DB2 Data Management Software

IBM

e business software

# Chapter 6: Working with DB2 Data

IBM DB2 Universal Database V8.1
Database Administration Certification Preparation Course

Maintained by Clara Liu

IBM Software Group

# Objectives

- In this section, we will cover:
  - Basic Data Manipulation Language to work with data and database objects

# Basic SQL

- The SQL Language has been developed around 4 basic verbs used for 4 major tasks:
  - Data Retrieval: SELECT
  - Data Addition: INSERT
  - Data Modification: UPDATE
  - Data Removal: DELETE

# Data Retrieval: SELECT Statement

- SELECT is the QUERY element of the language
- Queries come in many styles:
  - Retrieving all table data
  - Limiting Columns or Rows retrieved
  - Cartesian Product
  - Inner Join
  - Set operators
  - with :
    - derived columns
    - sub-query etc.
    - sorting
    - using functions
    - grouping values
  - Also: Common table expressions and Set operators
- Do not forget - you need the correct privilege to SELECT

# Data Retrieval: SELECT Statement

- Optional clauses in SELECT statement
  - read-only clause
    - indicates that the result table is read-only
    - FOR FETCH ONLY has the same meaning
    - Example:

      **SELECT ALL FROM employee FOR READ ONLY ;**
  - update clause
    - identifies the columns that can be updated in a subsequent positioned UPDATE statement
    - Example:

      **SELECT empno, salary, comm FROM employee FOR UPDATE OF salary, comm ;**
  - fetch first clause
    - sets a maximum number of rows that can be retrieved
    - Example:

      **SELECT empno, salary FROM employee FETCH FIRST 10 ROWS ONLY ;**
  - optimize for clause
    - influences query optimization based on the assumption that n rows will be retrieved
    - does not limit the number of rows that can be fetched
    - Example:

      **SELECT empno, salary FROM employee OPTIMIZE FOR 10 ROWS ;**

# Cursors

- Declare cursor for a SELECT statement

```
DECLARE CURSOR C1 FOR SELECT ID, NAME FROM EMPLOYEE
```

- Row blocking
  - Retrieve a block of rows in a single operation, rows stored in cache
- Declare cursor WITH HOLD maintains certain resources across multiple unit of work
  - For units of work ending with COMMIT:
    - Open cursors defined with HOLD remain open
    - The cursor is positioned before the next logical row of the result set
    - All locks are released, except locks protecting the current WITH HOLD cursor position
  - For units of work ending with ROLLBACK:
    - All open cursors are closed
    - All locks acquired during the unit of work are released

# VALUES, ORDER BY Clauses

- VALUES clause
  - VALUES clause can be used alone to return constant values or a special register
  - Examples:
    - VALUES 'a', 'b', 'c'

      ```
      1
      ----
      a
      b
      c
      ```

    - VALUES CURRENT TIMESTAMP

      ```
      1
      --------------------------
      2002-11-16-22.01.15.684000
      ```

- Check the DB2 SQL Reference for complete listing of Special Registers
- ORDER BY clause
  - The ORDER BY clause specifies an ordering of the rows of the result table
  - If a single sort specification (one sort-key with associated direction) is identified, the rows are ordered by the values of that sort specification
  - If more than one sort specification is identified, the rows are ordered by the values of the first identified sort specification, then by the values of the second identified sort specification, and so on
  - Example:
    - **SELECT c1, c2 FROM t1 ORDER BY c1 ;**

# GROUP BY, HAVING Clauses

- GROUP BY ... <grouping expression>
  - A GROUP BY clause contains a grouping expression, it specifies an intermediate result table that consists of a grouping of the rows of R. R is the result of the previous clause of the subselect.
  - GROUP BY GROUPING SETS <grouping expression>
  - GROUP BY ROLLUP <grouping expression>
  - GROUP BY CUBE <grouping expression>
  - **Example: SELECT workdept, MAX ( salary ) FROM employee**
    **GROUP BY workdept ;**
- HAVING clause
  - The HAVING clause specifies an intermediate result table that consists of the groups of R for which the search-condition is true. R is the result of the previous clause of the subselect.
  - **Example: SELECT workdept, MAX ( salary ) FROM employee**
    **GROUP BY workdept**
    **HAVING MAX ( salary ) <**
    **( SELECT avg ( salary FROM employee ) ;**

# Other SQL Usage

- Obtain distinct value from a table
  - SELECT DISTINCT ( position ) FROM employee ;
- Use of EXISTS

```
SELECT empno, salary
  FROM employee
  WHERE NOT EXISTS ( SELECT empno FROM manager ) ;
```

- Use of IN, NOT IN clauses

```
SELECT empno, salary
  FROM employee
  WHERE department IN ( 'A00', 'B00', 'C00' ) ;
```

# Common Table Expression (CTE)

- A CTE is like a "temporary view"
- Valid only within one SQL statement
- Has a name, column-names, and a body, like any view
- Enables certain queries to be expressed in a single statement
- Example: Find the department(s) with the most employees:

```
WITH staff( deptno, headcount) AS
  (SELECT deptno, count(*)
   FROM emp GROUP BY deptno)

SELECT deptno, headcount
  FROM staff
  WHERE headcount =
   ( SELECT max( headcount )
     FROM staff );
```

# Data Addition: INSERT Statement

- You need to have appropriate table of view privilege
- Can insert one or more rows at a time
- You can use subselect to determine values
- May also be done to a subset of columns provided:
  - a column not specified accepts NULLs or
  - column defined WITH DEFAULT
- Large amounts of data? Look at using LOAD instead

```
INSERT INTO DEPARTMENT ( DEPTNO, DEPTNAME, ADMRDEPT )
     VALUES ( 'E31', 'ARCHITECTURE', 'E01' )
```

# Data Modification: UPDATE Statement

- UPDATES come in 3 main varieties :
  - Full table
  - Searched with a **WHERE** clause
  - Positioned using a **CURSOR** in a program

```
EXEC SQL  DECLARE C1 CURSOR FOR
            SELECT *
              FROM EMPLOYEE
              FOR UPDATE OF JOB;

EXEC SQL  OPEN C1;

EXEC SQL  FETCH C1 INTO ...     ;

if ( strcmp (change, "YES") == 0 )
    EXEC SQL  UPDATE EMPLOYEE
                SET JOB = :newjob
                WHERE CURRENT OF C1;

    EXEC SQL  CLOSE C1;
end if
```

# Data Removal: DELETE Statement

- DELETES can apply to single or multiple rows
  - You can also use subselect to determine values
- Two forms of statement :
  - The Searched DELETE form is used to delete one or more rows ( optionally determined by a search condition )
  - The Positioned DELETE form is used to delete exactly one row ( as determined by the current position of a cursor )
- Deleted rows not removed from table.
  - Space is marked as unused
  - Reclaim space using the "REORG" command - discussed later

```
DELETE FROM DEPARTMENT
  WHERE DEPTNO = 'D11'
```

# Set Operators

- UNION or UNION ALL
  - Derives a result table by combining two result tables (R1 and R2) with the duplicate rows eliminated
  - If UNION ALL is specified, the result consists of all rows in both tables
- EXCEPT or EXCEPT ALL
  - Derives a result table by combining two result tables (R1 and R2), the result consists of all rows that are only in R1, with duplicate rows in the result eliminated
  - If EXCEPT ALL is specified, the result consists of all rows that do not have a corresponding row in R2, where duplicate rows are significant
- INTERSECT or INTERSECT ALL
  - Derives a result table by combining two result tables (R1 and R2), the result consists of all rows that are only in both R1 and R2, with the duplicate rows eliminated
  - If INTERSECT ALL is specified, the result consists of all rows that are in both R1 and R2