Dr. Stefan Brass
School of Information Sciences
University of Pittsburgh

July 19, 2001

INFSCI 2711 "Database Analysis and Design" — Example I for Final Exam —

Instructions

- This is the midterm exam from the Tuesday session of INFSCI 2711 in Fall 1999. (In that course, I treated first physical design and then conceptual design.)
- There was 1 hour and 50 minutes time to work on the exercises.
- In the multiple-choice questions, it was possible that more than one answer is correct. (This was a mistake that I will not repeat: Students got no points unless the checked exact the right combination of correct answers. This is not very fair.)
- The results of the exam were not good and I gave everybody 8 additional points (also because of an error in one exercise). In addition, I introduced an optional project for extra credit.
- This time, I will try to make the exercises a bit simpler.
- In this course, we did not repeat SQL. So there will be no exercise like Exercise 1 below. But you need to know SQL in order to write queries to the data dictionary.

Exercise 1 (SQL Errors) 6 Points

In this exam, we consider a database containing information about films:

- `FILM(ID, TITLE, YEAR, DIRECTOR, OSCARS)`
- `ROLE(ID→FILM, ROLE_NAME, ACTOR_NAME)`
- `CRITIQUE(ID→FILM, SOURCE, GRADE, TEXT) /* Review */`

Suppose we want to find films without any critique/review (a newspaper article saying whether the film was good or bad). Which of these solutions are correct, and which are incorrect? If it is incorrect, please give a very short explanation.

a)
```
SELECT ID, TITLE
FROM   FILM
WHERE  ID NOT IN(SELECT C.ID FROM CRITIQUE C)
```

☐ Correct
☐ Wrong, Reason: ______________________________

b)
```
SELECT F.ID, F.TITLE
FROM   FILM F, CRITIQUE C
WHERE  F.ID = C.ID AND C.SOURCE IS NULL
```

☐ Correct
☐ Wrong, Reason: ______________________________

c)
```
SELECT F.ID, F.TITLE
FROM   FILM F
WHERE  NOT EXISTS(SELECT * FROM CRITIQUE C)
```

☐ Correct
☐ Wrong, Reason: ______________________________

Suppose that now we want to find films which were unanimously rated “A” (i.e. there is at least one review with the grade “A”, and there is no review with another grade). E.g. if the film got grades AAA from three different sources, it should be printed, if it got ABA, it should not be printed. Which of these solutions are correct, and which are incorrect?

d)
```
SELECT F.ID, F.TITLE
FROM   FILM F
WHERE  F.ID NOT IN (SELECT C.ID FROM CRITIQUE C
                    WHERE GRADE <> 'A')
```

☐ Correct
☐ Wrong, Reason: ____________________

e)
```
SELECT F.ID, F.TITLE
FROM   FILM F, CRITIQUE C
WHERE  F.ID = C.ID
AND    C.GRADE = 'A'
HAVING COUNT(DISTINCT GRADE) = 1
```

☐ Correct
☐ Wrong, Reason: ____________________

f)
```
SELECT F.ID, F.TITLE
FROM   FILM F, CRITIQUE C
WHERE  F.ID = C.ID AND C.GRADE = 'A'
AND    NOT EXISTS(SELECT * FROM CRITIQUE X
                  WHERE X.ID = F.ID AND X.GRADE <> 'A')
```

☐ Correct
☐ Wrong, Reason: ____________________

Exercise 2 (Disks and Buffering) 6 Points

a) When the DBMS sends a read request for a single block of 2 KByte to the disk, the total access time consists of three parts. Which one is on average the longest?

- ☐ Latency time
- ☐ Seek time
- ☐ Transfer time (from the disk surface into the disk buffer, starts when the disk head is at beginning of the block, ends at the end of the block.)

b) How many distinct accesses (requiring a seek and the transfer of a few blocks) can a disk carry out per second (approximately)?

- ☐ 5
- ☐ 50
- ☐ 500
- ☐ 5000
- ☐ 50000

c) Suppose your system is short in main memory, but the operating system supports paging (virtual memory, it can swap out pages or entire processes). Would it then improve the DBMS performance to further increase the size of the buffer cache?

- ☐ Yes, the performance improves.
- ☐ This has no influence on the performance.
- ☐ The performance will even decrease.

d) What is a good cache hit ratio?

- ☐ Less than 10%.
- ☐ Around 50%.
- ☐ Above 90%, probably even higher that 95%.

e) Suppose you do RAID level 0+1 (striping and mirroring). You use blockwise striping over 4 disks, and have mirrored the data on another 4 disks. Compared to the number of block read accesses a single disk allows, how would this number increase for the RAID system under optimal circumstances (i.e. equal distribution of requests, perfect load balancing)?

- ☐ It would be the same as for a single disk.
- ☐ It would be 2 times higher.
- ☐ It would be 4 times higher.
- ☐ It would be 8 times higher.

f) What information is temporarily stored in the buffer cache (in Oracle)? As always, there might be more than one correct answer.

- ☐ Blocks containing table data.
- ☐ Blocks containing index data.
- ☐ Blocks containing application programs written in C.
- ☐ Blocks containing program code of the DBMS itself
  (e.g. the procedure for doing a hash join).

Exercise 3 (Access Paths) 6 Points

a) Suppose you accidentally deleted a cluster. Will your application programs still run (maybe slower)?

- ☐ They will run, but slower.
- ☐ The end user will not note any difference.
- ☐ They will not run.

b) Consider again the table `FILM(ID, TITLE, YEAR, DIRECTOR, OSCARS)`.
Suppose you have created a B-tree index with the command

```
CREATE INDEX I_FILM_DIR_YEAR ON FILM(DIRECTOR, YEAR)
```

Which of the following conditions can be evaluated using the index (and not doing a full index scan)? As always, there can be more than one correct answer.

- ☐ `DIRECTOR = 'Lucas, George'`
- ☐ `YEAR > 1970`
- ☐ `DIRECTOR = 'Lucas, George' AND YEAR > 1980`
- ☐ `UPPER(DIRECTOR) LIKE 'LU%'`

c) Suppose you have to access 5% of the rows of a large table (significantly larger than your buffer cache). Each block contains about 50 rows. You do not know anything about the distribution of the rows among the blocks (i.e. assume that they are equally distributed). Which access method is more efficient?

- ☐ Full Table Scan
- ☐ Index Scan and Table Access by ROWID
- ☐ They are approximately equally efficient.

d) Suppose you sometimes want to find films with many Oscars, e.g. have a condition like `OSCARS >= 8` (which is satisfied only by a small fraction of the films). You find that a full-table scan is rather slow. When a new film is entered into the database, the number of Oscars is normally 0, and only later updated when Oscars are granted to the film. Which access method would you propose?

- ☐ Hash Cluster
- ☐ Index Cluster
- ☐ B-tree Index
- ☐ When the column is updated, no index can be created.

e) Suppose you have stored the table `ROLE` in a hash cluster accessed by the film ID. How many block accesses does this query need?

```
SELECT * FROM ROLE WHERE ID = 2456
```

A single row will be not more than 100 Bytes long (shorter than a block).

- ☐ At most one (0 if the block is in the cache)
- ☐ Normally one, at most 2 (0 if in cache)
- ☐ Normally one, but theoretically is no upper limit (0 if in cache).

f) Consider the table

`CRITIQUE(`<u>`ID`</u>`→FILM, `<u>`SOURCE`</u>`, GRADE, TEXT).`

Let us assume that `TEXT` can be relatively large (e.g. 1000 Byte). Suppose you create an index with the following statement:

```
CREATE UNIQUE INDEX I_CRITIQUE_ID_SOURCE_GRADE
                    ON CRITIQUE(ID, SOURCE, GRADE)
```

This would make some index-only query evaluation plans possible (if you get Oracle to use this index). What are the advantages of this solution? Please check all advantages:

- ☐ The index is smaller than the table, thus a full scan of the index is faster than a full table scan (useful for index-only QEPs).
- ☐ The partial rows stored in the index are more or less clustered by `ID`.
- ☐ This index enforces the key constraint.
  No other index on `(ID,SOURCE)` is needed.
- ☐ This index is useful for the condition `GRADE = 'A'`.

Exercise 4 (Heap Files) 1+4+1=6 Points

a) Suppose you delete 50% of the rows stored in a heap file. How long will a full table scan need afterwards in comparison to the time it needed before (in Oracle)?

- ☐ About 50% of the time before.
- ☐ About 75% of the time before.
- ☐ As long as before.
- ☐ It will actually need longer.

b) What would be a good `INITIAL` size for the table `FILM`? It is declared as follows:

```
CREATE TABLE FILM(ID NUMBER(6) PRIMARY KEY CHECK(ID >= 100001),
                  TITLE VARCHAR(40) NOT NULL,
                  YEAR NUMBER(4) NOT NULL CHECK(YEAR >= 1900),
                  DIRECTOR VARCHAR(20) NULL,
                  OSCARS NUMBER(2) NULL)
```

Note that the lengths given in the `CREATE TABLE` statement are only the upper limits. To compute the space requirements, you need to consider the following average sizes. The `ID` needs all 6 digits, since it was decided that it should start with `100001`. The `TITLE` is on average 20 characters long. The `DIRECTOR` name is on average 10 characters long, the number of null values is neglectable (very few director names are not known). The data to be inserted into the database does not record the number of Oscars, and it is expected that in the end 50% of all rows will have a null value in this column. Note that if the last column is null, no length byte is needed. The table will contain 100 000 rows. The table will be declared with `PCTFREE=10`, since there are only few updates (maybe for the number of Oscars). `DB_BLOCK_SIZE` is 2048 Byte.

`INITIAL` ______________

Please explain your solution (only show the main calculations):

c) Suppose you expect that the table will grow in one year to 200 000 rows (i.e. double its size). Suppose further that you have sufficient disk space. If you consider the time needed for full table scans, what would be the best solution?

- ☐ Define an `INITIAL` extent large enough for 200 000 rows
- ☐ Define an `INITIAL` extent for the current number of rows and `NEXT` extents for the growth in each quarter. In this way, full table scans done now are not penalized for the future growth.

Exercise 5 (Data Dictionary) 6 Points

Please write three SQL queries involving the Oracle data dictionary. You can answer them using the following tables (but you can also use other tables from the Oracle data dictionary):

- `TABS` with the following columns:

```
TABLE_NAME, TABLESPACE_NAME, CLUSTER_NAME, IOT_NAME,
PCT_FREE, PCT_USED, INI_TRANS, MAX_TRANS, INITIAL_EXTENT,
NEXT_EXTENT, MIN_EXTENTS, MAX_EXTENTS, PCT_INCREASE,
FREELISTS, FREELIST_GROUPS, LOGGING, BACKED_UP, NUM_ROWS,
BLOCKS, EMPTY_BLOCKS, AVG_SPACE, CHAIN_CNT, AVG_ROW_LEN,
AVG_SPACE_FREELIST_BLOCKS, NUM_FREELIST_BLOCKS, DEGREE,
INSTANCES, CACHE, TABLE_LOCK, SAMPLE_SIZE, LAST_ANALYZED,
PARTIONED, IOT_TYPE, TEMPORARY, NESTED, BUFFER_POOL.
```

- `USER_SEGMENTS` with the following columns:

```
SEGMENT_NAME, PARTITION_NAME, SEGMENT_TYPE, TABLESPACE_NAME,
BYTES, BLOCKS, EXTENTS, INITIAL_EXTENT, NEXT_EXTENT,
MIN_EXTENTS, MAX_EXTENTS, PCT_INCREASE, FREELISTS,
FREELIST_GROUPS, BUFFER_POOL.
```

  `SEGMENT_TYPE` can e.g. be `'CLUSTER'`, `'INDEX'`, `'TABLE'`.

- `USER_EXTENTS` with the following columns:

```
SEGMENT_NAME, PARTITION_NAME, SEGMENT_TYPE, TABLESPACE_NAME,
EXTENT_ID, BYTES, BLOCKS.
```

a) Write an SQL query which lists tables (not indexes) stored in more than one extent.

b) Write an SQL query which lists tables which were never analyzed (i.e. have no statistics) or which were analyzed before `'01-JAN-99'`.

c) Write an SQL query which lists tables where the storage capacity (disk space) below the high water mark is utilized to less than 20%. To simplify the task a bit, you can assume that every block has 1753 Bytes of available space (2048 Byte minus 90 Byte for the header and 205 Byte for `PCTFREE=10`). Use the number or rows and average row length contained in `TABS`. Add 2 Bytes to the average row length for the row directory entry. Do not count the segment header block, neither for the used space nor for the allocated space.

Exercise 6 (Query Optimization) 1+1+1+3=6 Points

a) Consider the following query:

```
SELECT TITLE
FROM   FILM
WHERE  YEAR < 1980
AND    DIRECTOR = 'Lucas, George'
```

Which of the following access paths would the rule-based optimizer choose? Here, only one answer is correct (the highest ranked among the available paths).

- ☐ Full Table Scan
- ☐ Unique Index on `FILM(ID)`
- ☐ Index on `FILM(YEAR)`
- ☐ Index on `FILM(DIRECTOR)`

Of course, you can assume that these three indexes really exist.

b) Consider this query:

```
SELECT F.TITLE
FROM   FILM F, ROLE R, CRITIQUE C
WHERE  F.ID = R.ID
AND    C.ID = F.ID
AND    R.ACTOR_NAME = 'Ford, Harrison'
AND    C.GRADE = 'A'
```

Suppose that the following indexes exist:

- Unique index on `FILM(ID)`
- Unique index on `ROLE(ID, ROLE_NAME)`
- Index on `ROLE(ACTOR_NAME)`
- Unique index on `CRITIQUE(ID, SOURCE)`.

Consider the QEP constructed by the rule-based optimizer which starts by accessing `ROLE`. Which table would the rule-based optimizer join first with `ROLE`?

- ☐ `FILM`
- ☐ `CRITIQUE`

c) Suppose we want to compute directors who have made films over a period of 30 years:

```
SELECT X.DIRECTOR
FROM   FILM X, FILM Y
WHERE  X.DIRECTOR = Y.DIRECTOR
AND    X.YEAR - Y.YEAR >= 30
```

Suppose that the only index which exists is on `FILM(ID)`. Which kind of join would the rule-based optimizer of Oracle use (if it starts with accessing `X`)?

- ☐ Proper Nested Loop Join (i.e. with full table scan on the right)
- ☐ Merge Join
- ☐ Index Join with the index on `FILM(ID)`

d) Consider the following query:

```
SELECT TITLE
FROM   FILM F, ROLE R
WHERE  YEAR < 1980 AND F.ID = R.ID
AND    ACTOR_NAME = 'Ford, Harrison'
```

The following indexes exist:

- Unique indexes on the keys, i.e. on `FILM(ID)` and on `ROLE(ID, ROLE_NAME)`.
- Index on `ROLE(ACTOR_NAME)`.

Please draw the Oracle QEP which the rule-based optimizer constructs starting with access to `ROLE`. You can use the tree notation with interconnected boxes or use one line for every operation with indentation to clarify the structure. You do not have to assign numbers to every node.