Amol Gavit 250240320013

1. **Create minimum 5 tables without any Key constraints(5)**

CREATE TABLE STUDENTS (

STUDENTID INT,

STUDENTNAME VARCHAR(30),

AGE INT,

CLASS VARCHAR(10)

);

```
mysql> USE EXAM;
Database changed
mysql> CREATE TABLE STUDENTS (
    -> STUDENTID INT,
    -> STUDENTNAME VARCHAR(30),
    -> AGE INT,
    -> CLASS VARCHAR(10)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

CREATE TABLE TEACHERS(

TEACHERID INT,

TEACHERNAME VARCHAR(30),

SUBJECT VARCHAR(30)

);

```
mysql> CREATE TABLE TEACHERS(
    -> TEACHERID INT,
    -> TEACHERNAME VARCHAR(30),
    -> SUBJECT VARCHAR(30)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

CREATE TABLE CLASSES(

CLASSID INT,

CLASSNAME VARCHAR(10),

SECTION VARCHAR(2)

);

```
mysql> CREATE TABLE CLASSES(
    -> CLASSID INT,
    -> CLASSNAME VARCHAR(10),
    -> SECTION VARCHAR(2)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

CREATE TABLE SUBJECTS(

SUBJECTID INT,

SUBJECTNAME VARCHAR(40)

);

```
mysql> CREATE TABLE SUBJECTS(
    -> SUBJECTID INT,
    -> SUBJECTNAME VARCHAR(40)
    -> );
Query OK, 0 rows affected (0.03 sec)
```

CREATE TABLE MARKS(

MARKSID INT,

SUDENTID INT,

SUBJECTID INT,

MARKS INT

);

```
mysql> CREATE TABLE MARKS(
    -> MARKSID INT,
    -> SUDENTID INT,
    -> SUBJECTID INT,
    -> MARKS INT
    -> );
Query OK, 0 rows affected (0.03 sec)
```

2. Add Primary Key to each table(3.5)

ALTER TABLE STUDENTS ADD PRIMARY KEY (STUDENTID);

```
mysql> ALTER TABLE STUDENTS ADD PRIMARY KEY (STUDENTID);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

ALTER TABLE TEACHERS ADD PRIMARY KEY (TEACHERID);

```
mysql> ALTER TABLE TEACHERS ADD PRIMARY KEY (TEACHERID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

ALTER TABLE CLASSES ADD PRIMARY KEY (CLASSID);

```
mysql> ALTER TABLE CLASSES ADD PRIMARY KEY (CLASSID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

ALTER TABLE SUBJECTS ADD PRIMARY KEY (SUBJECTID);

```
mysql> ALTER TABLE SUBJECTS ADD PRIMARY KEY (SUBJECTID);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

ALTER TABLE MARKS ADD PRIMARY KEY (MARKSID);

```
mysql> ALTER TABLE MARKS ADD PRIMARY KEY (MARKSID);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

3. Connect atleast 3 tables using Foreign Key(2)

ALTER TABLE MARKS ADD FOREIGN KEY (SUDENTID) REFERENCES STUDENTS(STUDENTID);

```
mysql> ALTER TABLE MARKS ADD FOREIGN KEY (SUDENTID) REFERENCES STUDENTS(STUD
ENTID);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

ALTER TABLE MARKS ADD FOREIGN KEY (SUBJECTID) REFERENCES SUBJECTS(SUBJECTID);

```
mysql> ALTER TABLE MARKS ADD FOREIGN KEY (SUBJECTID) REFERENCES SUBJECTS(SUB
JECTID);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

4. Insert minimum five records in each of the table(3.5)

INSERT INTO STUDENTS (STUDENTID, STUDENTNAME, AGE, CLASS)

VALUES

(1, 'ROBIN', 19, 'A1'),

(2, 'ABDULLA', 20, 'C2'),

(3, 'PURVA', 19, 'A3'),

(4, 'RUJUL', 18, 'B2'),

(5, 'JIVITA', 23, 'A1');

```
mysql> INSERT INTO STUDENTS (STUDENTID, STUDENTNAME, AGE, CLASS)
    -> VALUES
    -> (1, 'ROBIN', 19, 'A1'),
    -> (2, 'ABDULLA', 20, 'C2'),
    -> (3, 'PURVA', 19, 'A3'),
    -> (4, 'RUJUL', 18, 'B2'),
    -> (5, 'JIVITA', 23, 'A1');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

INSERT INTO TEACHERS (TEACHERID, TEACHERNAME, SUBJECT)

VALUES

(1, 'AMOL', 'MATH'),

(2, 'OMKAR', 'MARATHI'),

(3, 'VISHAL', 'ENGLISH'),

(4, 'SAGAR', 'HISTORY'),

(5, 'PRIYANKA', 'HINDI');

```
mysql> INSERT INTO TEACHERS (TEACHERID, TEACHERNAME, SUBJECT)
    -> VALUES
    -> (1, 'AMOL', 'MATH'),
    -> (2, 'OMKAR', 'MARATHI'),
    -> (3, 'VISHAL', 'ENGLISH'),
    -> (4, 'SAGAR', 'HISTORY'),
    -> (5, 'PRIYANKA', 'HINDI');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
INSERT INTO CLASSES (CLASSID, CLASSNAME, SECTION)
VALUES
(1, '2', 'A'),
(2, '3', 'C'),
(3, '1', 'B'),
(4, '3', 'A'),
(5, '2', 'C');
```

```
mysql> INSERT INTO CLASSES (CLASSID, CLASSNAME, SECTION)
    -> VALUES
    -> (1, '2', 'A'),
    -> (2, '3', 'C'),
    -> (3, '1', 'B'),
    -> (4, '3', 'A'),
    -> (5, '2', 'C');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
INSERT INTO SUBJECTS (SUBJECTID, SUBJECTNAME)
VALUES
(1, 'MATH'),
(2, 'MARATHI'),
(3, 'HINDI'),
(4, 'ENGLISH'),
(5, 'HISTORY');
```

```
mysql> INSERT INTO SUBJECTS (SUBJECTID, SUBJECTNAME)
    -> VALUES
    -> (1, 'MATH'),
    -> (2, 'MARATHI'),
    -> (3, 'HINDI'),
    -> (4, 'ENGLISH'),
    -> (5, 'HISTORY');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql>
```

INSERT INTO MARKS (MARKSID, SUDENTID, SUBJECTID, MARKS)

VALUES

(1, 1, 1, 75),

(2, 2, 2, 60),

(3, 3, 3, 35),

(4, 4, 3, 95),

(5, 5, 4, 83);

```
mysql> INSERT INTO MARKS (MARKSID, SUDENTID, SUBJECTID, MARKS)
    -> VALUES
    -> (1, 1, 1, 75),
    -> (2, 2, 2, 60),
    -> (3, 3, 3, 35),
    -> (4, 4, 3, 95),
    -> (5, 5, 4, 83);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

5. Write an appropriate command to Use of Where with OR clause (2)

SELECT * FROM SUBJECTS

WHERE SUBJECTID = 2 OR SUBJECTNAME = 'ENGLISH';

```
mysql> SELECT * FROM SUBJECTS
    -> WHERE SUBJECTID = 2 OR SUBJECTNAME = 'ENGLISH';
+-----------+-------------+
| SUBJECTID | SUBJECTNAME |
+-----------+-------------+
|         2 | MARATHI     |
|         4 | ENGLISH     |
+-----------+-------------+
2 rows in set (0.00 sec)
```

6. Write a command to show first five records from any one table (2)

SELECT * FROM MARKS

LIMIT 5;

```
mysql> SELECT * FROM MARKS
    -> LIMIT 5;
+---------+----------+-----------+-------+
| MARKSID | SUDENTID | SUBJECTID | MARKS |
+---------+----------+-----------+-------+
|       1 |        1 |         1 |    75 |
|       2 |        2 |         2 |    60 |
|       3 |        3 |         3 |    35 |
|       4 |        4 |         3 |    95 |
|       5 |        5 |         4 |    83 |
+---------+----------+-----------+-------+
5 rows in set (0.00 sec)
```

7. Write a command for appropriate and logical use of Group By clause(2)

SELECT SUBJECTID, AVG(MARKS) FROM MARKS

GROUP BY SUBJECTID;

```
mysql> SELECT SUBJECTID, AVG(MARKS) FROM MARKS
    -> GROUP BY SUBJECTID;
+-----------+------------+
| SUBJECTID | AVG(MARKS) |
+-----------+------------+
|         1 |    75.0000 |
|         2 |    60.0000 |
|         3 |    65.0000 |
|         4 |    83.0000 |
+-----------+------------+
4 rows in set (0.00 sec)
```

8. Write a query to show the name of 2nd highest marks of student along with its cost(2)

SELECT SUDENTID, MARKS FROM MARKS

ORDER BY MARKS DESC

LIMIT 1 OFFSET 1;

```
mysql> SELECT SUDENTID, MARKS FROM MARKS
    -> ORDER BY MARKS DESC
    -> LIMIT 1 OFFSET 1;
+----------+-------+
| SUDENTID | MARKS |
+----------+-------+
|        5 |    83 |
+----------+-------+
1 row in set (0.00 sec)
```

9. Create ONE simple view using any of the created table and update the data from view to reflect in base table (2)

CREATE VIEW VIEW_STUDENT_MARKS AS

SELECT SUDENTID, MARKS FROM MARKS;

```
SELECT SUDENTID, MARKS FROM MARKS' at line 1
mysql> CREATE VIEW VIEW_STUDENT_MARKS AS
    -> SELECT SUDENTID, MARKS FROM MARKS;
Query OK, 0 rows affected (0.01 sec)

mysql>
```

UPDATE VIEW_STUDENT_MARKS

SET MARKS = 63

WHERE SUDENTID = 4;

```
mysql> UPDATE VIEW_STUDENT_MARKS
    -> SET MARKS = 63
    -> WHERE SUDENTID = 4;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM MARKS;
+---------+----------+-----------+-------+
| MARKSID | SUDENTID | SUBJECTID | MARKS |
+---------+----------+-----------+-------+
|       1 |        1 |         1 |    75 |
|       2 |        2 |         2 |    60 |
|       3 |        3 |         3 |    35 |
|       4 |        4 |         3 |    63 |
|       5 |        5 |         4 |    83 |
+---------+----------+-----------+-------+
5 rows in set (0.00 sec)
```

10. Write a command to use LEFT JOIN (2)

SELECT S.STUDENTNAME, M.MARKS FROM STUDENTS S

LEFT JOIN MARKS M ON M. SUDENTID = S.STUDENTID;

```
mysql> SELECT S.STUDENTNAME, M.MARKS FROM STUDENTS S
    -> LEFT JOIN MARKS M ON M. SUDENTID = S.STUDENTID;
+-------------+-------+
| STUDENTNAME | MARKS |
+-------------+-------+
| ROBIN       |    75 |
| ABDULLA     |    60 |
| PURVA       |    35 |
| RUJUL       |    63 |
| JIVITA      |    83 |
+-------------+-------+
5 rows in set (0.00 sec)

mysql>
```

11. Create the temporary table showing only Student name. How to remove this table permnantly? (2)

CREATE TEMPORARY TABLE TEMP_STUDENTS AS

SELECT STUDENTNAME FROM STUDENTS;

```
mysql> CREATE TEMPORARY TABLE TEMP_STUDENTS AS
    -> SELECT STUDENTNAME FROM STUDENTS;
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM TEMP_STUDENTS;
+-------------+
| STUDENTNAME |
+-------------+
| ROBIN       |
| ABDULLA     |
| PURVA       |
| RUJUL       |
| JIVITA      |
+-------------+
5 rows in set (0.00 sec)
```

DROP TABLE TEMP_STUDENTS;

```
mysql> DROP TABLE TEMP_STUDENTS;
Query OK, 0 rows affected (0.00 sec)

mysql>
```

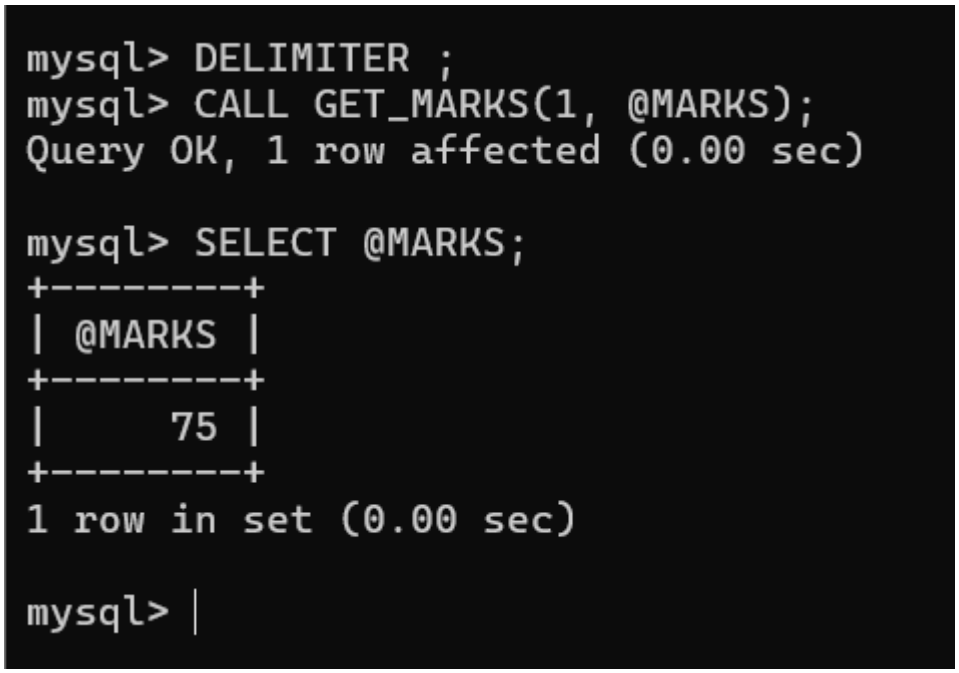
12. Create at least one procedure (with one INPUT and one OUTPUT parameter) based on the created tables with appropriate utility (4)

```
DELIMITER //

CREATE PROCEDURE GET_MARKS( IN STUDENT_ID INT, OUT MARKS_OUT INT)
BEGIN
        SELECT MARKS  INTO MARKS_OUT FROM MARKS
        WHERE STUDENT_ID = SUDENTID;
END //

DELIMITER ;
```

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE GET_MARKS( IN STUDENT_ID INT, OUT MARKS_OUT INT)
    -> BEGIN
    ->  SELECT MARKS  INTO MARKS_OUT FROM MARKS
    ->  WHERE STUDENT_ID = SUDENTID;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
```

CALL GET_MARKS(1, @MARKS);

SELECT @MARKS;



14. Create a stored function name DISC_CALC that takes two parameters:- ORIGINAL_PRICE and DISCOUNT_PERCENTAGE. The function should calculate the discounted price based on the original price and the discount percentage. The function should return the discounted price.(4)

```
DELIMITER //

CREATE FUNCTION DISC_CALC(
ORIGINAL_PRICE FLOAT(10,2),
DISCOUNT_PERCENTAGE FLOAT(10,2)
)
RETURNS FLOAT(10,2)
DETERMINISTIC
BEGIN
        DECLARE DISC_PRICE FLOAT(10,2);
SET DISC_PRICE = ORIGINAL_PRICE - ((ORIGINAL_PRICE *
DISCOUNT_PERCENTAGE)/100);
RETURN DISC_PRICE;

END //

DELIMITER ;
```

```
END' at line 6
mysql> CREATE FUNCTION DISC_CALC(
    -> ORIGINAL_PRICE FLOAT(10,2),
    -> DISCOUNT_PERCENTAGE FLOAT(10,2)
    -> )
    -> RETURNS FLOAT(10,2)
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE DISC_PRICE FLOAT(10,2);
    -> SET DISC_PRICE = ORIGINAL_PRICE - ((ORIGINAL_PRICE * DISCOUNT_PERCENT
AGE)/100);
    -> RETURN DISC_PRICE;
    ->
    -> END //
Query OK, 0 rows affected, 4 warnings (0.01 sec)

mysql> DELIMITER ;
mysql> select * DISC_CALC(100, 10);
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual t
hat corresponds to your MySQL server version for the right syntax to use nea
r 'DISC_CALC(100, 10)' at line 1
mysql> select DISC_CALC(100, 10);
+--------------------+
| DISC_CALC(100, 10) |
+--------------------+
|              90.00 |
+--------------------+
1 row in set (0.00 sec)
```