BSCCS2001: Practice Assignment with Solutions
Week 1

1. A programmer is working on the data structures that store data internally in a database. At what level of abstraction is the programmer working?

   ◯ Logical Level

   ◯ View Level

   ◯ Programming Level

   √ Physical Level

Solution:
Logical Level represents the relational model and conceptual schema of the data.
View Level represents the user's view of the data i.e how the user sees the data

Physical level deals with the actual storage structure of the data internally in the form of trees and other data structures.

Programming level is not a level of abstraction.

2. From among the given types of applications, choose the ones for which DBMS will be a preferred choice over filesystems.

   √ Applications with large datasets.

   √ Applications with concurrent transactions.

   ◯ Applications with small datasets.

   ◯ Applications with no dedicated database administrators.

Solution:
For applications with small datasets, the overhead in installing the DBMS will be much more than the advantage obtained due to reduced retrieval time.
If there is no dedicated personnel for maintaining a database, the performance will begin to deteriorate after a period of time.

3. By the concept of Logical Data Independence, a change in the logical level of DBMS should not affect which other level(s) of abstraction?

   √ View Level

   ◯ Physical Level

◯ Both Physical and View Level

◯ None of the above

Solution:
Logical Data Independence: A change in Logical Level of DBMS should not affect the View Level.

4. By the concept of Physical Data Independence, a change in the physical level of DBMS should not affect which other level(s) of abstraction?

◯ View Level

◯ Logical Level

√ Both Logical and View Level

◯ None of the above

Solution:
Physical Data Independence: A change in Physical Level of DBMS should not affect either the View Level or the Logical Level.

5. Which model is widely used during the planning and designing phase of a database system?

◯ View Model

√ Entity-Relationship Model

◯ Object Model

◯ Relational Model

Solution: Entity-Relationship Model is used to define a high level view of the data entities and the relationships between them. It is used mainly for designing and planning the database structure.

6. Which of the following are advantages of DBMS over file based data management applications?

√ Easy recovery of data

√ Consistency of data

√ Efficiency of operation

◯ Ease of initial setup

Solution: Initial system setup is costly in the case of DBMS, whereas in file based systems, it is relatively easy and economical.

7. Which among the following is a good option for exchanging data among different systems over the internet?

   ◯ HTML
   ◯ MS Access
   ◯ SQL
   √ XML

Solution: HTML is a markup language used mainly for data presentation, whereas XML is widely used for sharing data over different systems over the web. The other two are not meant for information exchange over the internet.

8. Planning what attributes should be placed in which table of a database is a part of

   ◯ Attribute Design
   √ Logical Design
   ◯ Physical Design
   ◯ Subsystem Design

Solution: The design of structure of tables is a part of logical design.

9. Which of the following options are examples of data transactions?

   √ Transferring funds using e-wallets
   √ Booking a reservation in railways
   ◯ Increasing the data storage capacity
   √ Updating KYC in bank.

Solution: Any event which modifies the current state (value) of the data is a data transaction.
Increasing storage capacity does not make any changes in the current data.

10. Which data model aptly satisfies the need of a system which maintains large set of complex interconnected data, where the semantics of interconnection changes dynamically (like a social networking site)?

◯ Relational Model

◯ XML Model

◯ Object-Relational Model

√ Graph Model

Solution: Refer Lecture 1.4.

BSCCS2001: Practice Assignment with Solutions
Week 2

Modules covered:

1. Attribute Types, Relation Schema and Instance, Keys, Relational Query Languages
2. Operations, Select, Project, Union, Difference, Intersection, Cartesian Product
3. Natural Join, Aggregate Operations
4. Introduction to SQL - History of SQL, Data Definition Language (DDL), Basic Query Structure (DML)
5. Additional Basic Operations, Set Operations, Null Values, Aggregate Functions

1. Consider the table **taskAssignment** (in Figure 1) which represents tasks assigned to each employee for a given day. [MCQ: 2 points]

taskAssignment

employee_num	task_num	task_duration	date_of_assignment	supervisor_num	location
101	P103	7	10-01-2020	112	Block-C
102	P103	5	10-01-2020	112	Block-C
101	P103	4	11-01-2020	112	Block-C
101	P103	6	12-01-2020	112	Block-C
104	P102	6	10-01-2020	111	Block-B
105	P101	7	10-01-2020	110	Block-A
104	P101	7	11-01-2020	110	Block-A
105	P101	6	11-01-2020	110	Block-A
102	P102	6	11-01-2020	111	Block-B

Figure 1: Table **taskAssignment**

Select the appropriate compound key for the table.

- ◯ { *employee_num* }
- ◯ { *employee_num*, *task_num* }
- √ { *employee_num*, *task_num*, *date_of_assignment* }
- ◯ { *employee_num*, *supervisor_num* }

Solution: A compound key is a set of more than one attribute which uniquely identifies all rows of a relation.
The entries for { *employee_num* } are not unique for all the tuples. It is neither a set of more than one attribute nor uniquely identifies all rows of the given relation. Thus, it cannot be a compound key.
The entries for { *employee_num*, *task_num* } are not unique for all the tuples. Thus, it cannot be a compound key.
The entries for {*employee_num*, *task_num*, *date_of_assignment*} are unique for all the rows. So, **it is a valid candidate key for the given relation**.
The entries for { *employee_num*, *supervisor_num* } are not unique for all the tuples. Thus, it cannot be a compound key.

2. Consider the following relational schema on students of a school. [MCQ: 2 points]
studentInfo *(enrollment_num, class, section, roll, name).*
{*enrollment_num*} and {*class, section, roll*} are two possible candidate keys. What is the maximum number of possible superkeys of **studentInfo**?

- ◯ 16
- √ 18
- ◯ 20
- ◯ 22

Solution: This question will be discussed in the *Solve with the Instructor* session.

3. Consider the tables **vendor** and **component** as shown in Figure 2. The table **component** has attribute *vendor_num* which is a foreign key that refers to table **vendor(*vendor_num*)**.

vendor

vendor_num	vendor_name	vendor_location
10	YADAV	CHENNAI
11	AKHTAR	KOLKATA
12	PRASAD	TRICHY
13	SHARMA	BENGALURU

component

item_num	name	cost	vendor_num
1011	RAM	2500.00	11
1012	CPU	8000.50	12
1013	MONITOR	5000.00	10
1014	KEYBOARD	500.50	13
1013	MONITOR	2250.00	13
1014	KEYBOARD	450.50	11
1011	RAM	3300.00	10

Figure 2: Tables **vendor** and **component**

Identify the appropriate "CREATE TABLE" statement for table **component**. [MCQ: 2 points]

◯
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num),
FOREIGN KEY (vendor_num) REFERENCES vendor(vendor_num));
```

◯
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num, name),
FOREIGN KEY (vendor_num) REFERENCES vendor(vendor_num));
```

◯
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num, vendor_num),
FOREIGN KEY (vendor_num) REFERENCES vendor(item_num, vendor_num));
```

√
```
CREATE TABLE component(
item_num int NOT NULL,
name varchar(20),
```

```
cost numeric(6, 2) NOT NULL,
vendor_num int NOT NULL,
PRIMARY KEY (item_num, vendor_num),
FOREIGN KEY (vendor_num) REFERENCES vendor(vendor_num));
```

Solution:
Option 1: since the table **component** has duplicate values in the column *item_num*, the attribute set {*item_num*} cannot be a primary key. Hence, option 1 is incorrect.
Option 2: since table **component** has duplicate values in columns {*item_num, name*}, the attribute set {*item_num, name*} cannot be a primary key. Hence, option 2 is incorrect.
Option 3: since the number of columns in the foreign key does not match the number of columns in the referenced table, option 3 is incorrect.
Option 4: since table **component** has unique entries in columns {*item_num, vendor_num*}, attribute set {*item_num, vendor_num*} is the proper primary key and hence, option 4 is the correct option.

4. Identify the appropriate "ALTER TABLE" statement for table **vendor** such that we can add a column named *vendor_phone* to store the phone number of the vendors. [MSQ: 2 points]

◯ `ALTER TABLE vendor APPEND COLUMN vendor_phone numeric(10);`

√ `ALTER TABLE vendor ADD COLUMN vendor_phone numeric(10);`

√ `ALTER TABLE vendor ADD vendor_phone numeric(10);`

◯ `ALTER TABLE vendor INSERT COLUMN vendor_phone numeric(10);`

Solution: To add a new column in the given table, we use **"ADD"** or **"ADD COLUMN"** commands. Thus, option 2 and option 3 are correct.

5. Identify the correct statement(s) about a Foreign Key. [MSQ: 2 points]

- ◯ A FOREIGN KEY is a data element/attribute within a data field of a data record that is not unique, and cannot be used to distinguish one data record in a database from another data record within a database table.
- ◯ A FOREIGN KEY constraint prevents data from being inserted into the foreign key column.
- √ A FOREIGN KEY is a data element/attribute within a data field of a data record within a database table that refers to either a primary key or an attribute with unique constraint.
- √ A FOREIGN KEY can be added at the time of ALTER TABLE query.

Solution: Option 1 - A FOREIGN KEY is a data element/attribute within a data field of a data record that is unique, and this can be used to distinguish one data record in a database from another data record within a database table.
Option 2 - The FOREIGN KEY constraint prevents invalid data from being inserted into the foreign key column.
Option 3, 4 are factual statements about Foreign Key.

6. Consider the table **Employee** given in Figure 3.

Employee

ID	Name	Salary
1	MIKE	35
2	KYLE	50
3	JAMES	50
4	JONES	NULL
5	LIMA	70
6	PACY	50

Figure 3: Table Employee

Determine the correct relation based on the three queries given below: [MCQ: 2 points]

- `SELECT COUNT(DISTINCT Salary) as A FROM Employee;`
- `SELECT COUNT(*) as B FROM Employee;`
- `SELECT COUNT(Salary) as C FROM Employee;`

◯ value of B>A and B=C

◯ value of A>B and B>C

◯ value of A>B and B=C

√ value of B>A and B>C

Solution: This question will be discussed in the *Solve with the Instructor* session.

7. Consider the table **Employee** given in Figure 4.

Employee

ID	Name	Salary
1	MIKE	35
2	KYLE	50
3	JAMES	50
4	JONES	NULL
5	LIMA	70
6	PACY	50

Figure 4: Table Employee

Determine the suitable query that returns the following table: [MCQ: 2 points]

AVG_SAL
55

◯
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND ID>5;
```

◯
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND Salary>50;
```

◯
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND ID>3;
```

√
```
SELECT AVG(Salary) AS AVG_SAL
FROM Employee WHERE Salary IS NOT NULL AND Salary>35;
```

Solution: From the table **Employee**, NOT NULL values that are greater than 35 are 50, 50, 70, 50. The average of 50, 50, 70, 50 is 55.

8. Identify the output for the following SQL statement. [MCQ: 2 points]

```
SELECT max(temperature) - min(temperature)
FROM weatherReport
WHERE state='Karnataka';
```

- ◯ 4
- √ 5
- ◯ 2
- ◯ 0

Solution: The part of the statement:

```
WHERE state='Karnataka';
```

extracts all rows having *state* as "Karnataka".
`max(temperature)` returns the maximum *temperature* from among the given cities in "Karnataka", i.e. 36 (for "Bellary").
`min(temperature)` returns the minimum *temperature* from among the given cities in "Karnataka", i.e. 31 (for "Bengaluru").
Therefore, the output is 36 - 31 = 5;

9. Using the table **Citizen** given in Figure 5, answer the question that follows.

Citizen

ID	profession	lastname	firstname
23	clerk	Holmes	Mark
45	firefighter	Singh	Vikram
23	police	Samson	Rana
31	clerk	Butler	Jones
67	gardener	Holmes	John

Figure 5: Table Citizen

Identify the correct SQL statement(s) to create the given table. [MSQ: 2 points]

◯
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (ID));
```

√
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (ID, lastname));
```

√
```
CREATE TABLE Citizen (ID int,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (firstname, lastname));
```

√
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (ID, profession));
```

◯
```
CREATE TABLE Citizen (ID int NOT NULL,
profession varchar(255), lastname varchar(255) NOT NULL,
firstname varchar(255), PRIMARY KEY (profession));
```

Solution: Only an attribute, or a set of attributes that can uniquely identify a row, can be chosen as the PRIMARY KEY. Here (ID, lastname), (firstname, lastname) and (ID, profession) can do so. Hence, options 2, 3, 4 are correct.

10. Using the table **Citizen** given in Figure 6, answer the question that follows.

Citizen

ID	profession	lastname	firstname
23	clerk	Holmes	Mark
45	firefighter	Singh	Vikram
23	police	Samson	Rana
31	clerk	Butler	Jones
67	gardener	Holmes	John

Figure 6: Table Citizen

Let the table **Citizen** be created using the SQL statement given below: [MSQ: 2 points]

- `CREATE TABLE Citizen (ID int, profession varchar(255), lastname varchar(255) NOT NULL, firstname varchar(255));`

Identify the correct INSERT INTO statement for this table.

√ `INSERT INTO Citizen (ID, profession, lastname, firstname) VALUES (23,'clerk', 'Holmes', 'Mark');`

◯ `INSERT INTO Citizen TABLE VALUES (23,'clerk', 'Holmes', 'Mark');`

√ `INSERT INTO Citizen VALUES (23,'clerk', 'Holmes', 'Mark');`

◯ `INSERT INTO Citizen VALUES ('23','clerk', 'Holmes', 'Mark');`

Solution: To insert values in a table, the generic format is -
INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);
OR
INSERT INTO table_name VALUES (value1, value2, ...);
Also, varchar type attributes need to be in ' ' and INT type attributes without ' '.

11. Using the table **Citizen** given in Figure 7, answer the question that follows.

Citizen

ID	profession	lastname	firstname
23	clerk	Holmes	Mark
45	firefighter	Singh	Vikram
23	police	Samson	Rana
31	clerk	Butler	Jones
67	gardener	Holmes	John

Figure 7: Table Citizen

Let the table **Citizen** be created with *firstname* as the primary key, and the values be inserted as per the schema in the table. [MCQ: 1 points]

Choose the SQL statement to remove the primary key constraint Citizen_pkey of the table **Citizen**.

◯ `ALTER TABLE Citizen`
`DROP Citizen CONSTRAINT Citizen_pkey;`

◯ `MODIFY TABLE Citizen`
`DROP CONSTRAINT Citizen_pkey;`

◯ `MODIFY TABLE Citizen`
`DROP Citizen CONSTRAINT;`

√ `ALTER TABLE Citizen`
`DROP CONSTRAINT Citizen_pkey;`

Solution: The syntax to remove an existing primary key constraint is -
ALTER TABLE table_name DROP CONSTRAINT primary_key_constraint_name;
Hence option 4 is correct.
ALTER TABLE Citizen DROP CONSTRAINT Citizen_pkey;

12. Let $\{sup\_num\}$ be the primary key of table **suppliers** and $\{part\_num, sup\_num\}$ be the primary key of table **parts**. [NAT: 2 points]
Consider the SQL query given below:

```
SELECT s.sup_num, sum(p.part_qty)
FROM suppliers s, parts p WHERE s.sup_num = p.sup_num
GROUP BY s.sup_num
HAVING SUM(p.part_qty) > 70
```

How many rows will be returned by the above SQL query?
Answer: 2

Solution: As per the given SQL statement, it first performs a Cartesian product between **suppliers** and **parts**, which output all possible combinations from both the tables.
The part of the statement:
`s.sup_num = p.sup_num`
eliminates the rows which do not satisfy the condition. The output is as shown below:

s.sup_num	s.sup_name	p.part_num	p.sup_num	p.part_qty
1001	Able	301	1001	32
1001	Able	302	1001	16
1002	Peter	301	1002	41
1002	Peter	302	1002	11
1002	Peter	304	1002	35
1003	Molina	302	1003	36
1003	Molina	304	1003	40
1004	Nikki	301	1004	17
1004	Nikki	303	1004	25

The part of the statement:
`SELECT s.sup_num, sum(p.part_qty) ...GROUP BY s.sup_num`
results in:

s.sup_num	SUM(p.part_qty)
1001	48
1002	87
1003	76
1004	42

Finally, the part of the statement: `HAVING SUM(p.part_qty) > 70`
results in:

s.sup_num	SUM(p.part_qty)
1002	87
1003	76

Thus, the result has 2 rows.

BSCCS2001: Practice Assignment with Solutions
Week 3

1. Consider the relational schema given in Figure 1.

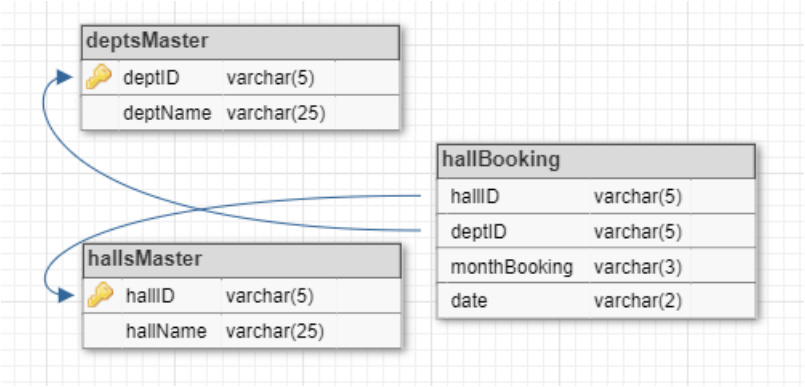


Figure 1: Hall Booking Relational Schema

Find the names of departments that have booked all the halls at least once in the month of January. [MCQ: 2 points]

◯
```
SELECT deptName FROM deptsMaster
WHERE deptID IN
        (SELECT DISTINCT deptID FROM hallBooking AS hb1
        WHERE NOT EXISTS
                (SELECT hm.hallID FROM hallsMaster AS hm
                INTERSECT
                SELECT hb2.hallID FROM hallBooking AS hb2
                WHERE hb1.deptID = hb2.deptID
                AND monthBooking = 'Jan'));
```

√
```
SELECT deptName FROM deptsMaster
WHERE deptID IN
        (SELECT DISTINCT deptID FROM hallBooking AS hb1
        WHERE NOT EXISTS
                (SELECT hm.hallID FROM hallsMaster AS hm
                EXCEPT
                SELECT hb2.hallID FROM hallBooking AS hb2
                WHERE hb1.deptID = hb2.deptID
                AND monthBooking = 'Jan'));
```

◯
```
SELECT deptName FROM deptsMaster
WHERE deptID IN
        (SELECT DISTINCT deptID FROM hallBooking AS hb1
        WHERE EXISTS
```

```
                    (SELECT hm.hallID FROM hallsMaster AS hm
                    EXCEPT
                    SELECT hb2.hallID FROM hallBooking AS hb2
                    WHERE hb1.deptID = hb2.deptID
                    AND monthBooking = 'Jan'));
```

◯
```
SELECT deptName FROM deptsMaster
WHERE deptID IN
        (SELECT DISTINCT deptID FROM hallBooking AS hb1
        WHERE EXISTS
                (SELECT hm.hallID FROM hallsMaster AS hm
                INTERSECT
                SELECT hb2.hallID FROM hallBooking AS hb2
                WHERE hb1.deptID = hb2.deptID
                AND monthBooking = 'Jan'));
```

Solution:

```
SELECT hm.hallID FROM hallsMaster AS hm
                EXCEPT
                SELECT hb2.hallID FROM hallBooking AS hb2
                WHERE hb1.deptID = hb2.deptID
                AND monthBooking = 'Jan'
```

The above query fetches all hallIDs that have not been booked in January.

```
SELECT DISTINCT deptID FROM hallBooking AS hb1
        WHERE NOT EXISTS (hallIDs that have not been booked in January)
```

The above query will retrieve all department IDs that have not booked any halls that have not been booked in January.

```
SELECT deptName FROM deptsMaster
WHERE deptID IN (all department IDs that have not booked any halls
that have not been booked in January)
```

The above query fetches the names of departments that have not booked any halls that have not been booked in January.
Note that if we execute the nested queries directly on the sql prompt, they will give errors due to the aliases used.

2. Consider the relational schema given in Figure 2.

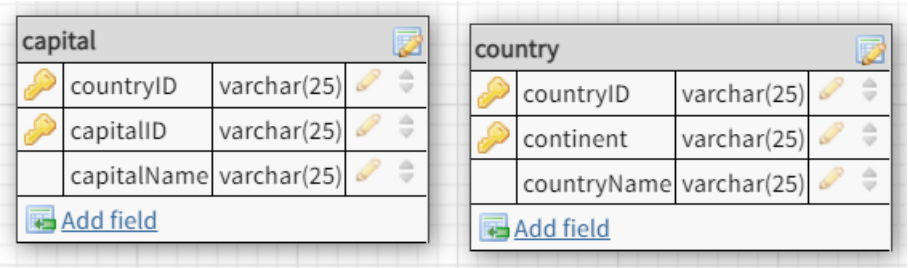


Figure 2: Country Capitals Relational Schema

What should be filled in the blank so that the following query will return the capitals of all countries that belong to Asia but not Europe? (Write the answer as a single word in all CAPS) [NAT: 2 points]

```
SELECT capitalName FROM capital
WHERE countryID IN (SELECT countryID FROM country
                    WHERE continent ='Asia'
                    -----------------
                    SELECT countryID FROM country
                    WHERE continent ='Europe');
```

Answer: EXCEPT

Solution: The first part of the inner query returns all countries that belong to Asia. If we need to find countries that belong to Asia but not Europe, then from the rows returned by the first part of the inner query, we have to remove those that contain countries that belong to Europe as well. Hence, to remove those rows, we use EXCEPT.

3. Based on the relations given in Figure 3 answer the question that follows.

employee

empID	empName	deptID	desgID
E00001	Akash	D0002	G0001
E00002	Akshay	D0002	----(a)----
E00003	Subha	D0003	G0003
E00004	Lavanya	---(b)---	G0002
E00005	Diya	D0001	G0001

department

deptID	deptName
D0001	Purchase
D0002	Sales
D0003	Accounts

designation

desgID	desgName	Salary
G0001	Clerk	5000
G0002	Supervisor	7000
G0003	Manager	10000

Figure 3: Employee instance

What should be filled in blank (a) in the table `employee` in Figure 3, if the query given below returns the value: Akshay? [NAT: 2 points]

```
SELECT empName FROM employee
WHERE desgID LIKE '%2' AND deptID LIKE '%2';
```

Answer: G0002

Solution: Since the value returned is Akshay, it corresponds to second row of table `employee`. desgID in employee table is a foreign key that references department table, hence the desgID can only be any one value from {G0001, G0002, G0003}. The `WHERE` condition specifies `desgID LIKE '%2'`. It follows from all three reasons that the only possible value that can be filled in blank (a) is G0002.

4. Consider the following SQL statement: [MSQ: 2 points]

```
CREATE TABLE boats(
    boatID VARCHAR (8),
    boatName VARCHAR (20),
    boatColour VARCHAR (8),
    yearOfPurchase INTEGER,
    weight INTEGER,
    PRIMARY KEY (boatID),
    CHECK (boatColour IN ('Black', 'White', 'Red', 'Yellow')));
```

Which among the following will cause an integrity constraint violation in the `boats` table?

- ◯ `INSERT INTO boats('B1', 'Liberty', 'Red', 2003, 500);`
- √ `INSERT INTO boats('B1', 'Liberty', 'Blue', 2003, 500);`
- √ `UPDATE boats SET boatColour = 'Green' WHERE boatID = 'B1';`
- ◯ `DELETE FROM boats;`

Solution: In option 1, there is no constraint violation.
In option 2, since the permitted colors do not include blue, it will cause a violation.
In option 3, the permitted colors do not include green and hence it will cause a violation.
In option 4, there is no constraint violation.

5. Consider the relational schema given in Figure 4.

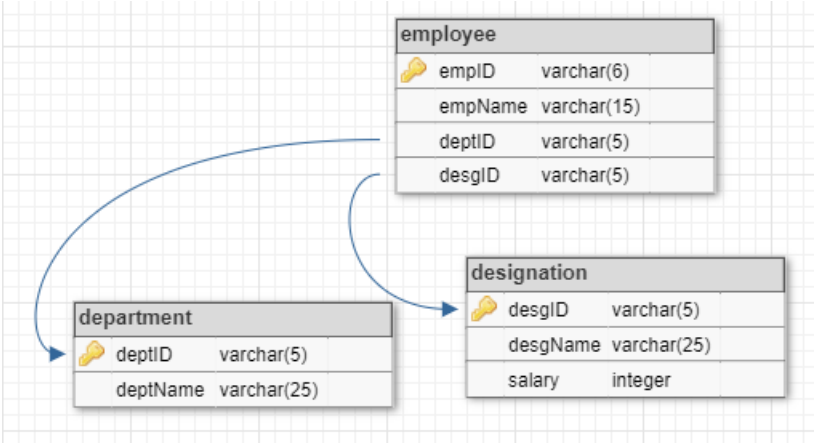


Figure 4: Employee Schema

If the relations **employee, designation** and **department** have 100, 6, 5 rows respectively, what is the difference between the maximum and the minimum number of rows returned by the following query? [NAT: 2 points]

```
SELECT * FROM employee LEFT OUTER JOIN designation
ON employee.desgID = designation.desgID;
```

Answer: 0

Solution: Left outer join (also known as Left join) returns all tuples returned by natural join along with those tuples in the left table (here, `employee`) that does not have matching entry in the right table. In the given question, however, `desgID` is the foreign key in Table `employee` that references Table `designation`. Therefore, there will not be any tuple in the left table that does not have a matching entry in the right table. Thus, the maximum number of rows returned by the left join in the given example is 100.
The case when `employee` table has no rows is the case when left outer join will have the minimum number of rows. In this case, however, the `employee` table has 100 rows. So, there will be at least 100 rows returned by the left join.
The answer is $100 - 100 = 0$.

6. Choose the appropriate query/queries to find the names of batsmen who scored the second-highest runs. [MSQ: 2 points]

√
```
SELECT name, MAX(runs) AS runs
FROM batsman WHERE runs < (SELECT MAX(runs) FROM batsman);
```

√
```
SELECT name, MAX(runs) AS runs
FROM batsman WHERE runs IN
(SELECT runs FROM batsman MINUS (SELECT MAX(runs) FROM batsman));
```

√
```
SELECT name, runs AS runs
FROM batsman WHERE runs = (SELECT runs FROM batsman
ORDER BY runs LIMIT 1,1);
```

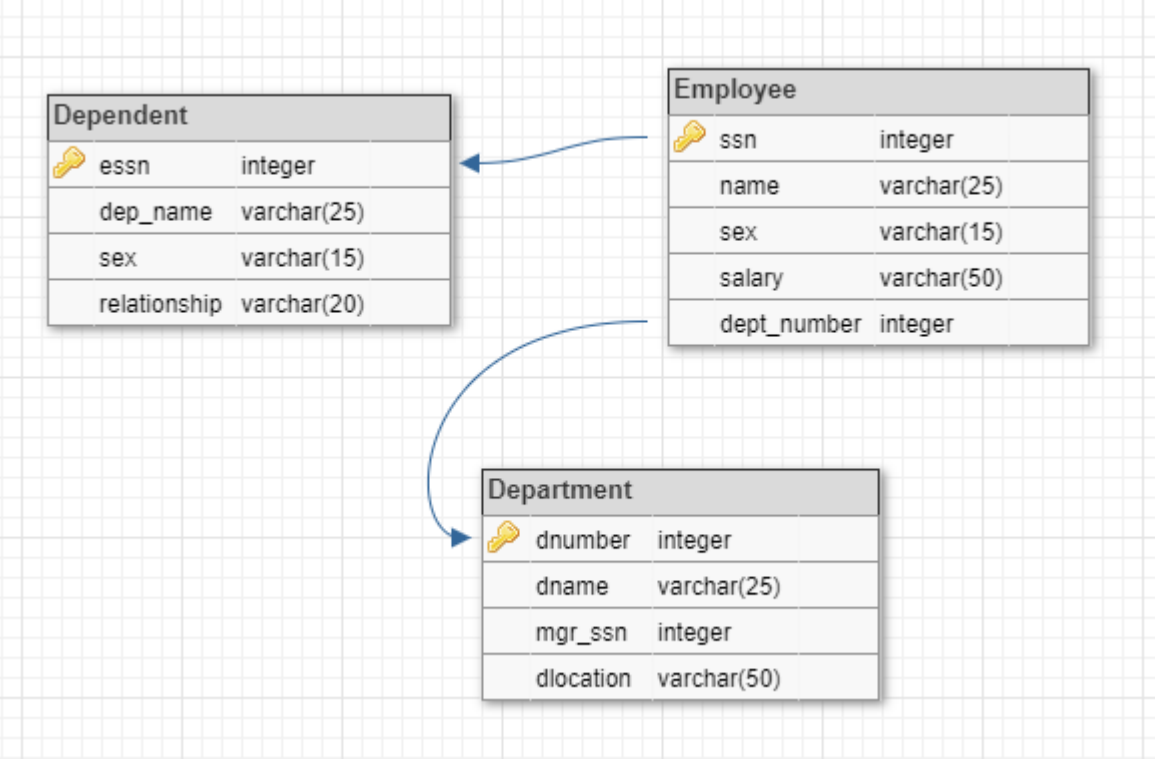
◯
```
SELECT name, MAX(runs) AS runs FROM batsman
WHERE runs > (SELECT MIN(runs) FROM batsman);
```

Solution:

- MAX, MIN functions are used to find out the record with maximum and minimum values respectively among a record set.
- The SQL MINUS operator is used to return all rows in the first SELECT statement that are not returned by the second SELECT statement.
- The LIMIT statement is used to limit the number of records returned based on a limit value.
- The ORDER BY keyword is used to sort the result-set in ascending or descending order.

Option 1 - The inner query will fetch the maximum runs and then the outer query will return the runs value which is maximum among all and lesser than the value retrieved by the inner query. Hence, the second-highest value of runs is fetched.
Option 2 - The inner query will fetch all the runs values other than the maximum runs and from this set, the IN operator will retrieve the maximum value. Hence, the second-highest value of runs is fetched.
Option 3 - The inner query will fetch the second-highest value of runs using the Limit operator, then using the '=' the outer query will retrieve it.
Option 4 - The inner query will return the minimum runs and the outer query will fetch the maximum runs greater than the runs value of the inner query. Hence it is incorrect.

Consider the table **Employee**, table **Department** and table **Dependent**, and answer the questions 7 and 8.



7. Select the suitable query to retrieve the names of employees who have no dependents. [MCQ: 2 points]

◯ `SELECT name FROM Employee WHERE NOT EXISTS (SELECT * FROM Dependent AS D WHERE D.ssn = essn);`

◯ `SELECT name FROM Dependent WHERE NOT EXISTS (SELECT * FROM Employee WHERE ssn = essn);`

√ `SELECT name FROM Employee WHERE NOT EXISTS (SELECT * FROM Dependent WHERE ssn = essn);`

◯ `SELECT name FROM Employee WHERE IN (SELECT * FROM Employee WHERE ssn = essn);`

Solution: The EXISTS/NOT EXISTS condition in SQL is used to check whether the result of a correlated nested query is empty (contains no tuples) or not.
As per the question, to retrieve the names of employees who have no dependents, the outer query needs to fetch data from the table Employee and the inner query needs

to fetch data from the table Dependent. Hence, options 2 and 4 are incorrect.

In option 1- After aliasing Dependent as D, the condition must be ssn = D.essn. Hence, option 1 is incorrect.

In option 3 - Inner query will fetch all the dependents where attribute ssn of Employee is matched with essn from Dependent. Hence, only if there is no matched value, NOT EXISTS will be true and the names of the employees who have no dependents will be retrieved.

8. Select the suitable query to retrieve the names of employees who have some dependent(s) whose name ends with 'KUMAR'. [MCQ: 2 points]

◯
```
SELECT name FROM Dependent
WHERE ssn IN (SELECT essn FROM Employee
WHERE dep_name LIKE '%KUMAR');
```

◯
```
SELECT name FROM Employee
WHERE essn IN (SELECT ssn FROM Dependent
WHERE dep_name LIKE '%KUMAR%');
```

√
```
SELECT name FROM Employee
WHERE ssn IN (SELECT essn FROM Dependent
WHERE dep_name LIKE '%KUMAR');
```

◯
```
SELECT name FROM Employee
WHERE ssn IN (SELECT essn FROM Dependent
WHERE dep_name LIKE 'KUMAR');
```

Solution: The LIKE operator is used in a WHERE clause to search for a specified pattern in a column. The percent sign (%) represents zero, one, or multiple characters and the underscore sign (_) represents one, single character.

So to retrieve all the names ending with KUMAR, it has to match '%KUMAR'. Hence, options 2 and 4 are incorrect.

In option 1, the inner query needs to fetch from table Dependent and outer query from table Employee. Hence, incorrect.

In option 3, the inner query will fetch the Dependent(s) whose name ends with KUMAR and using IN keyword, the outer query will retrieve the names of the corresponding employees. Hence, correct.

9. Consider the table **employee** and table **department** as shown in Figure 5, and answer the question that follows. [MCQ: 2 points]

employee

emp_name	emp_id	age	dept_id
WADE	1	23	10
MADDEN	4	54	10
HARM	6	34	13
TALLY	3	41	16
RODEY	2	46	14
JONES	7	38	14
MULE	5	49	16

department

dept_name	dept_id	dept_location
MATHS	10	Houston
ENGLISH	15	San Antonio
PHYSICS	14	Houston
COMPUTER	13	New York
CHEMISTRY	16	Chicago

Figure 5: employee & department

What will be the output of the following query?

```
SELECT emp_id, dept_name
FROM employee NATURAL JOIN department
ORDER BY age desc;
```

√ Output:

emp_id	dept_name
4	MATHS
5	CHEMISTRY
2	PHYSICS
3	CHEMISTRY
7	PHYSICS
6	COMPUTER
1	MATHS

◯ Output:

emp_id	dept_name
1	MATHS
4	MATHS
6	COMPUTER
3	CHEMISTRY
2	PHYSICS
7	PHYSICS
5	CHEMISTRY

◯ Output:

emp_id	dept_name
4	MATHS
3	CHEMISTRY
2	PHYSICS
5	CHEMISTRY
7	PHYSICS
6	COMPUTER
1	MATHS

◯ Output:

emp_id	dept_name
4	MATHS
3	CHEMISTRY
7	PHYSICS
5	CHEMISTRY
2	PHYSICS
6	COMPUTER
1	MATHS

Solution: As per the query, after NATURAL JOIN on employee table and department table, the resultant table will be -

emp_id	dept_name
1	MATHS
4	MATHS
6	COMPUTER
3	CHEMISTRY
2	PHYSICS
7	PHYSICS
5	CHEMISTRY

And as and when we put ORDER BY age in descending order, we will fetch the following resultant table -

emp_id	dept_name
4	MATHS
5	CHEMISTRY
2	PHYSICS
3	CHEMISTRY
7	PHYSICS
6	COMPUTER
1	MATHS

10. Consider a table **Employee**(*eid, edept, ename, esalary, ebonus*). The table has no records initially.

[MCQ:2 points]

```
CREATE OR REPLACE FUNCTION bonus_fun() RETURNS TRIGGER AS $$
    BEGIN
         IF NEW.edept = 'R/D' THEN
               NEW.ebonus = NEW.esalary * .75;
         END IF;
         RETURN NEW;
    END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER bonus_trig
    BEFORE INSERT ON Employee
    EXECUTE PROCEDURE bonus_fun();

INSERT INTO Employee VALUES (4,'R/D','Diksha',30000);
```

```
INSERT INTO Employee VALUES (2,'Accounts','Raj',40000);
SELECT ebonus FROM Employee;
```

If the given code is executed, then what will be the output?

- ◯ 22500
  0
- ◯ 22500
  NULL
- ◯ 22500
  30000
- √ The code has errors.

Solution: The code is erroneous because the trigger definition does not explicitly mention its granularity (*for each row* or *for each statement*). This trigger checks each insertion and modifies the value of an attribute (*ebonus*) when the described condition satisfies, therefore it should work as a row level trigger.

11. Consider an instance of the table **Employee** given below.

[MCQ:2 points]

eid [PK] integer	edept character varying	ename character varying	esalary integer
1	Accounts	Rekha	35000
2	HR	Joseph	30000
3	HR	Arif	50000
4	Development	Debraj	45000
5	Accounts	Abhijit	90000
6	Marketing	Shahid	76000
7	Sales	Shabana	25000
8	Marketing	Meenakshi	42000
9	Sales	Digvijay	66000
10	Marketing	Shashi	54000

Figure 6: Table: Employee

If the given code is executed on this instance, then what will be the output/error?

[MCQ:2 points]

```
CREATE OR REPLACE FUNCTION salary_fun() RETURNS TRIGGER AS $$
DECLARE
    counter INT := 0;
BEGIN
    IF  NEW.esalary > 75000 THEN
            counter = counter + 1;
            RAISE NOTICE 'Number of affected rows : %', counter;
            --//This statement prints => NOTICE: <whatever follows>
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER salary_trig
    AFTER UPDATE ON Employee
    FOR EACH ROW
    EXECUTE PROCEDURE salary_fun();

UPDATE Employee SET esalary = esalary * 1.5;
```

◯ NOTICE: Number of affected rows : 4

◯ NOTICE: Number of affected rows : 10

◯ NOTICE: Number of affected rows : 4
NOTICE: Number of affected rows : 4
NOTICE: Number of affected rows : 4
NOTICE: Number of affected rows : 4

√ None of the above

Solution: The trigger will be executed on every row affected by the `UPDATE` statement. Only 4 rows in the given instance will have their new salary more than 75000. The `RAISE NOTICE` statement is inside the `IF` clause, thus it will be executed 4 times. Observe the fact that the trigger fires for each updated row, and hence every time the variable *counter* is reinitialized with 0. It will be incremented to 1 with respect to that specific row, and thus we will get the output as:
NOTICE: Number of affected rows : 1
NOTICE: Number of affected rows : 1
NOTICE: Number of affected rows : 1
NOTICE: Number of affected rows : 1

12. If we want to store/print the number of affected rows when an update or delete statement is executed, then which type of trigger should we use to count?

[MCQ:2 points]

√ Statement level trigger

◯ Row level trigger

◯ Both are equally efficient

◯ Table level trigger

Solution: If we want to count the number of affected rows then we need not execute a trigger every time for each row. After all the modifications are over, a single execution of a trigger to count the affected rows should be done. Running a row level trigger will simply do the same job again and again for all the affected rows. Hence, Statement level triggers should be used here.

BSCCS2001: Practice Assignment with Solutions
Week 4

1. Consider the following relations : [MCQ: 2 points]
   A = (P, Q, R)
   B = (X, Y, Z)
   Let relations $a(A)$ and $b(B)$ be given. Which of the following expressions in the tuple relational calculus is equivalent to $\Pi_{P,Z}(\sigma_{R=X}(a \times b))$?

   - √ $\{t \mid \exists p \in a, \exists q \in b(t[P] = p[P] \wedge t[Z] = q[Z] \wedge p[R] = q[X])\}$
   - ◯ $\{t \mid \exists p \in a, \exists q \in b(t[P] = p[P] \wedge t[Z] = q[Z] \vee p[R] = q[X])\}$
   - ◯ $\{t \mid \exists p \in a, \exists q \in b(t[P] = p[P] \wedge t[Z] = q[Z] \wedge p[X] = q[R])\}$
   - ◯ $\{t \mid \exists p \in a, \exists q \in b(t[Z] = p[Z] \wedge t[P] = q[P] \wedge p[R] = q[X])\}$

   Solution:
   $\Pi_{P,Z}(\sigma_{R=X}(a \times b))$ will return attributes P from relation $a(A)$ and Z from relation $b(B)$. So, options C and D are incorrect.
   For SELECT operation, condition $R = X$ must be satisfied, So option B is incorrect.
   Thus, option 1 is correct.

   Consider the following relational schema and answer questions 2 and 3.

   [MCQ:2 points]

   - $Owner(aadhar\_number, o\_name)$
   - $Vehicle(v\_number, v\_model)$
   - $Registration(aadhar\_number, v\_number, purchase\_year)$

2. Which of the following relational algebra expressions is equivalent to the statement given below?

   - Find the Aadhaar numbers of owners who purchased the vehicle model $V20$ after year 2020.
     - ◯ $\sigma_{aadhar\_number}(\prod_{v\_model=\text{“}V20\text{”} \ \vee \ purchase\_year>\text{”}2020\text{”}}(Registration \bowtie Vehicle))$
     - ◯ $\sigma_{aadhar\_number}(\sigma_{v\_model=\text{“}V20\text{”} \ \wedge \ purchase\_year>\text{”}2020\text{”}}(Registration \bowtie Vehicle))$
     - √ $\prod_{aadhar\_number}(\sigma_{v\_model=\text{“}V20\text{”} \ \wedge \ purchase\_year>\text{”}2020\text{”}}(Registration \bowtie Vehicle))$
     - ◯ $\prod_{aadhar\_number}(\sigma_{v\_model=\text{“}V20\text{”} \ \vee \ purchase\_year>\text{”}2020\text{”}}(Registration \bowtie Vehicle))$

Solution: Selection Operator (σ), selects those rows or tuples from the relation that satisfies the selection condition.
Project operator is denoted by the symbol Π, and it is used to select desired columns (or attributes) from a table (or relation).
Option 1 and Option 2 are incorrect. Here, the SELECT operator is used, requiring a specific condition to select tuples from a relation.
Option 3: It will return the Aadhaar number of all the owners who purchased the vehicle model $V20$ **and** after year 2020.
So, Option 3 is correct.
Option 4: It will return the Aadhaar number of all the owners who purchased the vehicle model $V20$ **or** after year 2020.

3. Which of the following queries is equivalent to the statement given below?

   - Find the names of all owners who purchased vehicles with number 123 before the year 2019.

     ◯ $\{T \mid \exists O \in Owner, \exists R \in Registration(O.aadhar\_number = R.aadhar\_number \wedge R.v\_number = 123 \vee R.purchase\_year < 2019 \wedge T.o\_name = O.o\_name)\}$

     ◯ $\{T \mid \exists O \in Owner, \exists R \in Registration(O.aadhar\_number = R.aadhar\_number \wedge R.v\_number = 123 \vee R.purchase\_year < 2019 \vee T.o\_name = O.o\_name)\}$

     ◯ $\{T \mid \exists O \in Owner, \exists R \in Registration(O.aadhar\_number = R.aadhar\_number \vee R.v\_number = 123 \wedge R.purchase\_year < 2019 \vee T.o\_name = O.o\_name)\}$

     √ $\{T \mid \exists O \in Owner, \exists R \in Registration(O.aadhar\_number = R.aadhar\_number \wedge R.v\_number = 123 \wedge R.purchase\_year < 2019 \wedge T.o\_name = O.o\_name)\}$

Solution:
$\exists O \in Owner, \exists R \in Registration(O.aadhar\_number = R.aadhar\_number)$ will perform the NATURAL JOIN operation of Owner and Registration schema and $(R.v\_number = 123 \wedge R.purchase\_year < 2019 \wedge T.o\_name = O.o\_name)$ is the required condition for the names of all the owners who purchased the vehicle number 123 before year 2019.

4. Consider the E-R diagram for a cricket-training-camp database as given in Figure 1.

[MCQ: 2 points]

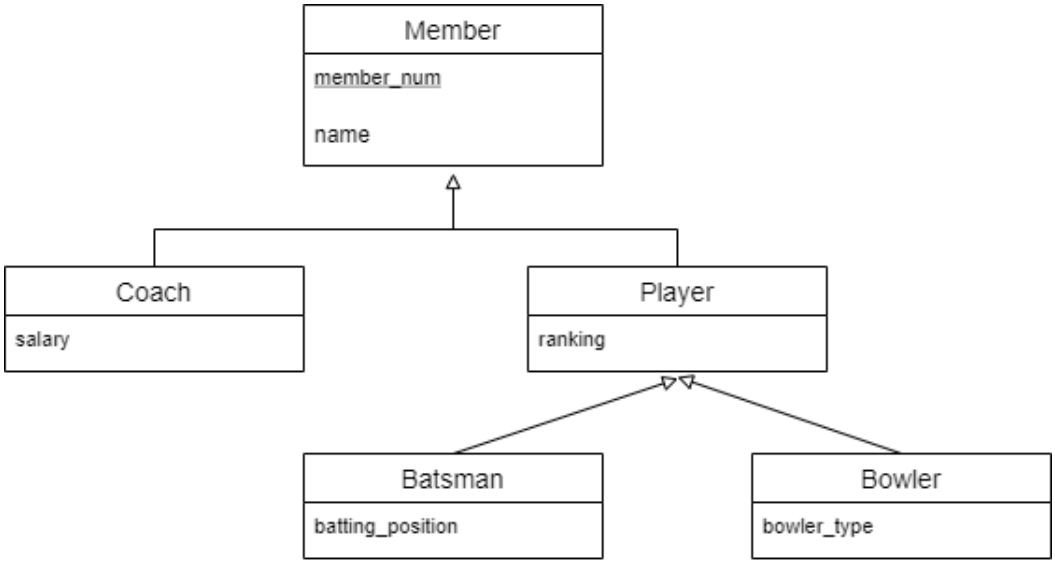


Figure 1: E-R diagram of cricket-training-camp database

Identify the option in which both the statements correctly describe the relations between the given entity sets.

◯ 1. Each member can be either a coach or a player or both in the cricket-training-camp.
   2. Each player can be a batsman or a bowler or both.

◯ 1. Each member can be a coach or a player or both.
   2. Each player can be either a batsman or a bowler. However, a player cannot be both, a batsman and a bowler at the same time.

◯ 1. Each member can be either a coach or a player. But, a member cannot be a coach and a player at the same time.
   2. Each player can be either a batsman or a bowler, but cannot be both.

√ 1. Each member can be either a coach or a player or just a member of the cricket-training-camp. But, a member cannot be a coach and a player at the same time.
   2. Each player can be a batsman or a bowler or both.

Solution:

- **Coach** and **Player** are disjoint specializations of **Member**.

- **Batsman** and **Bowler** are overlapping specializations of **Member**.
- Both kind of specializations given in Figure 1 are partial specializations.

Hence,

- Each member must be a coach or a player or just a member. But, a member cannot be a coach and a player at the same time.
- Each player can be a batsman or a bowler or both.

5. Consider the E-R diagram given in Figure 2.

[MCQ: 2 points:Solve with instructor]

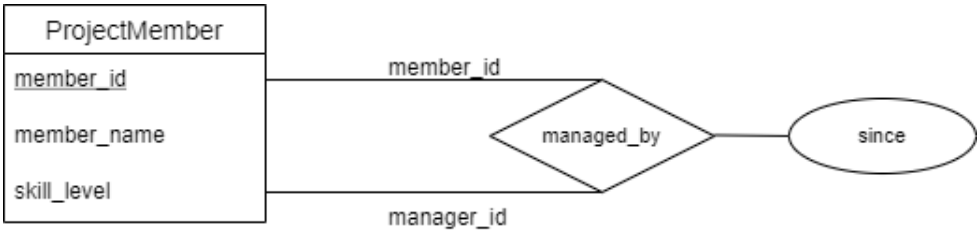


Figure 2: E-R diagram

The table for entity set **ProjectMember** is created using the command below:

```
CREATE TABLE ProjectMember(
member_id INT NOT NULL,
member_name VARCHAR(20) NOT NULL,
skill_level VARCHAR(20) NOT NULL,
PRIMARY KEY (member_id)
);
```

Select the appropriate command to create the table for relationship set **managed_by**.

◯
```
CREATE TABLE managed_by(
member_id INT NOT NULL,
since INT NOT NULL,
PRIMARY KEY (member_id),
FOREIGN KEY (member_id) REFERENCES ProjectMember(member_id)
);
```

√
```
CREATE TABLE managed_by(
member_id INT,
manager_id INT,
since INT NOT NULL,
PRIMARY KEY (member_id, manager_id),
FOREIGN KEY (member_id) REFERENCES ProjectMember(member_id),
FOREIGN KEY (manager_id) REFERENCES ProjectMember(member_id)
);
```

◯
```
CREATE TABLE managed_by(
manager_id INT NOT NULL,
since INT NOT NULL,
PRIMARY KEY (manager_id),
```

```
FOREIGN KEY (manager_id) REFERENCES ProjectMember(member_id)
);
```

◯
```
CREATE TABLE managed_by(
member_id INT NOT NULL,
manager_id INT NOT NULL,
since INT NOT NULL,
PRIMARY KEY (member_id, manager_id),
FOREIGN KEY (manager_id) REFERENCES ProjectMember(member_id)
);
```

Solution: The table **managed_by** must have $\{member\_id, manager\_id\}$ as primary key, both reference to **ProjectMember**($member\_id$), and the descriptive attribute *since* also becomes an attribute in the table.
Thus, it must be created by the command:

```
CREATE TABLE managed_by(
member_id INT,
manager_id INT,
since INT NOT NULL,
PRIMARY KEY (member_id, manager_id),
FOREIGN KEY (member_id) REFERENCES ProjectMember(member_id),
FOREIGN KEY (manager_id) REFERENCES ProjectMember(member_id)
);
```

Please note that since $\{member\_id,\ manager\_id\}$ is the primary key, the prime attributes *member_id* and *manager_id* by default not `NULL`.

6. Consider the entity set given in Figure 3. [MCQ: 2 points]

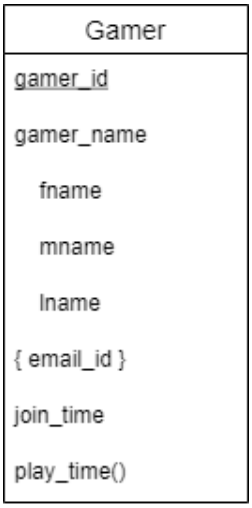


Figure 3: Entity set **Gamer**

Which of the following relational schemas appropriately represents the E-R diagram?

- ◯ **Gamer**($\underline{gamer\_id}, gamer\_name, fname, mname, lname, email\_id, join\_time$)
- ◯ **Gamer**($\underline{gamer\_id}, fname, mname, lname, email\_id, join\_time, play\_time$)
  Gamer_email($\underline{gamer\_id}, email\_id$)
- √ **Gamer**($\underline{gamer\_id}, fname, mname, lname, join\_time$)
  Gamer_email($\underline{gamer\_id}, \underline{email\_id}$)
- ◯ **Gamer**($\underline{gamer\_id}, email\_id, join\_time$)
  Gamer_name($\underline{gamer\_id}, gamer\_name, fname, mname, lname$)

Solution:

- The identifying attribute *gamer_id* becomes primary key for the schema.
- The composite attribute *gamer_name* will be replaced by its parts *fname*, *mname* and *lname* in the schema.
- The simple attribute *join_time* becomes another attribute.
- The derived attribute *play_time* does not need to be added in the schema.
- For the multivalued attribute *email_id* a separate relation has to be created which will be:
  $Gamer\_email(\underline{gamer\_id}, \underline{email\_id})$.

7. Consider the relations given below: [MSQ: 2 points:Solvewithinstructor]

- **doctor**($\underline{doc\_id}, doc\_name, specialization$)
- **patient**($\underline{patient\_num}, patient\_name$)
- **operationRoster**($\underline{doc\_id}, \underline{patient\_num}, operation\_cost$)

Identify the appropriate statement(s) to find the names of all doctors having specialization in orthopedics and who have charged more than $1000 as surgery charges.

√ $\prod_{doc\_name}(\sigma_{specialization=\text{“orthopedic”} \ \wedge \ operation\_cost>1000}(doctor \bowtie operationRoster))$

◯ $\{s \mid \exists s \in doctor, \ \exists r \in operationRoster(s.doc\_id = r.doc\_id \wedge s.specialization = \text{“orthopedic”} \wedge r.operation\_cost > 1000)\}$

√ $\{t \mid \exists s \in doctor, \ \exists r \in operationRoster(s.doc\_id = r.doc\_id \wedge s.specialization = \text{“orthopedic”} \ \wedge r.operation\_cost > 1000 \ \wedge \ t.doc\_name = s.doc\_name)\}$

√ $\{< D_N > \ \mid \exists D_I \exists R_P \exists R_C (< D_I, D_N, \text{“orthopedic”} > \in doctor \wedge < D_I, R_P, R_C > \in operationRoster) \wedge R_C > 1000\}$

Solution: As per the specifications given in the question, a natural join needs to be applied between **doctor** and **operationRoster** as:
$doctor \bowtie operationRoster$.
Then, a select operation can be applied as:
$\sigma_{specialization=\text{“orthopedic”} \ \wedge \ operation\_cost>1000}(doctor \bowtie operationRoster)$.
Finally, apply project the doc_name as:
$\prod_{doc\_name}(\sigma_{specialization=\text{”orthopedic”} \ \wedge \ operation\_cost>1000}(doctor \bowtie operationRoster))$.
The equivalent **tuple relational calculus** is:
$\{t \mid \exists s \in doctor, \ \exists r \in operationRoster(s.specialization = \text{“orthopedic”} \wedge r.operation\_cost > 1000 \ \wedge \ t.doc\_name = s.doc\_name)\}$.
The equivalent **domain relational calculus** is:
$\{< D_N > \ \mid \exists D_I \exists R_P \exists R_C (< D_I, D_N, \text{“orthopedic”} > \in doctor \wedge < D_I, R_P, R_C > \in operationRoster) \wedge R_C > 1000\}$.
Please note the tuple relation calculus:
$\{s \mid \exists s \in doctor \exists \ r \in operationRoster(s.doc\_id = r.doc\_id \wedge s.specialization = \text{“orthopedic”} \wedge r.operation\_cost > 1000)\}$
projects all attributes rather than doc_name alone.

8. Consider the relations below: [MSQ: 2 points]

- **customer**($\underline{customer\_id}, customer\_name, customer\_city$)
- **invoice**($\underline{invoice\_number}, \underline{customer\_id}, amount\_payable$)

Choose the correct relational algebra expressions that return the names of all customers having amount payable ($amount\_payable$) more than $1,000 and who are located in **Chennai**.

○ $\prod_{customer\_name}(\sigma_{amount\_payable>1000 \ \wedge \ customer\_city=\text{“}Chennai\text{”}}(customer \times invoice))$

○ $\prod_{amount\_payable>1000 \ \vee \ customer\_city=\text{“}Chennai\text{”}}(customer \bowtie invoice))$

√ $\prod_{customer\_name}(\sigma_{amount\_payable>1000 \ \wedge \ customer\_city=\text{“}Chennai\text{”}}(customer \bowtie invoice))$

√ $\prod_{customer\_name}$
$(\sigma_{amount\_payable>1000 \ \wedge \ customer\_city=\text{“}Chennai\text{”} \ \wedge \ customer.customer\_id=invoice.customer\_id}$
$(customer \times invoice))$

Solution: First, natural join can be applied between the two relations **customer** and **invoice** such that tuples will be combined by equality in *customer_id*. Then, σ with predicate $amount\_payable > 1000 \ \wedge \ customer\_city = \text{“}Chennai\text{”}$ can be applied to find the tuples as per the specification given.
Alternatively, the same can also be achieved by a Cartesian product between **customer** and **invoice** along with σ with predicate equality of *customer_id* of both the relations and σ with predicate $amount\_payable > 1000 \ \wedge \ customer\_city = \text{“}Chennai\text{”}$.
Finally, we project ($\prod$) the *customer_name*.

9. Consider the table given in Figure 4. [MSQ: 2 points]

S1	
A	B
5	25
6	36
7	49
8	64
9	91

Figure 4: Relation **S1**

Choose the correct set of expressions that will return the tuple given below.

A
7

◯ $\sigma_A(\Pi_{B=49}(S1))$

√ $\{t \mid \exists p \in S1(t[A] = p[A] \wedge p[B] = 49)\}$

◯ $\{t \mid \exists p \in S1(t[A] = p[A] \wedge p[B] = 7)\}$

√ $\{< a > \mid \exists b(< a, b > \in S1 \wedge b = 49)\}$

Solution:
$\sigma_A(\Pi_{B=49}(S1))$ is logically incorrect, the correct TRC query is $\Pi_A(\sigma_{B=49}(S1))$, this will first perform the Select operation and return the row having B = 49 then it will project the corresponding value of attribute A.

$\{t \mid \exists p \in S1(t[A] = p[A] \wedge p[B] = 49)\}$ is equivalent to $\Pi_A(\sigma_{B=49}(S1))$

$\{< a > \mid \exists b(< a, b > \in S1 \wedge b = 49)\}$ is equivalent to $\Pi_A(\sigma_{B=49}(S1))$

10. Consider the E-R diagram in Figure 5. [NAT: 2 points]

Figure 5: ERD

What is the minimum number of tables needed to represent this E-R diagram?

Solution: 1
The minimum and maximum cardinality is 1 (1..1).

- A minimum value of 1 indicates total participation.
- A maximum value of 1 indicates that the entity participates in at most one relationship.

Thus, it can be represented using a single table:
team_captain($\underline{team\_code}, team\_name, \underline{player\_num}, player\_name$).

Consider the E-R diagram given in Figure 6 and answer the questions 11 and 12.

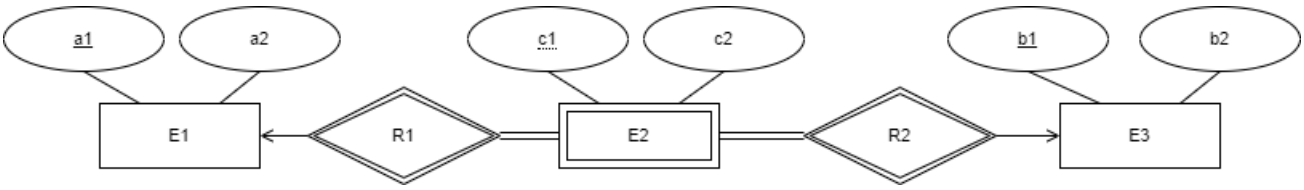


Figure 6: E-R diagram

11. The minimum number of tables required to represent the entity sets and relationship sets is ..... [NAT: 2 points]
Answer: 3

Solution: 3

- **E1** is associated with **E2** via **R1** in a one-to-many relation.
- **E3** is associated with **E2** via **R2** in a one-to-many relation.

Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side. Thus, we can represent the entire ERD using 3 tables as follows:

- **E1**$(\underline{a1}, a2)$
- **E2**$(\underline{c1}, c2, \underline{a1}, \underline{b1})$
- **E3**$(\underline{b1}, b2)$

12. What will be the correct attribute set for the table corresponding to the entity set **E2**? [MCQ: 2 points:Solvewithinstructor]

◯ **E2**$(\underline{c1}, c2)$
◯ **E2**$(\underline{c1}, \underline{a1}, c2)$
◯ **E2**$(\underline{c1}, a1, b1, c2)$
√ **E2**$(\underline{c1}, c2, \underline{a1}, \underline{b1})$

Solution:

- **R1** is a one-to-many relationship set from **E2** to **E1**.
- **R2** is a one-to-many relationship set from **E2** to **E3**.

Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side. Thus, we can represent the entire ERD using 3 tables as follows:

- **E1**$(\underline{a1}, a2)$
- **E2**$(\underline{c1}, c2, \underline{a1}, \underline{b1})$
- **E3**$(\underline{b1}, b2)$

13. Consider the E-R diagram with aggregation given in Figure 7. [MCQ: 2 points]

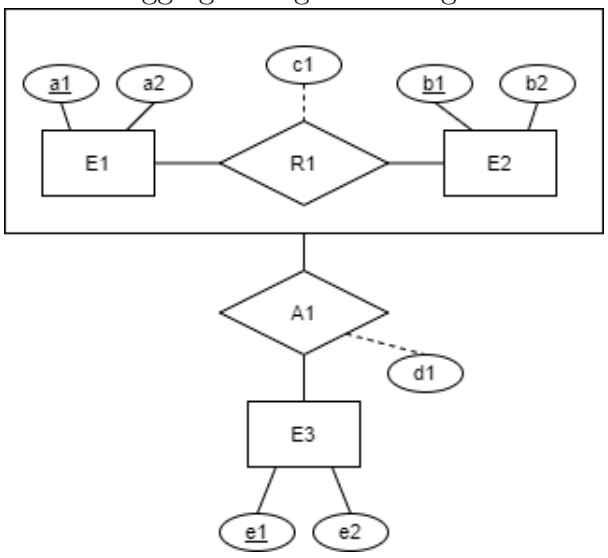


Figure 7: ERD

What will be the correct attribute set for the table corresponding to relationship-set **A1**?

◯ $c1, e1, d1$
◯ $a1, b1, d1, c1, e1, e2$
◯ $a1, b1, d1, e1, e2$
√ $a1, b1, e1, d1$

Solution: The ER-diagram presents a scenario of aggregation. Thus, the relationship set **A1** must be mapped to a table having the following:

- Primary keys of **E1**, **E2** and **E3**.
- Any descriptive attributes of **A1**.

So the attribute set for **A1** is: $\{a1, b1, e1, d1\}$

BSCCS2001: Practice Solutions
Week 5

1. Consider the following [MCQ: 2 points]
   $X = \{A \to BC, B \to A, C \to A\}$
   $Y = \{A \to B, B \to C, C \to A\}$

   ◯ X covers Y

   ◯ Y covers X

   ◯ X and Y are equivalent

   √ All the above

Solution: Let us check X covers Y, every functional dependency in Y logically implies in X.
FDs in Y : $A \to B$, $B \to C$, $C \to A$
check for $A \to B$,
$A^+ \to ABC$ $\{A \to BC$ in $X\}$
Check for $B \to C$
$B^+ \to BAC$, $\{A \to BC, B \to A$ in $X\}$
Check for $C \to A$
$C^+ \to CAB$ $\{C \to A, A \to BC$ in $X\}$
Therefore, X covers Y
Let us now check, If Y covers X
FDs in X: $A \to BC$, $B \to A$, $C \to A$
Check for $A \to BC$
$A^+ \to ABC$ $\{A \to B$, $B \to C$ in $Y\}$
Check for $B \to A$
$B^+ \to BCA$ $\{B \to C$, $C \to A$ in $Y\}$
Check for $C \to A$
$C^+ \to A$ $\{C \to A$ in $X\}$
Therefore, Y covers X
Since, both X & Y covers each other, so they are equivalent $X \equiv Y$

2. Consider relation **R***(A,B,C,D,E,F)* with the following functional dependencies:

$$\mathcal{F} = \{AB \rightarrow C, AC \rightarrow E, EF \rightarrow D, AB \rightarrow F\}$$

[MSQ: 2 points]

Which among the following is true about R?

- ◯ *R1(D, E, F), R2(A, B, C, E, F)* is a lossy decomposition of R.
- √ *R1(A, B, D), R2(C, E, F)* is a lossy decomposition of R.
- ◯ *R1(A, D, E, C), R2(B, E, C)* is a lossless decomposition of R.
- √ *R1(D, E, A, B), R2(C, F, B)* is a lossy decomposition of R.

Solution:

This problem will be solved in the *Solve With the Instructor* session.

3. Consider relation **T20WC** defined as **W***(Team, Ranking, Captain, Points, Players)* with the following functional dependencies:

$$\mathcal{F} = \{Team, Ranking \rightarrow Captain, Ranking \rightarrow Players, Captain \rightarrow Points\}$$

Then, which of the following is true ? [MCQ:2points]

- ◯ *W1(Team, Ranking, Captain)*, *W2(Captain, Points)* is a lossless-join decomposition.
- √ *W1(Team, Ranking, Captain)*, *W2(Points, Players)* is a lossy-join decomposition
- ◯ *W1(Team, Ranking, Captain)*, *W2(Ranking, Points, Players)* is a lossless-join decomposition.
- ◯ None of the above

Solution:
Option 1 : *W1(Team, Ranking, Captain)*, *W2(Captain, Points)*
$W1 \cup W2 \neq W$
Thus, it is lossy join decomposition.
Option 2 : *W1(Team, Ranking, Captain)*, *W2(Points, Players)*
$W1 \cup W2 = W$
$W1 \cap W2 = \phi$
Thus, it is lossy join decomposition. So, option 2 is correct
Option 3 : *W1(Team, Ranking, Captain)*, *W2(Ranking, Points, Players)*
$W1 \cup W2 = W$
$W1 \cap W2 = Ranking$
$Ranking^+ \rightarrow Ranking, Players$
Ranking is not superkey for any relation. Hence, we can't determine $W1$ and $W2$ from it. So, it is lossy decomposition.

Numerical Answer Type

4. Consider the relation **Book***(Author, Publisher, Pages, Ratings, Type)* having the following functional dependencies : [NAT: 2 points]
$\mathcal{F}$= { $Author \rightarrow Publisher, Pages$
$Publisher \rightarrow Ratings$
$Pages, Ratings \rightarrow Type$
$Type \rightarrow Author$ }
What is the maximum number of candidate keys for **Book**?

   Ans : 4

   Solution: By estimating the closure of all combination of attributes, it can be observed that the closure of the following attributes produces all other attributes:
   $Author, Type, (Publisher, Pages), (Pages, Ratings)$.
   Hence, these 4 are the candidate keys.

5. In a relation **R***(A, B, C, D, E)*, each attribute is a candidate key. Then, what is the maximum number of super keys possible for **R**?

[NAT: 2 points]

Ans: 31

Solution: Consider a relation $R(A_1, A_2, A_3.....A_n)$, then maximum number of super keys are 2^n-1. (If Each attribute of a relation is candidate key)
Here, n = 5, so, the number of super keys for a given relation R is 31.

6. Which among the following is/are the use(s) of finding closure of attributes? [MSQ: 2 points]

◯ Find if an attribute or set of attributes is a superkey.
◯ Compute the canonical cover of a given set of functional dependencies
◯ Test if a specific functional dependency holds
◯ Compute the closure of a given set of functional dependencies
√ All the above

Solution: The solution follows from the lectures.

7. Let $\mathbf{R}(A, B, C, D, E)$ be a relation with the following functional dependencies:

$$\mathcal{F} = \{A \to C, A \to B, C \to D, BC \to E\}$$

[MCQ: 2 points]

Then,

- ◯ $\mathcal{F}^+ = \{A \to C, A \to B, C \to D, BC \to E\}$
- ◯ $\mathcal{F}^+ = \{A \to C, A \to B, C \to D, A \to D, BC \to E\}$
- ◯ $\mathcal{F}^+ = \{A \to BC, C \to D, BC \to E\}$
- √ $\mathcal{F}^+ = \{C \to D, BC \to E, A \to BCDE\}$

Solution: This problem will be solved in the *Solve With the Instructor* session.

8. Let $\mathbf{R}(A, B, C, D, E)$ be a relation with the following functional dependencies:

[MCQ: 2 points]

$\mathcal{F} = \{A \to B, C \to D, BD \to E\}$
Then, which of the following functional dependencies can be derived from $\mathcal{F}$ using Armstrong's Axioms?

- √ $AC \to E$
- ◯ $BE \to D$
- ◯ $B \to A$
- ◯ $C \to E$

Solution:
$A \to B$
$AC \to BC$ {Augmentation}
$C \to D$
$BC \to BD$ {Augmentation}
$BD \to E$
$AC \to E$ {Transitivity}

9. Choose the correct canonical cover of the set of functional dependencies $\mathcal{F}$ that occur in a relation $\mathbf{R}(A, B, C, D)$, where

$$\mathcal{F} = \{A \to BC, AB \to C, A \to D, D \to C\}$$

[MCQ: 2 points]

- ◯ $A \to BC, AB \to C$
- ◯ $A \to BC, AB \to C, A \to D$
- √ $A \to B, A \to D, D \to C$
- ◯ $A \to B, B \to C, A \to D$

Solution: Given $A \to BC, AB \to C, A \to D, D \to C$.
$A \to D, D \to C \Rightarrow A \to C$
That is, in the FD $A \to BC$, $A \to C$ is redundant.
Hence we can remove $A \to C$ from $A \to BC$.

$$\mathcal{F} = \{A \to B, AB \to C, A \to D, D \to C\}$$

Since $A \to C$ is a stronger constraint than $AB \to C$ and since $A \to C$ can be derived from $A \to D, D \to C$, we can remove $AB \to C$ from $\mathcal{F}$.
Therefore

$$\mathcal{F} = \{A \to B, A \to D, D \to C\}$$

10. Choose the set of FDs equivalent to:

$$A \to BC, B \to CE, C \to ED$$

[MSQ: 2 points]

- √ $A \to BE, B \to CD, C \to ED$
- ◯ $A \to B, B \to D, C \to E$
- √ $A \to B, B \to C, C \to ED$
- ◯ $A \to B, B \to C, C \to D, D \to A$

Solution:

This problem will be solved in the *Solve With the Instructor* session.

11. Given the relation **hospital** and its decomposition into **hosp1** and **hosp2** as shown in Figure 1, choose the correct set of options. [MCQ: 2 points]

hospital

hospitalNum	patientNum	doctorID
H0001	P0001	D0001
H0002	P0002	D0002
H0003	P0001	D0003

hosp1

hospitalNum	patientNum
H0001	P0001
H0002	P0002
H0003	P0001

hosp2

patientNum	doctorID
P0001	D0001
P0002	D0002
P0001	D0003

Figure 1: Decomposition of **hospital** relation

◯ The given decompostion is lossless and the natural join of **hosp1** and **hosp2** has 5 rows.

◯ The given decompostion is lossless and the natural join of **hosp1** and **hosp2** has 3 rows.

√ The given decompostion is lossy and the natural join of **hosp1** and **hosp2** has 5 rows.

◯ The given decompostion is lossy and the natural join of **hosp1** and **hosp2** has 3 rows.

Solution:

hospital

hospitalNum	patientNum	doctorID
H0001	P0001	D0001
H0002	P0002	D0002
H0003	P0001	D0003
H0001	P0001	D0003
H0003	P0001	D0001

Figure 2: Natural join of **hosp1** and **hosp2**

12. Consider a relation $R(A, B, C, D, E)$ having the following functional dependencies:

$$\mathcal{F} = \{A \rightarrow BCD, D \rightarrow E, C \rightarrow D\}$$

Which among the following are lossy decompositions? [MSQ: 2 points]

- ◯ $R_1(A, B, C), R_2(B, C, D), R_3(C, D, E)$
- ◯ $R_1(A, B, C), R_2(A, C, D), R_3(A, D, E)$
- √ $R_1(A, B, C), R_2(A, C), R_3(A, D)$
- ◯ $R_1(A, B, C, D), R_2(A, C, D, E)$

Solution:

This problem will be solved in the *Solve With the Instructor* session.

BSCCS2001: Practice with Solutions
Week 6

1. Consider the relational schema $\mathbf{R}(A, B, C, D, E)$, where the domains of A, B, C, D and E include only atomic values. Identify the possible set of functional dependencies that $\mathbf{R}$ can have such that $\mathbf{R}$ is in BCNF.

[MSQ: 2 points]

- √ FD: $\{AB \rightarrow CDE\}$
- ◯ FD: $\{AB \rightarrow CD, B \rightarrow E\}$
- ◯ FD: $\{AB \rightarrow CD, C \rightarrow D, D \rightarrow E\}$
- ◯ FD: $\{AB \rightarrow CDE, D \rightarrow A, E \rightarrow B\}$

Solution: Given that in $\mathbf{R}$ each attribute is a single-valued attribute. Thus $\mathbf{R}$ is already in 1NF.
Option-1: FD: $\{AB \rightarrow CDE\}$
The only candidate key (thus primary key) is: AB as $(AB)^+ = \{ABCDE\}$.
As all the non-prime attributes are fully functionally dependent on the candidate key, it is already in 2NF.
$\{AB \rightarrow CDE\}$, where AB is a superkey. Thus, **it is in 3NF and also in BCNF**.
Option-2: $\{AB \rightarrow CD, B \rightarrow E\}$
The only candidate key (thus primary key) is: AB as $(AB)^+ = \{ABCDE\}$.
$B \rightarrow E$ is a partial functional dependency. Thus, it is in 1NF but not in 2NF.
Option-3: FD: $\{AB \rightarrow CD, C \rightarrow D, D \rightarrow E\}$
The only candidate key (thus primary key) is: AB as $(AB)^+ = \{ABCDE\}$.
There is no partial functional dependency. Thus, it is already in 2NF.
$AB \rightarrow CD$, where AB is superkey.
But, for $C \rightarrow D$, $D \rightarrow E$

- the functional dependencies are not trivial.
- L.H.S of the functional dependencies are not superkeys.
- R.H.S of the functional dependencies are not prime attributes.

Thus, these two FDs violate 3NF rules. So, $\mathbf{R}$ is in 2NF but not in 3NF based on this set of FDs.
Option-4: FD: $\{AB \rightarrow CDE, D \rightarrow A, E \rightarrow B\}$
The candidate keys are: AB and DE as $(AB)^+ = \{ABCDE\}$ and $(DE)^+ = \{ABCDE\}$. The prime attributes are A, B, D, E.
There is no partial functional dependency. Thus, it is already in 2NF.
$AB \rightarrow CDE$, where AB is superkey.
For $D \rightarrow A$, $E \rightarrow B$ R.H.S of the functional dependencies are prime attributes. Thus, it is in 3NF. However, These two FDs do not satisfy BCNF (as L.H.S are not superkeys). So, $\mathbf{R}$ is in 3NF but not in BCNF based on this set of FDs.

2. Consider the relational schema $\mathbf{R}(A, B, C, D, E, F)$, where the domains for A, B, C, D, E and F include atomic values only. If $\mathbf{R}$ satisfies the functional dependencies $\{AB \rightarrow CDE, E \rightarrow F, BF \rightarrow A, C \rightarrow B\}$, then identify the correct statement(s).

[MSQ: 2 points]

- ◯ **R** is in 1NF but not in 2NF
- √ **R** is in 2NF and also in 3NF
- √ **R** is in 3NF but not in BCNF
- ◯ **R** is in 3NF also in BCNF

Solution:
Candidate keys are: AB, BF, AC, BE, CF and CE. So, prime attributes are: A, B, C, E and F. For the FDs: $E \rightarrow F$ and $C \rightarrow B$, B and F are prime attributes. Thus, there is no partial dependency, thus R is in 2NF.
$AB \rightarrow CDE$ and $BF \rightarrow A$, as AB and BF both are candidate keys, the FDs are in 3NF.
$C \rightarrow B$ and $E \rightarrow F$ also in 3NF, since B and F are prime attributes. Thus, R is in 3NF.
$C \rightarrow B$ and $E \rightarrow F$ violate BCNF conditions as C and E are not superkeys. Thus, R is not in BCNF.

3. Consider the relational schema **Z**(P, Q, R, S) and the following functional dependencies on **Z**. [MCQ: 2 points]

- $P \rightarrow QRS$
- $Q \rightarrow R$
- $RS \rightarrow P$

Which of the following is/are correct?

◯ **Z** is in 3NF and also in BCNF

√ **Z** is in 3NF but not in BCNF

◯ **Z** is in 2NF but not in 3NF

◯ **Z** is in BCNF but not in 3NF

Solution: FD = $\{P \rightarrow QRS,\ Q \rightarrow R,\ RS \rightarrow P\}$
$P^+ = PQRS$
$RS^+ = PQRS$
$QS^+ = PQRS$
So, candidate keys are P, QS & RS and prime attribute are P, Q, R & S.
Since the schema **Z** has no partial dependencies or transitive dependencies, so it is in 3NF.

Check for BCNF
$P \rightarrow QRS$ (P is candidate key) ✓
$Q \rightarrow R$ (Q is not candidate key) ×
$RS \rightarrow P$ (RS is candidate key) ✓

So, **Z** is in 3NF but not in BCNF.

4. Let $\mathbf{R}(P,Q,R,S,T,U,V,W)$ be a relation (all attributes have atomic values only) with the following functional dependencies:

   - $\{PQ \rightarrow RSTU\}$
   - $\{P \rightarrow R\}$
   - $\{Q \rightarrow S\}$
   - $\{R \rightarrow UV\}$
   - $\{V \rightarrow W\}$
   - $\{W \rightarrow U\}$
   - $\{V \rightarrow U\}$

   Find the highest normal form in which the relation **R** is in.

   [MCQ: 2 points]

   √ 1NF
   ◯ 2NF
   ◯ 3NF
   ◯ BCNF

   Solution: Since all attributes in **R** have atomic values, it follows that **R** is in 1NF.

   In order to check if **R** is in 2NF, we must find the candidate keys. Using the given FDs, we find that PQV is the only candidate key. Hence P, Q and V are the prime attributes and the rest are non-prime.
   Now due to the presence of partial dependency, the relation **R** is not in 2NF.

   Note: Partial dependency occurs when a non-prime attribute is functionally dependent on a subset of a candidate key.

5. Consider the instance of relation **Course** given in Figure 1. [MSQ: 2 points]

course_name	instructor	book	edition
DBMS	Geeta	DBMS-Beginner	3
DBMS	Arjun	DBMS-Beginner	3
DBMS	Geeta	DBMS-Expert	2
DBMS	Arjun	DBMS-Expert	2
Java	Rahul	Java-Beginner	5
Java	Rahul	Java-Intermediate	3
Java	Rahul	Java-Expert	4
Java	Armaan	Java-Beginner	5
Java	Armaan	Java-Intermediate	3
Java	Armaan	Java-Expert	4

Figure 1: An instance of relation **Course**

Which among the following multivalued dependencies can be inferred from the given information?

- √ $course\_name \twoheadrightarrow instructor$
- ◯ $course\_name \twoheadrightarrow book$
- ◯ $course\_name \twoheadrightarrow edition$
- √ $course\_name \twoheadrightarrow book, edition$

Solution:
Let us first number the tuples as $t_1, t_2, \cdots, t_{10}$.
Test for $course\_name \twoheadrightarrow instructor$**:**
In relation **Course**, there exist two tuples t_1 and t_2 such that $t_1[course\_name] = t_2[course\_name]$.
We also have two tuples t_3 and t_4 in **Course** with the following properties:

- $t_1[course\_name] = t_2[course\_name] = t_3[course\_name] = t_4[course\_name]$,
- $t_3[instructor] = t_1[instructor]$ and $t_2[instructor] = t_4[instructor]$,
- $t_1[book, edition] = t_2[book, edition]$ and $t_3[book, edition] = t_4[book, edition]$.

Thus it satisfies MVD conditions.
In the relation **Course**, there are three tuples t_5, t_6 and t_7 such that $t_5[course\_name] = t_6[course\_name] = t_7[course\_name]$.
We also have three tuples t_8, t_9 and t_{10} in **Course** with the following properties:

- $t_5[course\_name] = t_6[course\_name] = t_7[course\_name] = t_8[course\_name] = t_9[course\_name] = t_{10}[course\_name]$,

- $t_5[instructor] = t_6[instructor] = t_7[instructor]$ and
  $t_8[instructor] = t_9[instructor] = t_{10}[instructor]$,

- $t_5[book, edition] = t_8[instructor, edition]$,
  $t_6[book, edition] = t_9[instructor, edition]$
  and $t_7[book, edition] = t_{10}[book, edition]$.

Thus, MVD conditions are satisfied.
Test for $course\_name \rightarrow\rightarrow book, edition$:
MVD Complementation rule: In a relation R, if $X \rightarrow\rightarrow Y$, then $X \rightarrow\rightarrow R - XY$.
Since we already have $course\_name \rightarrow\rightarrow instructor$, it follows that $course\_name \rightarrow\rightarrow book, edition$ also correct.
If we follow the same procedures as discussed above, we will be able to show that the MVDs:
$course\_name \rightarrow\rightarrow book$
$course\_name \rightarrow\rightarrow edition$
are not satisfied on relation **Course**.

6. Consider the relational schema:
Intern($intern\_code, intern\_name, project, hobby$).
An intern can work in several projects and can have several hobbies. However, it maintains the FD: $intern\_code \rightarrow intern\_name$.
Identify the most appropriate 4NF decomposition for the given schema.

[MCQ: 2 points]

- ◯ **R1**($intern\_code, intern\_name, project, hobby$), **R2**($intern\_code, project, hobby$)
- ◯ **R1**($intern\_code, intern\_name, project$), **R2**($intern\_code, hobby$)
- ◯ **R1**($intern\_code, intern\_name, hobby$), **R2**($intern\_code, project$)
- √ **R1**($intern\_code, intern\_name$), **R2**($intern\_code, project$), **R3**($intern\_code, hobby$)

Solution:
From the given information in the question, $intern\_code$ cannot be a super key for the given relation. Thus, $intern\_code \rightarrow intern\_name$ violates BCNF conditions.
Thus, a possible BCNF decomposition would be:
R1($intern\_code, intern\_name$), where $intern\_code$ is the candidate key, and
R2($intern\_code, project, hobby$).
R2 violates 4NF conditions as it has the following MVDs:
$intern\_code \twoheadrightarrow project$, and
$intern\_code \twoheadrightarrow hobby$
So the 4NF decomposition is:
R2($intern\_code, project$), and
R3($intern\_code, hobby$).
Thus, the 4NF decomposition is:
R1($intern\_code, intern\_name$),
R2($intern\_code, project$),
R3($intern\_code, hobby$).

7. Let $\mathbf{S}(Y,\ U,\ V)$ be a relation. Let $\mathbf{R}(P,\ W,\ X,\ Y,\ Z)$ be another relation with the following functional dependencies:
$\mathcal{F} = \{X \rightarrow ZW, Y \rightarrow X, W \rightarrow P\}$

$\mathbf{R}$ contains 300 tuples and $\mathbf{S}$ contains 250 tuples. What is the maximum number of tuples possible as output of $\mathbf{R} \bowtie \mathbf{S}$?

[MCQ: 2 point]

- ◯ 75000
- √ 250
- ◯ 300
- ◯ 50

Solution: From the given set of functional dependencies, Y is a candidate key of relation $\mathbf{R}$. So all 300 values of Y must be unique in $\mathbf{R}$.
There is no functional dependency given for S and to get the maximum number of tuples in output, there can be two possibilities for S.

- All 250 values of Y in $\mathbf{S}$ are same and there is an entry in $\mathbf{R}$ that matches with this value. In this case, we get 250 tuples in output.
- All 100 values of Y in $\mathbf{S}$ are different and these values are present in $\mathbf{R}$ also. In this case also, we get 250 tuples.

8. Let $\mathbf{A}(T,U,V,W)$ be a relational schema with the following functional dependencies:
$\mathcal{F} = \{W \rightarrow UT, UV \rightarrow W, V \rightarrow T, W \rightarrow U\}$
It is given that **A** is not in BCNF.
Suppose **A** is decomposed into two relational schemas, $\mathbf{B}(TV)$ and $\mathbf{C}(UVW)$.
Which of the following statement(s) is/are correct?

[MSQ: 2 points]

◯ Decomposition of schema A into B and C is dependency preserving

√ Decomposition of schema A into B and C is lossless

◯ Both B and C are in BCNF

√ Relation B is in BCNF

Solution:

- $\mathbf{B}(TV)$ preserves $\{V \rightarrow T\}$ and has V as the candidate key. So, relation **B** is in BCNF.
- $\mathbf{C}(UVW)$ preserves $\{UV \rightarrow W,\ W \rightarrow U\}$ and has UV and VW as the candidate keys. So, relation **C** is in 3NF but not in BCNF, as W is not a superkey.
- The decomposition of schema **A** into two relational schemas, **B** and **C**, does not cover all the functional dependencies of the original relation **A**. Hence, it is not dependency preserving.
- The decomposition has common attribute (i.e., V) which is superkey of relation $\mathbf{B}(TV)$, so decomposition of **A** into **B** and **C** is lossless join decomposition.

9. Consider the relational schema:
prescription($doctor\_id, doctor\_name, patient\_id, patient\_name, medicine\_id, medicine\_name$),
where the domains of all the attributes consist of atomic values. Consider the following FDs for the relation *department*.

[MCQ: 2 points]

- $doctor\_id \rightarrow doctor\_name$,
- $patient\_id \rightarrow patient\_name$,
- $medicine\_id \rightarrow medicine\_name$,
- $doctor\_id \rightarrow\!\!\rightarrow patient\_id$,
- $doctor\_id \rightarrow\!\!\rightarrow medicine\_id$

From among the decompositions given, identify the one that is in 4NF.

◯ $(doctor\_id, doctor\_name)$,
$(patient\_id, patient\_name)$,
$(medicine\_id, medicine\_name)$,

◯ $(doctor\_id, doctor\_name)$,
$(patient\_id, patient\_name)$,
$(medicine\_id, medicine\_name)$,
$(doctor\_id, patient\_id, medicine\_id)$

◯ $(doctor\_id, doctor\_name, patient\_id, patient\_name)$,
$(doctor\_id, doctor\_name, medicine\_id, medicine\_name)$

√ $(doctor\_id, doctor\_name)$,
$(patient\_id, patient\_name)$,
$(medicine\_id, medicine\_name)$,
$(doctor\_id, patient\_id)$,
$(doctor\_id, medicine\_id)$

Solution: For the given relation, the candidate key is $\{doctor\_id, patient\_id, medicine\_id\}$ and it is in 1NF. However, it is not in 2NF as the FDs:
$doctor\_id \rightarrow doctor\_name$,
$patient\_id \rightarrow patient\_name$,
$medicine\_id \rightarrow medicine\_name$, are partial functional dependencies. Thus, a possible decomposition is:
R1($doctor\_id, doctor\_name$), where $doctor\_id$ is the candidate key,
R2($patient\_id, patient\_name$), where $patient\_id$ is the candidate key,
R3($medicine\_id, medicine\_name$), where $medicine\_id$ is the candidate key,
R4($doctor\_id, patient\_id, medicine\_id$), where $\{doctor\_id, patient\_id, medicine\_id\}$ is the candidate key,
R1, **R2**, **R3** and **R4** are already in 3NF and BCNF.

R1, **R2** and **R3** are already in 4NF. The MVDs,
$doctor\_id \twoheadrightarrow patient\_id$, and
$doctor\_id \twoheadrightarrow medicine\_id$ violate 4NF conditions. Thus, **R4** is decomposed as:
R41$(doctor\_id, patient\_id)$ and
R42$(doctor\_id, medicine\_id)$.
Thus, the 4NF decomposition is:
R1$(doctor\_id, doctor\_name)$,
R2$(patient\_id, patient\_name)$,
R3$(medicine\_id, medicine\_name)$
R41$(doctor\_id, patient\_id)$ and
R42$(doctor\_id, medicine\_id)$.

10. Consider the relational schema **R** as:
R(A, B, C, D, E, F, G, H), where the domains of all the attributes consist of atomic values. Consider the following FDs for the relation *department*.

- $A \rightarrow D$,
- $D \rightarrow EF$,
- $BH \rightarrow CG$,
- $G \rightarrow H$,

From among the decompositions given, identify the one that is in BCNF.

[MCQ: 2 points]

◯ $(A, D, E, F), (G, H), (B, C, G, H)$ and (A, B, H)
◯ $(D, E, F), (A, D), (G, H)$ and (B, C, G)
√ $(D, E, F), (A, D), (G, H), (B, C, G)$ and (A, B, H)
◯ $(D, E, F), (A, D), (B, C, G, H)$ and (A, B, H)

Solution: Due to atomic values, the relation **R** is in 1NF.
Candidate key is: ABH as $(ABH)^+ = R$.
Test for 2NF: FD: $A \rightarrow D$ and $BH \rightarrow CG$ violate 2NF conditions (these are partial functional dependencies). Thus, the decomposition of **R** is:
Since $(A)^+ = ADEF$, **R1**(A, D, E, F), where A is the candidate key,
since $(BH)^+ = BHCG$, **R2**(B, H, C, G), where BH is the candidate key, and
R3(A, B, H) consists of the original candidate key of R.
Now, **R1**, **R2** and **R3** are in 2NF.
Test for 3NF: In **R1**, FD $D \rightarrow EF$ violates 3NF conditions (as D is not a superkey). Thus, the decomposition is:
R11(D, E, F), where D is the candidate key and **R12**(A, D), where A is the candidate key.
The relations **R3** and **R2** are already in 3NF (since in **R2**, FD: $G \rightarrow H$ satisfies 3NF conditions as H is a prime attribute).
Test for BCNF: The relations **R11**, **R12** and **R3** are already in BCNF. However, in relation **R2**, FD: $G \rightarrow H$ violates BCNF conditions as G is not a super key). Thus, the decomposition is:
R22(G, H), where G is the candidate key and **R22**(B, C, G), where BCG is the candidate key.
The final relations after decomposition are:
R11(D, E, F), **R12**(A, D), **R21**(G, H), **R22**(B, C, G) and **R3**(A, B, H). Please note that although the decomposition is lossless, it is not dependency preserving.

11. Which of the following statements is/are true regarding temporal relations?

[MSQ: 2 points]

◯ A uni-temporal relation can have only valid time.

◯ A uni-temporal relation can have only transaction time.

√ A uni-temporal relation can have either valid transaction time or transaction time.

√ A bi-temporal relation can have both valid transaction time and transaction time.

Solution:

- An uni-temporal relations has one axis of time, either valid time or transaction time.
- A bi-temporal relation has both axis of time, valid time and transaction time. It includes valid start time, valid end time, transaction start time, transaction end time.

BSCCS2001: Practice with Solutions
Week 7

1. Select the correct statement(s) from the following options: [MSQ: 2 points]

- √ Common Gateway Interface (CGI) is a standard interface between web and database server.
- ◯ The main function of the server side scripting is to correspond within a webpage.
- √ URIs can be classified as locators(URLs), or as names (URNs), or both.
- ◯ HTTP can be used to format most of the web documents into hypertext documents.

Solution:

- The Common Gateway Interface (CGI) provides the middleware between WWW servers and external databases and information sources.
- The main function of the server side scripting is to carry out a task at the server's end and then send the result to the client side.
- URNs and URLs are the subsets of the URIs.
- HTML can format most of the web documents into hypertext documents.

2. Identify the three main components of Application Architecture Layer. [MSQ: 2 points]

◯ Controller Layer, Data Access Layer, Backend Layer

◯ Presentation Layer, Controller Layer, Model Access Layer

√ Presentation Layer, Middle Layer, Backend

√ Presentation Layer, Business Logic Layer, Data Access Layer

Solution: Application layer consists of 3 sub-layers namely:

- Frontend or Presentation Layer
- Middle Layer or Application / Business Logic Layer
- Backend or Data Access Layer

Presentation layer follows model-view-controller (MVC) architecture.

3. Identify A, B and C marked in Figure 1. [MCQ: 2 points]

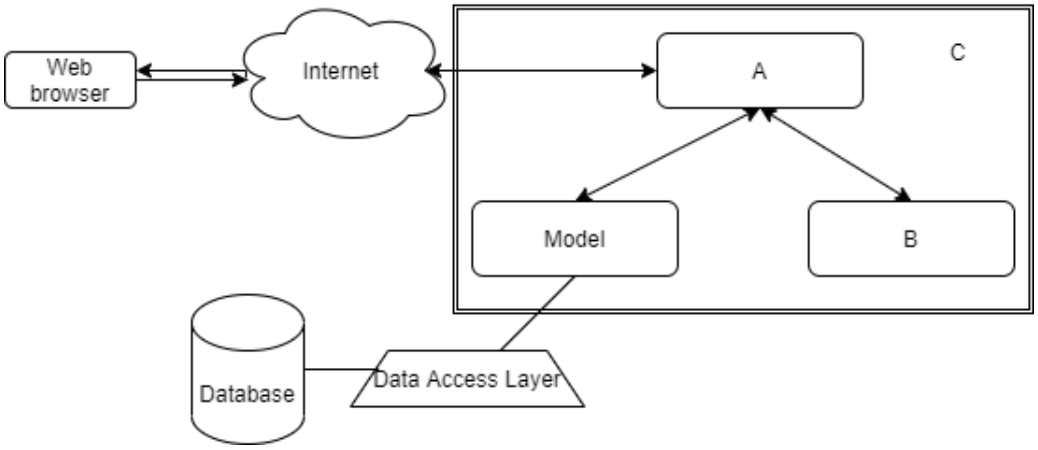


Figure 1: Web/Application Server

◯ A - View, B - Controller, C - Business Logic Layer,

◯ A - View, B - Controller, C - Presentation Layer

◯ A - Controller, B - View, C - Business Logic Layer

√ A - Controller, B - View, C - Presentation Layer

Solution: Presentation Layer is constituted by MVC architecture where M is model, V is view, C is controller.

- The Controller component acts as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output.
- The Model component corresponds to all the data-related logic that the user works with.
- The View component is used for all the UI logic of the application.

4. Which among the following is a Python library for parsing HTML? [MCQ: 2 points]

- ◯ Requests
- √ Beautiful Soup
- ◯ Paramiko
- ◯ Twisted Python

Solution: Beautiful Soup is an HTML parser that can handle all sorts of HTML.
Requests is a powerful HTTP client library.
Paramiko is used for implementing the SSH2 protocol.
Twisted Python is a framework for asynchronous network programming.

5. Which of the following is/are the disadvantage(s) of using single factor authentication with password? [MSQ: 2 points]

- √ The password can be disclosed by guessing or sniffing of packets, if passwords are not encrypted.
- ◯ Using single factor authentication can be a time-consuming process, as it involves multiple steps like password plus one-time password sent by SMS.
- √ Passwords can be exposed if passwords are reused by a user across sites.
- √ Passwords can be captured by the specially designed spyware.

Solution: Please refer to slide: 35.13

6. Match the appropriate statements for

   a. 1-tier architecture

   b. 2-tier architecture

   c. 3-tier architecture

   d. n-tier architecture

   Statement-1: It distributes different components of the 3-tiers between different servers and add interface tiers to enable interactions and load balancing.
   Statement-2: It keeps all the components of an application on a single server or platform.
   Statement-3: It separates its tiers as Presentation, Logical and Data Access.
   Statement-4: It is based on client-server architecture, where all the interactions between client and server take place directly, without presence of any intermediate.

   [MSQ: 2 points]

   ◯ a)-statement-1, b)-statement-3, c)-statement-2, and d)-statement-4.

   ◯ a)-statement-4, b)-statement-2, c)-statement-3, and d)-statement-1.

   ◯ a)-statement-4, b)-statement-3, c)-statement-2, and d)-statement-1.

   √ a)-statement-2, b)-statement-4, c)-statement-3, and d)-statement-1.

   Solution: Please refer to slide: 31.17 - 31.20.

7. Which of the following tasks is/are performed by a web server?
[MSQ: 2 points]

◯ It has the core software component, named as rendering engine, that transforms HTML documents and other resources of a web page into an interactive visual representation on a user's device.

√ It receives HTTP/HTTPS requests, and responds to the requests with the content of that requested resource or an error message.

√ If the requested document is an executable program, it executes the program, and sends back the HTML document that is generated.

◯ It is used to access World Wide Web, and it can fetch content from the Web and display it on a user's device.

Solution:

- A web server is software and underlying hardware that accepts requests via HTTP or its secure variant, HTTPS.
- A web browser or crawler, requests for a specific resource using HTTP, and the server responds with the content of that resource or an error message.
- When a web server receives a request for a document which is an executable program, it executes the program, and sends back the HTML document that is generated.

8. Which of the following tasks is/are widely performed by a client-side script?
[MSQ: 2 points]

- √ It can check input validity of Web pages to avoid many round trips to server.
- ◯ It can make any system call, and can also access the file system of the host machine to perform read, write operations.
- √ It can provide rich user interface.
- ◯ It always executes at server-end and sends the result back to the client-end.

Solution: Client-side scripts are widely used to:

- forms basis of new generation of Web applications (called Web 2.0 applications) offering rich user interfaces
- check input for validity

However, in general, client-side scripts are

- firstly downloaded at the client-end and then interpreted and executed by the browser.
- not allowed to make any system calls directly, and disallowed with dangerous actions such as file writes.

9. Which of the following is not true about ODBC?
[MSQ: 2 points]

◯ ODBC is designed with an objective to support various Windows versions. Thus, it is not supported by the non-Windows operating systems.

√ ODBC is a standard API for database connectivity, used by the application programs to communicate with database servers.

◯ ODBC is a Java-based technology developed by Sun Microsystems.

√ An application written using ODBC can be easily ported to heterogeneous client and server platforms.

Solution: Please refer to Slide: 33.8

10. Consider the following tasks in a Java program to execute an SQL query at database server using JDBC.

    1. Create a "Statement" object
    2. Use the "Statement" object to execute the SQL statement
    3. Create a "Connection" object
    4. Fetch the query results in a "ResultSet" object

    Identify the correct order in which the tasks must be performed. [MCQ: 2 points]

    ◯ $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$

    ◯ $2 \rightarrow 1 \rightarrow 3 \rightarrow$

    √ $3 \rightarrow 2 \rightarrow 1 \rightarrow 4$

    ◯ $3 \rightarrow 1 \rightarrow 2 \rightarrow 4$

    Solution: The appropriate order must be:

    1. Create a "Connection" object
    2. Create a "Statement" object
    3. Use the "Statement" object to execute the SQL statement
    4. Fetch the query results in a "ResultSet" object

11. Among the given options, which is/are a challenge in Web Application development? [MSQ: 2 points]

- ◯ The number of rows that have been affected (modified, inserted, or deleted) by the last `execute()` procedure.
- √ Knowledge of framework and platforms
- ◯ Limited computing power
- √ Web security threats
- ◯ Limited memory

Solution: The challenges for Web application development are:

- User interface and user experience
- Scalability
- Performance
- Knowledge of framework and platforms
- Security

Limited computing power and limited memory are the challenges for Mobile application development.