# Database Management System: Assignment 2

Total Marks : 20

June 17, 2022

## Question 1

Consider the following instance:

| books | | | |
|---|---|---|---|
| genre | title | author | theme |
| horror | The haunting | Steve K | forest horror |
| comedy | Echoing Laughters | Charles G | friends comedy |
| comedy | Happiens | Carl | parody |
| lifestyle | CookBook | Julien S | Recipes |
| lifestyle | CookBook | Julien S | Recipes |
| thriller | The Cause | Julien S | Treasure Mystery |

Consider the query:

```
SELECT title, author FROM books
WHERE theme LIKE "%y" AND genre NOT LIKE "%r" ORDER BY theme DESC
```

Which `title` will be present in the first (topmost) tuple of the output produced by the above query? *Marks: 2* **MCQ**

a) `The Cause`

b) `Happiens`

c) `Echoing Laughters`

d) `The haunting`

**Answer**: b)
**Explanation:** The clause `ORDER BY theme DESC` sorts the tuples in descending alphabetical order of `theme`.
Hence, the theme `parody` will come before `friends comedy`.
The clause `SELECT title, author FROM books WHERE theme LIKE "%y" AND genre NOT LIKE "%r"` projects the tuples `<Happiens,Carl>, <Echoing Laughters, Charles G>`.
Based on the sorted order, `Happiens` associated with the theme `parody` will appear first among the two tuples.
Hence, option (b) is correct.

## Question 2

Consider the following instance:

| books | | | |
|---|---|---|---|
| genre | title | author | theme |
| horror | The haunting | Steve K | forest horror |
| comedy | Echoing Laughters | Charles G | friends comedy |
| comedy | Happiens | Carl | parody |
| lifestyle | CookBook | Julien S | Recipes |
| lifestyle | CookBook | Julien S | Recipes |
| thriller | The Cause | Julien S | Treasure Mystery |

Consider the following queries:

```
CREATE VIEW THEMEVIEW AS
SELECT theme FROM books WHERE author LIKE "% %";
SELECT COUNT(*) FROM THEMEVIEW;
```

What is the final output after the two queries are executed? *Marks: 2* **MCQ**

a) 3

b) 4

c) 5

d) 6

**Answer**: c)
**Explanation:** `CREATE VIEW THEMEVIEW AS SELECT theme FROM books WHERE author LIKE "% %"` selects all `themes` associated with an `author`'s name where there is a space.
`SELECT COUNT(*) FROM THEMEVIEW` produces 5 as output, as all the tuples (even those with non-distinct values) are counted.
Hence, option (c) is correct.

## Question 3

Consider the following instance:

| scoreboard | | | |
|---|---|---|---|
| gameID | TeamName | Score | Day |
| 1 | RacerLeague | 87 | 1 |
| 1 | HeroZ | 12 | 1 |
| 2 | HeroZ | 42 | 2 |
| 2 | Legend21 | 72 | 2 |
| 3 | Legend21 | 90 | 1 |

Which is the correct option to change the value of `gameID` from 3 to 2? *Marks: 2* **MCQ**

a) `UPDATE scoreboard SET gameID=2 WHERE gameID=3`

b) `ALTER TABLE scoreboard SET gameID=2 WHERE gameID=3`

c) `ALTER scoreboard SET gameID=2 WHERE gameID=3`

d) `UPDATE TABLE scoreboard SET gameID=2 WHERE gameID=3`

**Answer**: a)
**Explanation:** According to the syntax of SQL queries, option (a) is correct.

## Question 4

Consider the following instance:

| scoreboard | | | |
|---|---|---|---|
| gameID | TeamName | Score | Day |
| 1 | RacerLeague | 87 | 1 |
| 1 | HeroZ | 12 | 1 |
| 2 | HeroZ | 42 | 2 |
| 2 | Legend21 | 72 | 2 |
| 2 | Legend21 | 90 | 1 |

The query
‘‘SELECT *X*, AVG(Score) FROM scoreboard GROUP BY *X*"
produces the following tuples as output:

| | |
|---|---|
| 1 | 63 |
| 2 | 57 |

Which attribute(s) is (are) represented by ***X*** in the above query? *Marks: 2* **MCQ**

a) gameID

b) gameID, Day

c) Day

d) Day, Score

**Answer**: c)
**Explanation:** The Day-wise average of Scores produces 63 on Day 1 and 57 on Day 2. Hence, option (c) is correct.

## Question 5

Consider the following two tables:

`Supplier`

| sup_id | sup_name | sup_city |
|---|---|---|
| 18 | Order All | Boston |
| 15 | Jack Hill Ltd | London |
| 16 | Akas Foods | Delhi |
| 17 | Foodies. | London |
| 19 | sip-n-Bite. | New York |

`Product`

| p_id | p_name | sup_id |
|---|---|---|
| 6 | Cheez-It | 15 |
| 2 | BN Biscuit | 15 |
| 3 | Mighty Munch | 16 |
| 5 | Jaffa Cakes | 18 |
| 7 | Salt n Shake | 17 |
| 8 | Marie Biscuit | 20 |

An operation on these two tables generates the following output. Identify the correct operation.
*Marks: 2* **MCQ**

`output`

| sup_id | sup_name | sup_city | p_id | p_name |
|---|---|---|---|---|
| 18 | Order All | Boston | 5 | Jaffa Cakes |
| 15 | Jack Hill Ltd | London | 6 | Cheez-It |
| 15 | Jack Hill Ltd | London | 2 | BN Biscuit |
| 16 | Akas Foods | Delhi | 3 | Mighty Munch |
| 17 | Foodies. | London | 7 | Salt n shake |
| 19 | sip-n-Bite. | New York | | |

a) `Supplier NATURAL LEFT OUTER JOIN Product`

b) `Supplier INNER JOIN Product`

c) `Supplier NATURAL RIGHT OUTER JOIN Product`

d) `Supplier NATURAL FULL OUTER JOIN Product`

**Answer**: a)
**Explanation:** In this question the left relation is `Supplier` and the right relation is `Product`. The operation cannot be Natural JOIN or RIGHT OUTER JOIN as the last tuple of `Supplier` (for which the value of the common attribute of the `Supplier` does not match with any value of `Product` is present in the output.
When a NATURAL LEFT OUTER JOIN is applied, firstly the tuples are selected from both left and right relations which have the same `sup_id`. Hence, we get 1 record each of `sup_id` = 16, 17, 18 and 2 records for `sup_id` = 15.
Secondly, the remaining tuples from the left relation which did not have same `sup_id` with the right relation are also added to the resulting relation `output` as shown in the last row of `output`.
Hence, option (a) is correct.

## Question 6

Consider the following schema:

`player(p_id, tournament_code, month, match_count)`

In the `player` schema, `p_id` uniquely identifies every record. *Marks:2* **MCQ**

Given that a player cannot play more than 5 matches in a month for a tournament, identify the correct `CHECK` constraint for the create query of this table.

a) `CONSTRAINT CK_mat_count for CHECK (if match_count < 6)`

b) `CONSTRAINT CK_mat_count for CHECK (if match_count < 6 is true)`

c) `CONSTRAINT CK_mat_count CHECK (if match_count < 6)`

d) `CONSTRAINT CK_mat_count CHECK (match_count < 6)`

**Answer**: d)
**Explanation:** As per the syntax and semantics of SQL notation.

## Question 7

Consider the following schema:

```
Football_Club(fc_code, fc_name, fc_address)
Player(p_id, name, dob, match_count, fc_code)
```

You want to design the Player table in such a way that if a fc_code value needs to be deleted from the Football_Club table, the corresponding records in the Player table that use this fc_code will also be deleted.

Identify the correct CREATE statement(s) for the Player table from the following.

*Marks: 2* **MCQ**

a)
```
CREATE TABLE Player(
p_id INT,
name VARCHAR(30) NOT NULL,
dob DATE,
match_count INT,
fc_code INT,
PRIMARY KEY(p_id),
FOREIGN KEY (fc_code) REFERENCES Football_Club(fc_code)
ON DELETE CASCADE);
```

b)
```
CREATE TABLE Player(
p_id INT,
name VARCHAR(30) NOT NULL,
dob DATE,
match_count INT,
fc_code INT,
PRIMARY KEY(p_id),
FOREIGN KEY (fc_code) REFERENCES Football_Club(fc_code));
```

c)
```
CREATE TABLE Player(
p_id INT NOT NULL,
name VARCHAR(30),
dob DATE,
match_count INT,
FOREIGN KEY (fc_code) REFERENCES Football_Club(fc_code)
ON DELETE CASCADE);
```

d)
```
CREATE TABLE Player(
p_id INT,
name VARCHAR(30) NOT NULL,
dob DATE,
match_count INT,
fc_code INT,
PRIMARY KEY(p_id),
FOREIGN KEY (fc_code) REFERENCES Football_Club(fc_code)
ON DELETE SET NULL);
```

**Answer**: a)

**Explanation:** SQL allows three different types of the definitions of the foreign key:

- `ON DELETE CASCADE`: When a referenced parent table row is removed, all the children are removed automatically.

- `ON DELETE SET NULL`: When referenced data in the parent key is deleted, all rows in the child table that depend on those parent key values have their foreign keys set to null.

- `ON DELETE NO ACTION` / `ON DELETE RESTRICT` / (which is the default) prevents deleting a parent when there are children.

Hence, option (a) is correct.

## Question 8

Consider the following instance of the relation `BookDetails`.

```
BookDetails (BOOK_ID, YEAR_PUB, BOOK_TITLE, AUTHOR_NAME, TOTAL_SOLD_COPIES)
```

*Marks: 2* **MSQ**

| BookDetails | | | | |
|---|---|---|---|---|
| BOOK_ID | YEAR_PUB | BOOK_TITLE | AUTHOR_NAME | TOTAL_SOLD_COPIES |
| 1001 | 2010 | DBMS | KORTH | 15000 |
| 1002 | 2010 | OPERATING SYSTEM | GALVIN | 25000 |
| 1003 | 2020 | COMPILER | ULLMAN | 12000 |
| 1004 | 1995 | DATA STRUCTURES | CORMEN | 20000 |
| 1005 | 1995 | DATA STRUCTURES | ULLMAN | 20000 |
| 1006 | 1990 | COMPUTER NETWORKS | FOROUZAN | 15000 |
| 1007 | 2010 | MACHINE LEARNING | MITCHELL | 18000 |

Identify the correct statement(s) to get the following output:

| BookDetails | | | | |
|---|---|---|---|---|
| BOOK_ID | YEAR_PUB | BOOK_TITLE | AUTHOR_NAME | TOTAL_SOLD_COPIES |
| 1001 | 2010 | DBMS | KORTH | 15000 |
| 1002 | 2010 | OPERATING SYSTEM | GALVIN | 25000 |
| 1003 | 2020 | COMPILER | ULLMAN | 12000 |
| 1005 | 1995 | DATA STRUCTURES | ULLMAN | 20000 |
| 1007 | 2010 | MACHINE LEARNING | MITCHELL | 18000 |

a)
```
SELECT * FROM BookDetails
WHERE YEAR_PUB=2010 AND AUTHOR_NAME='ULLMAN' ;
```

b)
```
SELECT * FROM BookDetails
WHERE YEAR_PUB=2010 OR AUTHOR_NAME='ULLMAN' ;
```

c)
```
(SELECT * FROM BookDetails
WHERE YEAR_PUB=2010)
INTERSECT
(SELECT * FROM BookDetails
WHERE AUTHOR_NAME='ULLMAN');
```

d)
```
(SELECT * FROM BookDetails
WHERE YEAR_PUB=2010)
UNION
(SELECT * FROM BookDetails
WHERE AUTHOR_NAME='ULLMAN');
```

**Answer**: (b), (d)

**Explanation:** Output table containing tuples whose `YEAR_PUB` is 2010 OR `AUTHOR_NAME` is 'ULLMAN'.

Hence, options b) and d) are correct.

## Question 9

Consider the following instance of the relation BookDetails.

BookDetails (BOOK_ID, YEAR_PUB, BOOK_TITLE, AUTHOR_NAME, TOTAL_SOLD_COPIES)

*Marks: 2* **MCQ**

| BookDetails | | | | |
|---|---|---|---|---|
| BOOK_ID | YEAR_PUB | BOOK_TITLE | AUTHOR_NAME | TOTAL_SOLD_COPIES |
| 1001 | 2010 | DBMS | KORTH | 15000 |
| 1002 | 2010 | OPERATING SYSTEM | GALVIN | 25000 |
| 1003 | 2020 | COMPILER | ULLMAN | 12000 |
| 1004 | 1995 | DATA STRUCTURES | CORMEN | 20000 |
| 1005 | 1995 | DATA STRUCTURES | ULLMAN | 20000 |
| 1006 | 1990 | COMPUTER NETWORKS | FOROUZAN | 15000 |
| 1007 | 2010 | MACHINE LEARNING | MITCHELL | 18000 |

Identify the correct SQL command to find the BOOK_ID with BOOK_TITLE of BookDetails whose AUTHOR_NAME is neither 'ULLMAN' nor 'KORTH' and have TOTAL_SOLD_COPIES is atleast 18000.

a)
```
SELECT BOOK_ID, BOOK_TITLE
FROM BookDetails
WHERE AUTHOR_NAME NOT BETWEEN ('ULLMAN', 'KORTH')
AND (TOTAL_SOLD_COPIES > 18000);
```

b)
```
SELECT BOOK_ID, BOOK_TITLE
FROM BookDetails
WHERE AUTHOR_NAME NOT IN ('ULLMAN', 'KORTH')
AND (TOTAL_SOLD_COPIES >= 18000);
```

c)
```
SELECT BOOK_ID, BOOK_TITLE
FROM BookDetails
WHERE AUTHOR_NAME NOT AS ('ULLMAN', 'KORTH')
AND (TOTAL_SOLD_COPIES >= 18000);
```

d)
```
SELECT BOOK_ID, BOOK_TITLE
FROM BookDetails
WHERE AUTHOR_NAME NOT IN ('ULLMAN', 'KORTH')
AND (TOTAL_SOLD_COPIES > 18000);
```

**Answer**: b)

**Explanation:** BETWEEN operator selects values within a given range. The values can be text, date, or numbers.
IN operator allows you to easily test if the expression matches any value in the list of values. It is used to remove the need of multiple OR condition.
Hence, we have used NOT IN to exclude that matches and have TOTAL_SOLD_COPIES atleast 18000 means TOTAL_SOLD_COPIES >= 18000.
Hence, option b) is correct and the rest are incorrect.

## Question 10

Consider the following relation schema:
`BookDetails(BOOK_ID, YEAR_PUB, BOOK_TITLE, AUTHOR_NAME, TOTAL_SOLD_COPIES).`

*Marks: 2* **MCQ**

Identify the correct statement to create an `INDEX` on `BOOK_ID` and `BOOK_TITLE` of `BookDetails` relation named as '`Books`'.

a)
```
Create INDEX Books
TO BookDetails(BOOK_ID, BOOK_TITLE);
```

b)
```
Create INDEX Books
ON BookDetails(BOOK_ID, BOOK_TITLE);
```

c)
```
Create INDEX Books
AS BookDetails(BOOK_ID, BOOK_TITLE);
```

d)
```
Create INDEX Books
OF BookDetails(BOOK_ID, BOOK_TITLE);
```

**Answer**: b)
**Explanation:** The general syntax to create any `INDEX` is

```
CREATE INDEX indexname
ON tablename (column1, column2, ...);
```

So, option (b) is correct.