idx81 |
| *IBM DB2 Warehouse Manager Standard Edition Managing ETI Solution Conversion Programs with DB2 Warehouse Manager* | SC18-7727 | iwhe1mstx80 |

## DB2 Connect information

The information in this category describes how to access data on mainframe and midrange servers using DB2 Connect Enterprise Edition or DB2 Connect Personal Edition.

Table 28. DB2 Connect information

| Name | Form number | PDF file name |
|---|---|---|
| *IBM Connectivity Supplement* | No form number | db2h1x81 |
| *IBM DB2 Connect Quick Beginnings for DB2 Connect Enterprise Edition* | GC09-4833 | db2c6x81 |
| *IBM DB2 Connect Quick Beginnings for DB2 Connect Personal Edition* | GC09-4834 | db2c1x81 |
| *IBM DB2 Connect User's Guide* | SC09-4835 | db2c0x81 |

## Getting started information

The information in this category is useful when you are installing and configuring servers, clients, and other DB2 products.

Table 29. Getting started information

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Universal Database Quick Beginnings for DB2 Clients* | GC09-4832, not available in hardcopy | db2itx81 |
| *IBM DB2 Universal Database Quick Beginnings for DB2 Servers* | GC09-4836 | db2isx81 |
| *IBM DB2 Universal Database Quick Beginnings for DB2 Personal Edition* | GC09-4838 | db2i1x81 |
| *IBM DB2 Universal Database Installation and Configuration Supplement* | GC09-4837, not available in hardcopy | db2iyx81 |
| *IBM DB2 Universal Database Quick Beginnings for DB2 Data Links Manager* | GC09-4829 | db2z6x81 |

## Tutorial information

Tutorial information introduces DB2 features and teaches how to perform various tasks.

Table 30. Tutorial information

| Name | Form number | PDF file name |
|---|---|---|
| *Business Intelligence Tutorial: Introduction to the Data Warehouse* | No form number | db2tux81 |
| *Business Intelligence Tutorial: Extended Lessons in Data Warehousing* | No form number | db2tax81 |
| *Information Catalog Center Tutorial* | No form number | db2aix81 |
| *Video Central for e-business Tutorial* | No form number | db2twx81 |
| *Visual Explain Tutorial* | No form number | db2tvx81 |

## Optional component information

The information in this category describes how to work with optional DB2 components.

Table 31. Optional component information

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Cube Views Guide and Reference* | SC18–7298 | db2aax81 |
| *IBM DB2 Query Patroller Guide: Installation, Administration and Usage Guide* | GC09–7658 | db2dwx81 |
| *IBM DB2 Spatial Extender and Geodetic Extender User's Guide and Reference* | SC27-1226 | db2sbx81 |

Table 31. Optional component information (continued)

| Name | Form number | PDF file name |
|---|---|---|
| *IBM DB2 Universal Database Data Links Manager Administration Guide and Reference* | SC27-1221 | db2z0x82 |
| *DB2 Net Search Extender Administration and User's Guide*<br>**Note:** HTML for this document is *not* installed from the HTML documentation CD. | SH12-6740 | N/A |

## Release notes

The release notes provide additional information specific to your product's release and FixPak level. The release notes also provide summaries of the documentation updates incorporated in each release, update, and FixPak.

Table 32. Release notes

| Name | Form number | PDF file name |
|---|---|---|
| *DB2 Release Notes* | See note. | See note. |
| *DB2 Installation Notes* | Available on product CD-ROM only. | Not available. |

**Note:** The Release Notes are available in:

- XHTML and Text format, on the product CDs
- PDF format, on the PDF Documentation CD

In addition the portions of the Release Notes that discuss *Known Problems and Workarounds* and *Incompatibilities Between Releases* also appear in the DB2 Information Center.

To view the Release Notes in text format on UNIX-based platforms, see the `Release.Notes` file. This file is located in the `DB2DIR/Readme/%L` directory, where `%L` represents the locale name and `DB2DIR` represents:

- For AIX operating systems: `/usr/opt/db2_08_01`
- For all other UNIX-based operating systems: `/opt/IBM/db2/V8.1`

**Related concepts:**

- "DB2 documentation and help" on page 757

**Related tasks:**

- "Printing DB2 books from PDF files" on page 774
- "Ordering printed DB2 books" on page 774
- "Invoking contextual help from a DB2 tool" on page 775

## Printing DB2 books from PDF files

You can print DB2 books from the PDF files on the *DB2 PDF Documentation* CD. Using Adobe Acrobat Reader, you can print either the entire book or a specific range of pages.

**Prerequisites:**

Ensure that you have Adobe Acrobat Reader installed. If you need to install Adobe Acrobat Reader, it is available from the Adobe Web site at www.adobe.com

**Procedure:**

To print a DB2 book from a PDF file:

1. Insert the *DB2 PDF Documentation* CD. On UNIX operating systems, mount the DB2 PDF Documentation CD. Refer to your *Quick Beginnings* book for details on how to mount a CD on UNIX operating systems.
2. Open index.htm. The file opens in a browser window.
3. Click on the title of the PDF you want to see. The PDF will open in Acrobat Reader.
4. Select **File** → **Print** to print any portions of the book that you want.

**Related concepts:**

- "DB2 Information Center" on page 758

**Related tasks:**

- "Mounting the CD-ROM (AIX)" in the *Quick Beginnings for DB2 Servers*
- "Mounting the CD-ROM (HP-UX)" in the *Quick Beginnings for DB2 Servers*
- "Mounting the CD-ROM (Linux)" in the *Quick Beginnings for DB2 Servers*
- "Ordering printed DB2 books" on page 774
- "Mounting the CD-ROM (Solaris Operating Environment)" in the *Quick Beginnings for DB2 Servers*

**Related reference:**

- "DB2 PDF and printed documentation" on page 769

## Ordering printed DB2 books

If you prefer to use hardcopy books, you can order them in one of three ways.

**Procedure:**

Printed books can be ordered in some countries or regions. Check the IBM Publications website for your country or region to see if this service is available in your country or region. When the publications are available for ordering, you can:

- Contact your IBM authorized dealer or marketing representative. To find a local IBM representative, check the IBM Worldwide Directory of Contacts at www.ibm.com/planetwide
- Phone `1-800-879-2755` in the United States or `1-800-IBM-4YOU` in Canada.

- Visit the IBM Publications Center at http://www.ibm.com/shop/publications/order. The ability to order books from the IBM Publications Center may not be available in all countries.

At the time the DB2 product becomes available, the printed books are the same as those that are available in PDF format on the *DB2 PDF Documentation CD*. Content in the printed books that appears in the *DB2 Information Center CD* is also the same. However, there is some additional content available in DB2 Information Center CD that does not appear anywhere in the PDF books (for example, SQL Administration routines and HTML samples). Not all books available on the DB2 PDF Documentation CD are available for ordering in hardcopy.

**Note:** The DB2 Information Center is updated more frequently than either the PDF or the hardcopy books; install documentation updates as they become available or refer to the DB2 Information Center at http://publib.boulder.ibm.com/infocenter/db2help/ to get the most current information.

**Related tasks:**

- "Printing DB2 books from PDF files" on page 774

**Related reference:**

- "DB2 PDF and printed documentation" on page 769

## Invoking contextual help from a DB2 tool

Contextual help provides information about the tasks or controls that are associated with a particular window, notebook, wizard, or advisor. Contextual help is available from DB2 administration and development tools that have graphical user interfaces. There are two types of contextual help:

- Help accessed through the **Help** button that is located on each window or notebook
- Infopops, which are pop-up information windows displayed when the mouse cursor is placed over a field or control, or when a field or control is selected in a window, notebook, wizard, or advisor and F1 is pressed.

The **Help** button gives you access to overview, prerequisite, and task information. The infopops describe the individual fields and controls.

**Procedure:**

To invoke contextual help:

- For window and notebook help, start one of the DB2 tools, then open any window or notebook. Click the **Help** button at the bottom right corner of the window or notebook to invoke the contextual help.

  You can also access the contextual help from the **Help** menu item at the top of each of the DB2 tools centers.

  Within wizards and advisors, click on the Task Overview link on the first page to view contextual help.
- For infopop help about individual controls on a window or notebook, click the control, then click **F1**. Pop-up information containing details about the control is displayed in a yellow window.

**Note:** To display infopops simply by holding the mouse cursor over a field or control, select the **Automatically display infopops** check box on the **Documentation** page of the Tool Settings notebook.

Similar to infopops, diagnosis pop-up information is another form of context-sensitive help; they contain data entry rules. Diagnosis pop-up information is displayed in a purple window that appears when data that is not valid or that is insufficient is entered. Diagnosis pop-up information can appear for:

– Compulsory fields.

– Fields whose data follows a precise format, such as a date field.

**Related tasks:**

- "Invoking the DB2 Information Center" on page 766
- "Invoking message help from the command line processor" on page 776
- "Invoking command help from the command line processor" on page 776
- "Invoking SQL state help from the command line processor" on page 777
- "Access to the DB2 Information Center: Concepts help"
- "How to use the DB2 UDB help: Common GUI help"
- "Setting the location for accessing the DB2 Information Center: Common GUI help"
- "Setting up access to DB2 contextual help and documentation: Common GUI help"

## Invoking message help from the command line processor

Message help describes the cause of a message and describes any action you should take in response to the error.

**Procedure:**

To invoke message help, open the command line processor and enter:

```
? XXXnnnnn
```

where *XXXnnnnn* represents a valid message identifier.

For example, `? SQL30081` displays help about the SQL30081 message.

**Related concepts:**

- "Introduction to messages" in the *Message Reference Volume 1*

**Related reference:**

- "db2 - Command Line Processor Invocation" on page 247

## Invoking command help from the command line processor

Command help explains the syntax of commands in the command line processor.

**Procedure:**

To invoke command help, open the command line processor and enter:

```
? command
```

where *command* represents a keyword or the entire command.

For example, `? catalog` displays help for all of the CATALOG commands, while `? catalog database` displays help only for the CATALOG DATABASE command.

**Related tasks:**

- "Invoking contextual help from a DB2 tool" on page 775
- "Invoking the DB2 Information Center" on page 766
- "Invoking message help from the command line processor" on page 776
- "Invoking SQL state help from the command line processor" on page 777

**Related reference:**

- "db2 - Command Line Processor Invocation" on page 247

## Invoking SQL state help from the command line processor

DB2 Universal Database returns an SQLSTATE value for conditions that could be the result of an SQL statement. SQLSTATE help explains the meanings of SQL states and SQL state class codes.

**Procedure:**

To invoke SQL state help, open the command line processor and enter:

```
   ? sqlstate or ? class code
```

where *sqlstate* represents a valid five-digit SQL state and *class code* represents the first two digits of the SQL state.

For example, `? 08003` displays help for the 08003 SQL state, and `? 08` displays help for the 08 class code.

**Related tasks:**

- "Invoking the DB2 Information Center" on page 766
- "Invoking message help from the command line processor" on page 776
- "Invoking command help from the command line processor" on page 776

## DB2 tutorials

The DB2® tutorials help you learn about various aspects of DB2 Universal Database. The tutorials provide lessons with step-by-step instructions in the areas of developing applications, tuning SQL query performance, working with data warehouses, managing metadata, and developing Web services using DB2.

**Before you begin:**

You can view the XHTML versions of the tutorials from the Information Center at http://publib.boulder.ibm.com/infocenter/db2help/.

Some tutorial lessons use sample data or code. See each tutorial for a description of any prerequisites for its specific tasks.

**DB2 Universal Database tutorials:**

Click on a tutorial title in the following list to view that tutorial.

*Business Intelligence Tutorial: Introduction to the Data Warehouse Center*
Perform introductory data warehousing tasks using the Data Warehouse Center.

*Business Intelligence Tutorial: Extended Lessons in Data Warehousing*
Perform advanced data warehousing tasks using the Data Warehouse Center.

*Information Catalog Center Tutorial*
Create and manage an information catalog to locate and use metadata using the Information Catalog Center.

*Visual Explain Tutorial*
Analyze, optimize, and tune SQL statements for better performance using Visual Explain.

## DB2 troubleshooting information

A wide variety of troubleshooting and problem determination information is available to assist you in using DB2® products.

**DB2 documentation**
Troubleshooting information can be found throughout the DB2 Information Center, as well as throughout the PDF books that make up the DB2 library. You can refer to the ″Support and troubleshooting″ branch of the DB2 Information Center navigation tree (in the left pane of your browser window) to see a complete listing of the DB2 troubleshooting documentation.

**DB2 Technical Support Web site**
Refer to the DB2 Technical Support Web site if you are experiencing problems and want help finding possible causes and solutions. The Technical Support site has links to the latest DB2 publications, TechNotes, Authorized Program Analysis Reports (APARs), FixPaks and the latest listing of internal DB2 error codes, and other resources. You can search through this knowledge base to find possible solutions to your problems.

Access the DB2 Technical Support Web site at http://www.ibm.com/software/data/db2/udb/winos2unix/support

**DB2 Problem Determination Tutorial Series**
Refer to the DB2 Problem Determination Tutorial Series Web site to find information on how to quickly identify and resolve problems you might encounter while working with DB2 products. One tutorial introduces you to the DB2 problem determination facilities and tools available, and helps you decide when to use them. Other tutorials deal with related topics, such as ″Database Engine Problem Determination″, ″Performance Problem Determination″, and ″Application Problem Determination″.

See the full set of DB2 problem determination tutorials on the DB2 Technical Support site at http://www.ibm.com/software/data/support/pdm/db2tutorials.html

**Related concepts:**

- “DB2 Information Center” on page 758
- “Introduction to problem determination - DB2 Technical Support tutorial” in the *Troubleshooting Guide*

# Accessibility

Accessibility features help users with physical disabilities, such as restricted mobility or limited vision, to use software products successfully. The following list specifies the major accessibility features in DB2® Version 8 products:

- All DB2 functionality is available using the keyboard for navigation instead of the mouse. For more information, see "Keyboard input and navigation."
- You can customize the size and color of the fonts on DB2 user interfaces. For more information, see "Accessible display."
- DB2 products support accessibility applications that use the Java™ Accessibility API. For more information, see "Compatibility with assistive technologies" on page 780.
- DB2 documentation is provided in an accessible format. For more information, see "Accessible documentation" on page 780.

## Keyboard input and navigation

### Keyboard input

You can operate the DB2 tools using only the keyboard. You can use keys or key combinations to perform operations that can also be done using a mouse. Standard operating system keystrokes are used for standard operating system operations.

For more information about using keys or key combinations to perform operations, see Keyboard shortcuts and accelerators: Common GUI help.

### Keyboard navigation

You can navigate the DB2 tools user interface using keys or key combinations.

For more information about using keys or key combinations to navigate the DB2 Tools, see Keyboard shortcuts and accelerators: Common GUI help.

### Keyboard focus

In UNIX® operating systems, the area of the active window where your keystrokes will have an effect is highlighted.

## Accessible display

The DB2 tools have features that improve accessibility for users with low vision or other visual impairments. These accessibility enhancements include support for customizable font properties.

### Font settings

You can select the color, size, and font for the text in menus and dialog windows, using the Tools Settings notebook.

For more information about specifying font settings, see Changing the fonts for menus and text: Common GUI help.

### Non-dependence on color

You do not need to distinguish between colors in order to use any of the functions in this product.

## Compatibility with assistive technologies

The DB2 tools interfaces support the Java Accessibility API, which enables you to use screen readers and other assistive technologies with DB2 products.

## Accessible documentation

Documentation for DB2 is provided in XHTML 1.0 format, which is viewable in most Web browsers. XHTML allows you to view documentation according to the display preferences set in your browser. It also allows you to use screen readers and other assistive technologies.

Syntax diagrams are provided in dotted decimal format. This format is available only if you are accessing the online documentation using a screen-reader.

**Related concepts:**

- "Dotted decimal syntax diagrams" on page 780

**Related tasks:**

- "Keyboard shortcuts and accelerators: Common GUI help"
- "Changing the fonts for menus and text: Common GUI help"

# Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the Information Center using a screen reader.

In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.
- ! means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.
- * means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

  **Notes:**

  1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
  2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.

3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

**Related concepts:**

- "Accessibility" on page 779

**Related tasks:**

- "Keyboard shortcuts and accelerators: Common GUI help"

**Related reference:**

- "How to read the syntax diagrams" in the *SQL Reference, Volume 2*

## Common Criteria certification of DB2 Universal Database products

DB2 Universal Database is being evaluated for certification under the Common Criteria at evaluation assurance level 4 (EAL4). For more information about Common Criteria, see the Common Criteria web site at: http://niap.nist.gov/cc-scheme/.

# Appendix D. Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country/region or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country/region where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product, and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information that has been exchanged, should contact:

IBM Canada Limited
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems, and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements, or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious, and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information may contain sample application programs, in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (*your company name*) (*year*). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _*enter the year or years*_. All rights reserved.

## Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both, and have been used in at least one of the documents in the DB2 UDB documentation library.

ACF/VTAM
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
BookManager
C Set++
C/370
CICS
Database 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Information Integrator
DB2 Query Patroller
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eServer
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
iSeries
LAN Distance
MVS
MVS/ESA
MVS/XA
Net.Data
NetView
OS/390
OS/400
PowerPC
pSeries
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/400
SQL/DS
System/370
System/390
SystemView
Tivoli
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WebSphere
WIN-OS/2
z/OS
zSeries

The following terms are trademarks or registered trademarks of other companies and have been used in at least one of the documents in the DB2 UDB documentation library:

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

# Appendix E. Contacting IBM

In the United States, call one of the following numbers to contact IBM:

- 1-800-IBM-SERV (1-800-426-7378) for customer service
- 1-888-426-4343 to learn about available service options
- 1-800-IBM-4YOU (426-4968) for DB2 marketing and sales

In Canada, call one of the following numbers to contact IBM:

- 1-800-IBM-SERV (1-800-426-7378) for customer service
- 1-800-465-9600 to learn about available service options
- 1-800-IBM-4YOU (1-800-426-4968) for DB2 marketing and sales

To locate an IBM office in your country or region, check IBM's Directory of Worldwide Contacts on the web at http://www.ibm.com/planetwide

## Product information

Information regarding DB2 Universal Database products is available by telephone or by the World Wide Web at http://www.ibm.com/software/data/db2/udb

This site contains the latest information on the technical library, ordering books, product downloads, newsgroups, FixPaks, news, and links to web resources.

If you live in the U.S.A., then you can call one of the following numbers:

- 1-800-IBM-CALL (1-800-426-2255) to order products or to obtain general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, go to the IBM Worldwide page at www.ibm.com/planetwide

# Index

Printed in USA

SC09-4828-01

Spine information:

IBM® DB2® Universal Database Command Reference Version 8.2