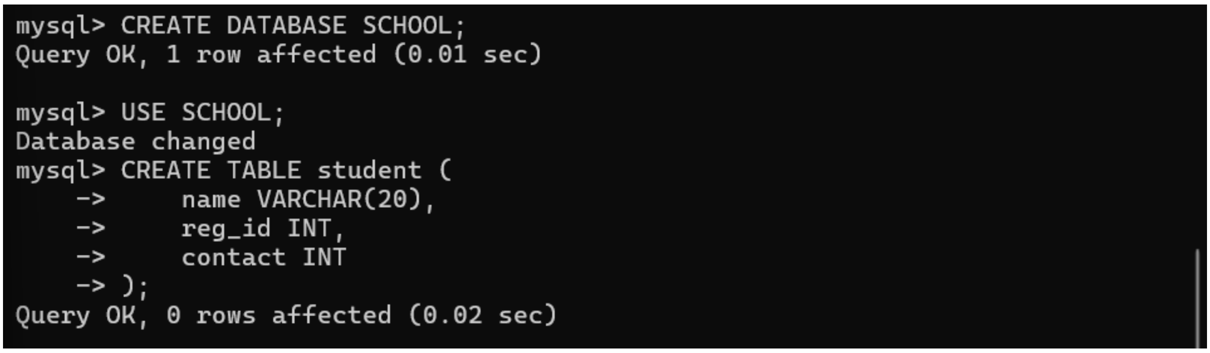
FULL STACK APPLICATION DEVELOPMENT (week 1)

TASK-1

A) Create table using DDL command:

Create table:

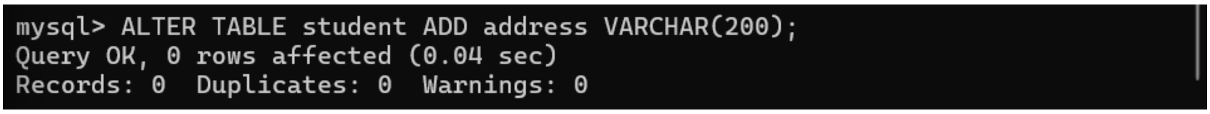
```
CREATE TABLE student (
   name VARCHAR(20),
   reg_id INT,
   contact INT
);
```



```
mysql> CREATE DATABASE SCHOOL;
Query OK, 1 row affected (0.01 sec)

mysql> USE SCHOOL;
Database changed
mysql> CREATE TABLE student (
    ->     name VARCHAR(20),
    ->     reg_id INT,
    ->     contact INT
    -> );
Query OK, 0 rows affected (0.02 sec)
```

Alter add:

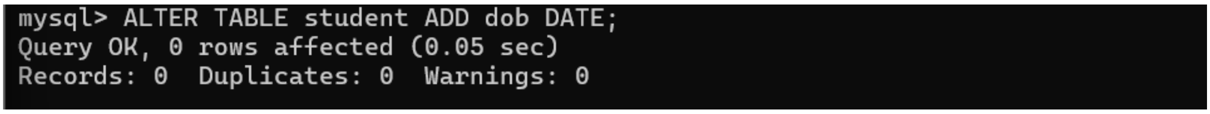
```
ALTER TABLE student ADD address VARCHAR(200);
```



```
mysql> ALTER TABLE student ADD address VARCHAR(200);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Alter default data add:

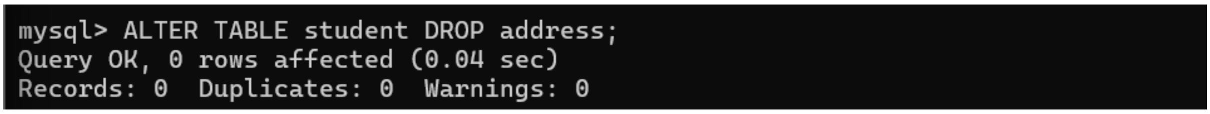
```
ALTER TABLE student ADD dob DATE DEFAULT '02-FEB-2010';
```



```
mysql> ALTER TABLE student ADD dob DATE;
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Alter drop:

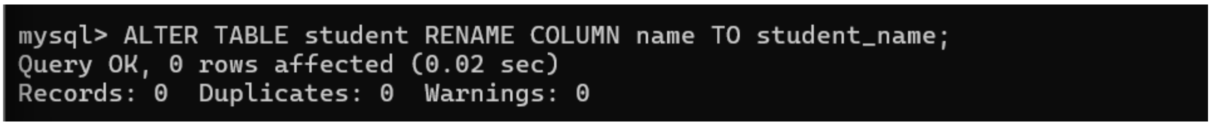
```
ALTER TABLE student DROP address;
```



```
mysql> ALTER TABLE student DROP address;
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Alter rename:

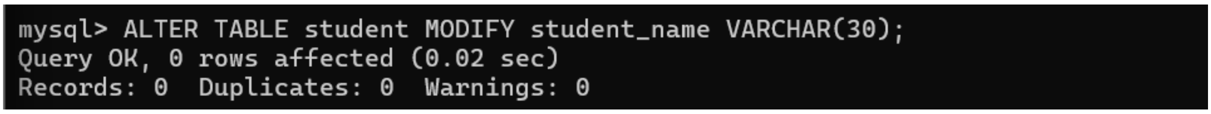
ALTER TABLE student RENAME COLUMN name TO student_name;



```
mysql> ALTER TABLE student RENAME COLUMN name TO student_name;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Alter Modify:

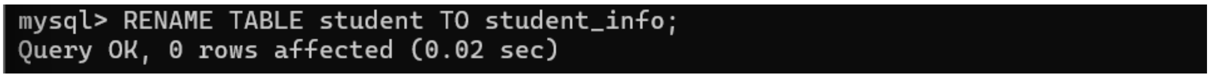
ALTER TABLE student MODIFY student_name VARCHAR(30);



```
mysql> ALTER TABLE student MODIFY student_name VARCHAR(30);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Rename table:

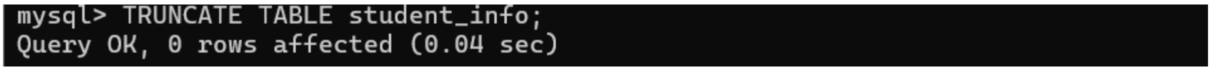
RENAME student TO student_info;



```
mysql> RENAME TABLE student TO student_info;
Query OK, 0 rows affected (0.02 sec)
```

Truncate table:

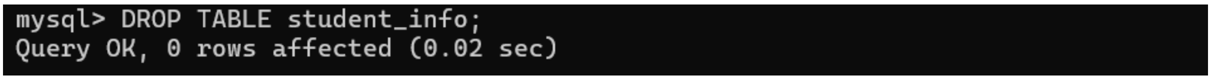
TRUNCATE TABLE student_info;



```
mysql> TRUNCATE TABLE student_info;
Query OK, 0 rows affected (0.04 sec)
```

Drop table:

DROP TABLE student_info;



```
mysql> DROP TABLE student_info;
Query OK, 0 rows affected (0.02 sec)
```

B) DDL Constrains:

Null command:

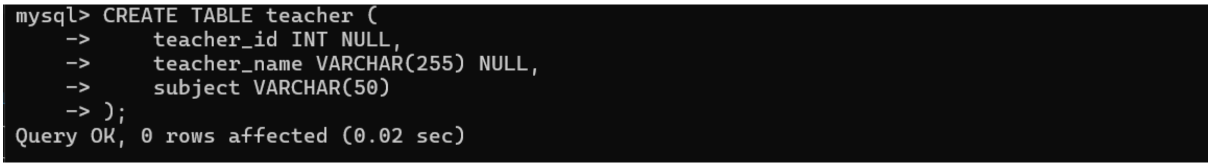
CREATE TABLE teacher (

teacher_id INT NULL,

teacher_name VARCHAR(255) NULL,

subject VARCHAR(50)

);



```
mysql> CREATE TABLE teacher (
    ->     teacher_id INT NULL,
    ->     teacher_name VARCHAR(255) NULL,
    ->     subject VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.02 sec)
```

Not null command:

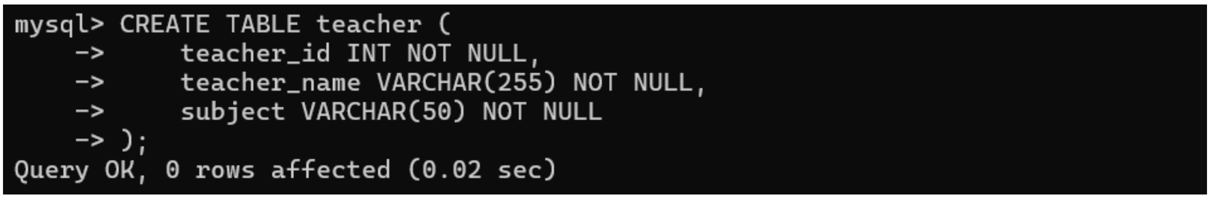
CREATE TABLE teacher (

teacher_id INT NOT NULL,

teacher_name VARCHAR(255) NOT NULL,

subject VARCHAR(50) NOT NULL

);



```
mysql> CREATE TABLE teacher (
    ->     teacher_id INT NOT NULL,
    ->     teacher_name VARCHAR(255) NOT NULL,
    ->     subject VARCHAR(50) NOT NULL
    -> );
Query OK, 0 rows affected (0.02 sec)
```

Check command:

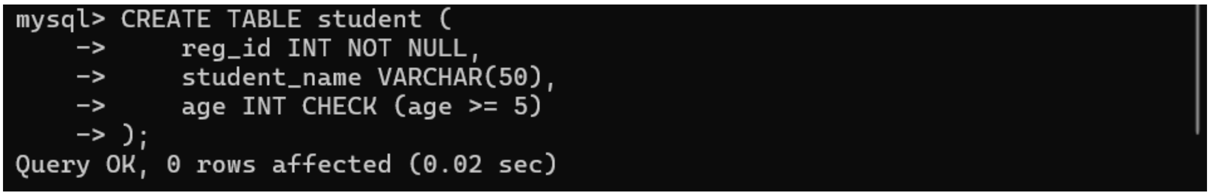
CREATE TABLE student (

reg_id INT NOT NULL,

student_name VARCHAR(50),

age INT CHECK (age >= 5)

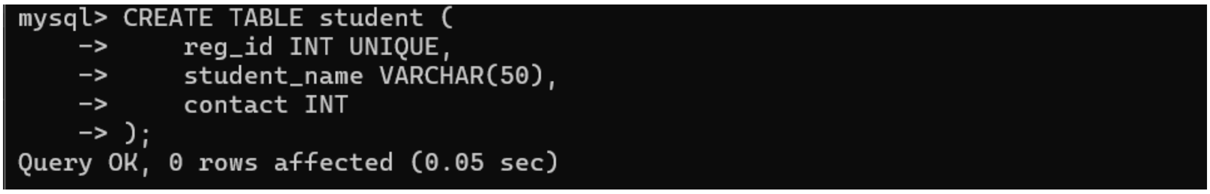
);



```
mysql> CREATE TABLE student (
    ->     reg_id INT NOT NULL,
    ->     student_name VARCHAR(50),
    ->     age INT CHECK (age >= 5)
    -> );
Query OK, 0 rows affected (0.02 sec)
```

Unique:

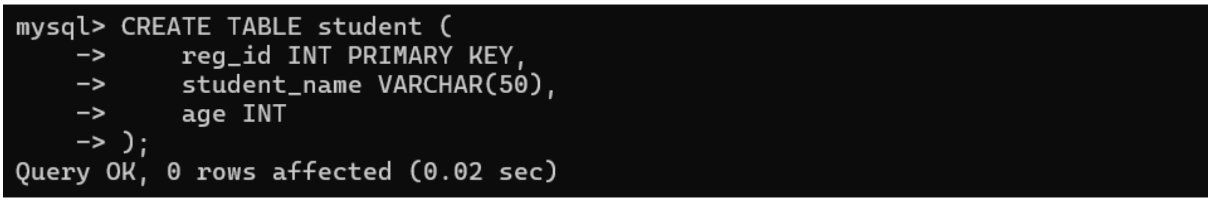
```
CREATE TABLE student (
  reg_id INT UNIQUE,
  student_name VARCHAR(50),
  contact INT
);
```



```
mysql> CREATE TABLE student (
    ->     reg_id INT UNIQUE,
    ->     student_name VARCHAR(50),
    ->     contact INT
    -> );
Query OK, 0 rows affected (0.05 sec)
```

Primary key:

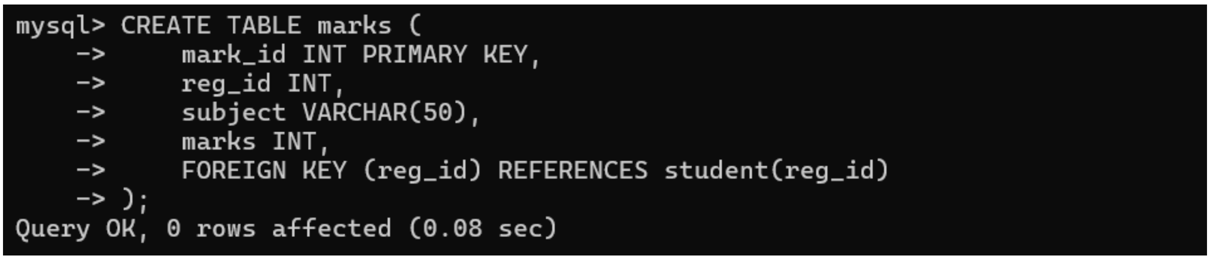
```
CREATE TABLE student (
   reg_id INT PRIMARY KEY,
   student_name VARCHAR(50),
   age INT
 );
```



```
mysql> CREATE TABLE student (
    ->     reg_id INT PRIMARY KEY,
    ->     student_name VARCHAR(50),
    ->     age INT
    -> );
Query OK, 0 rows affected (0.02 sec)
```

Foreign Key:

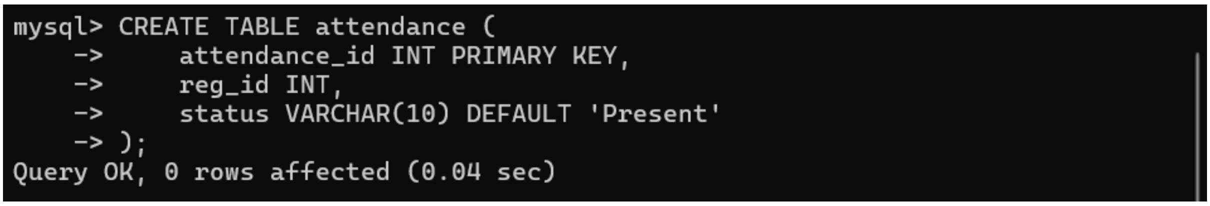
```
CREATE TABLE marks (
    mark_id INT PRIMARY KEY,
    reg_id INT,
    subject VARCHAR(50),
    marks INT,
    FOREIGN KEY (reg_id) REFERENCES student(reg_id)
);
```



```
mysql> CREATE TABLE marks (
    ->     mark_id INT PRIMARY KEY,
    ->     reg_id INT,
    ->     subject VARCHAR(50),
    ->     marks INT,
    ->     FOREIGN KEY (reg_id) REFERENCES student(reg_id)
    -> );
Query OK, 0 rows affected (0.08 sec)
```

Default:

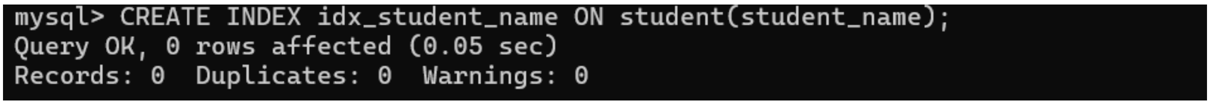
```
CREATE TABLE attendance (
  attendance_id INT PRIMARY KEY,
  reg_id INT,
  status VARCHAR(10) DEFAULT 'Present'
);
```



```
mysql> CREATE TABLE attendance (
    ->     attendance_id INT PRIMARY KEY,
    ->     reg_id INT,
    ->     status VARCHAR(10) DEFAULT 'Present'
    -> );
Query OK, 0 rows affected (0.04 sec)
```

index:

```
CREATE INDEX idx_student_name ON student(student_name);
```

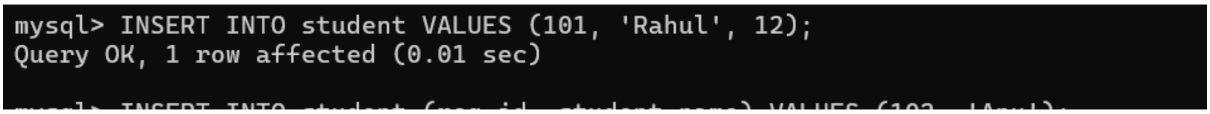


```
mysql> CREATE INDEX idx_student_name ON student(student_name);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

C) DML Quries:

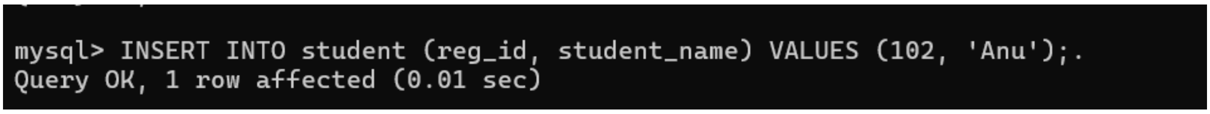
insert command:

```
INSERT INTO student VALUES (101, 'Rahul', 12);
```



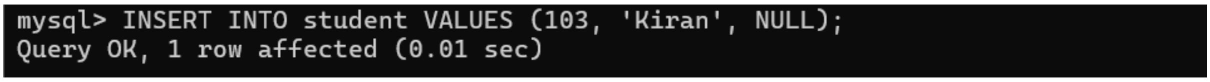
```
mysql> INSERT INTO student VALUES (101, 'Rahul', 12);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO student (reg_id, student_name) VALUES (102, 'Anu');
```

```
INSERT INTO student (reg_id, student_name) VALUES (102, 'Anu');
```



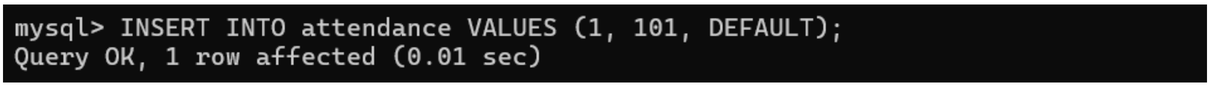
```
mysql> INSERT INTO student (reg_id, student_name) VALUES (102, 'Anu');.
Query OK, 1 row affected (0.01 sec)
```

```
INSERT INTO student VALUES (103, 'Kiran', NULL);
```



```
mysql> INSERT INTO student VALUES (103, 'Kiran', NULL);
Query OK, 1 row affected (0.01 sec)
```

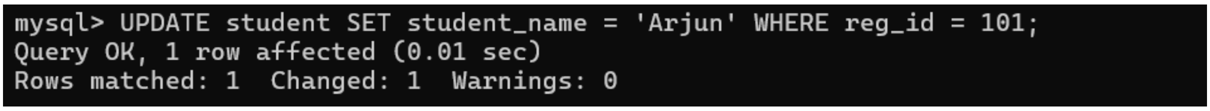
INSERT INTO attendance VALUES (1, 101, DEFAULT);



```
mysql> INSERT INTO attendance VALUES (1, 101, DEFAULT);
Query OK, 1 row affected (0.01 sec)
```

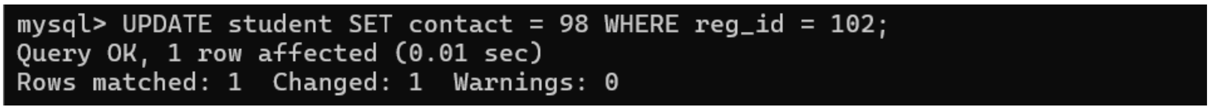
update command:

UPDATE student SET student_name = 'Arjun' WHERE reg_id = 101;



```
mysql> UPDATE student SET student_name = 'Arjun' WHERE reg_id = 101;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

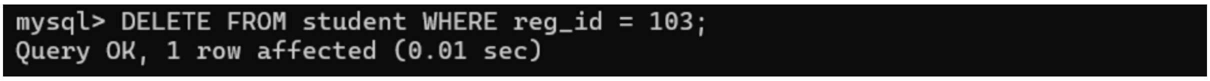
UPDATE student SET contact = 9876543210 WHERE reg_id = 102;



```
mysql> UPDATE student SET contact = 98 WHERE reg_id = 102;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

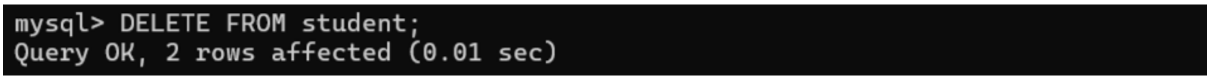
Delete command:

DELETE FROM student WHERE reg_id = 103;



```
mysql> DELETE FROM student WHERE reg_id = 103;
Query OK, 1 row affected (0.01 sec)
```

DELETE FROM student;



```
mysql> DELETE FROM student;
Query OK, 2 rows affected (0.01 sec)
```

TASK-2

1. Aggregation function:

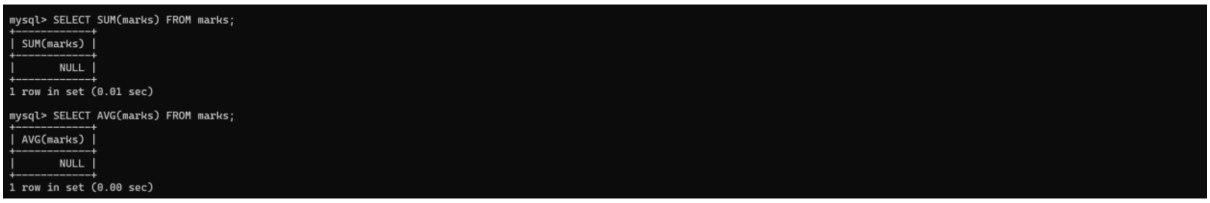
SELECT SUM(marks) FROM marks;

SELECT AVG(marks) FROM marks;

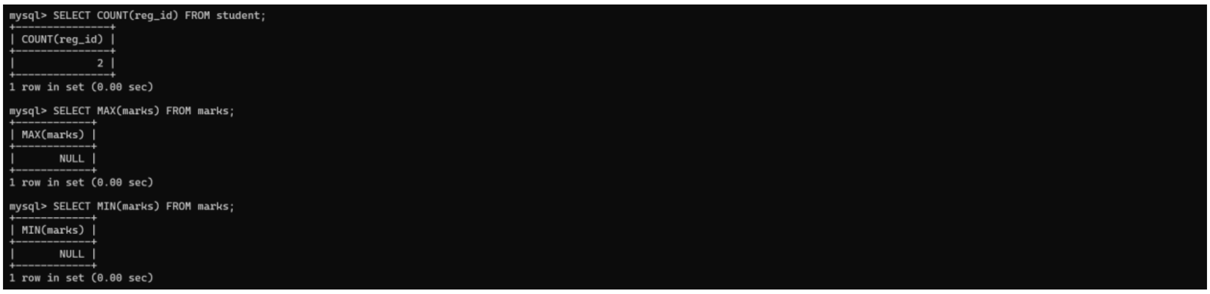
SELECT COUNT(reg_id) FROM student;

SELECT MAX(marks) FROM marks;

SELECT MIN(marks) FROM marks;



```
mysql> SELECT SUM(marks) FROM marks;
+------------+
| SUM(marks) |
+------------+
|       NULL |
+------------+
1 row in set (0.01 sec)

mysql> SELECT AVG(marks) FROM marks;
+------------+
| AVG(marks) |
+------------+
|       NULL |
+------------+
1 row in set (0.00 sec)
```



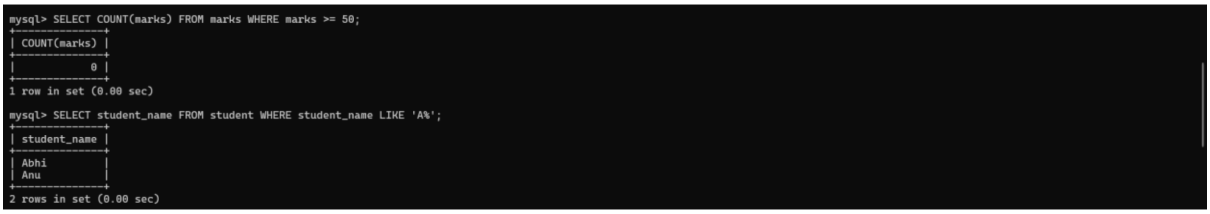
```
mysql> SELECT COUNT(reg_id) FROM student;
+---------------+
| COUNT(reg_id) |
+---------------+
|             2 |
+---------------+
1 row in set (0.00 sec)

mysql> SELECT MAX(marks) FROM marks;
+------------+
| MAX(marks) |
+------------+
|       NULL |
+------------+
1 row in set (0.00 sec)

mysql> SELECT MIN(marks) FROM marks;
+------------+
| MIN(marks) |
+------------+
|       NULL |
+------------+
1 row in set (0.00 sec)
```

2. <u>Clause:</u>

   SELECT COUNT(marks) FROM marks WHERE marks >= 50;

   Like:

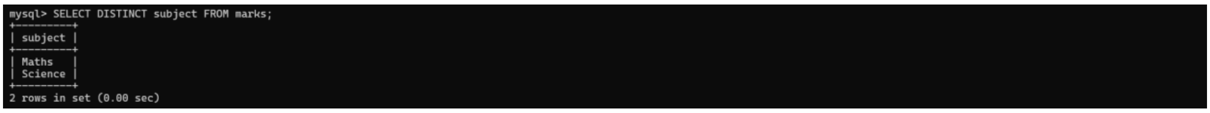
   SELECT student_name FROM student WHERE student_name LIKE 'A%';



```
mysql> SELECT COUNT(marks) FROM marks WHERE marks >= 50;
+--------------+
| COUNT(marks) |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)

mysql> SELECT student_name FROM student WHERE student_name LIKE 'A%';
+--------------+
| student_name |
+--------------+
| Abhi         |
| Anu          |
+--------------+
2 rows in set (0.00 sec)
```

Distinct:

SELECT DISTINCT subject FROM marks;



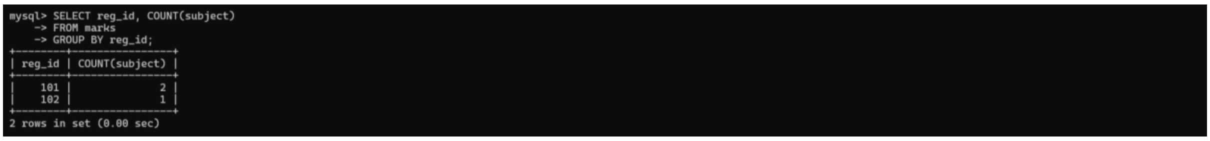
```
mysql> SELECT DISTINCT subject FROM marks;
+---------+
| subject |
+---------+
| Maths   |
| Science |
+---------+
2 rows in set (0.00 sec)
```

Group by:

SELECT reg_id, COUNT(subject)

FROM marks

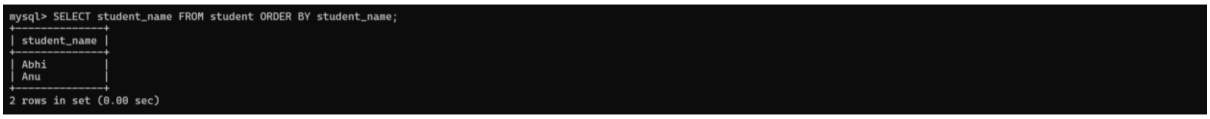
GROUP BY reg_id;



```
mysql> SELECT reg_id, COUNT(subject)
    -> FROM marks
    -> GROUP BY reg_id;
+--------+----------------+
| reg_id | COUNT(subject) |
+--------+----------------+
|    101 |              2 |
|    102 |              1 |
+--------+----------------+
2 rows in set (0.00 sec)
```

Order by:

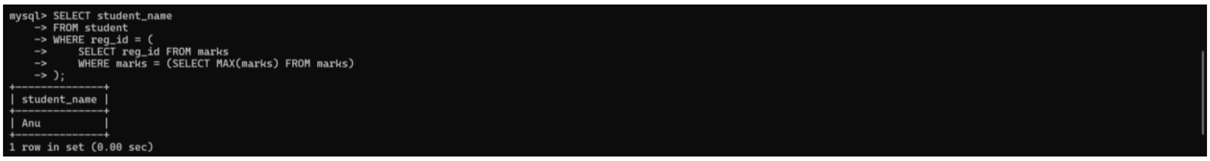
SELECT student_name FROM student ORDER BY student_name;



```
mysql> SELECT student_name FROM student ORDER BY student_name;
+--------------+
| student_name |
+--------------+
| Abhi         |
| Anu          |
+--------------+
2 rows in set (0.00 sec)
```

3. Nested Queries:

```
SELECT student_name
FROM student
WHERE reg_id = (
   SELECT reg_id FROM marks
   WHERE marks = (SELECT MAX(marks) FROM marks)
);
```

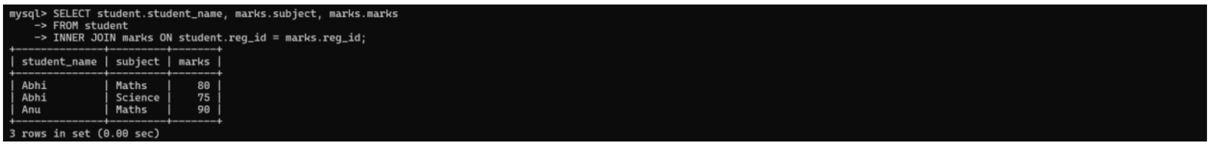


```
mysql> SELECT student_name
    -> FROM student
    -> WHERE reg_id = (
    ->     SELECT reg_id FROM marks
    ->     WHERE marks = (SELECT MAX(marks) FROM marks)
    -> );
+--------------+
| student_name |
+--------------+
| Anu          |
+--------------+
1 row in set (0.00 sec)
```

TASK-3

1. Inner join:

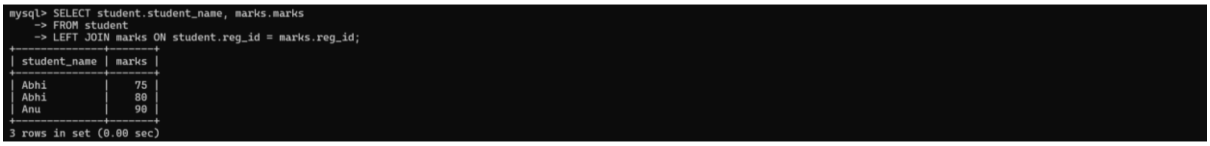
```
SELECT student.student_name, marks.subject, marks.marks
FROM student
INNER JOIN marks ON student.reg_id = marks.reg_id;
```



```
mysql> SELECT student.student_name, marks.subject, marks.marks
    -> FROM student
    -> INNER JOIN marks ON student.reg_id = marks.reg_id;
+--------------+---------+-------+
| student_name | subject | marks |
+--------------+---------+-------+
| Abhi         | Maths   |    80 |
| Abhi         | Science |    75 |
| Anu          | Maths   |    90 |
+--------------+---------+-------+
3 rows in set (0.00 sec)
```

2. Left join:

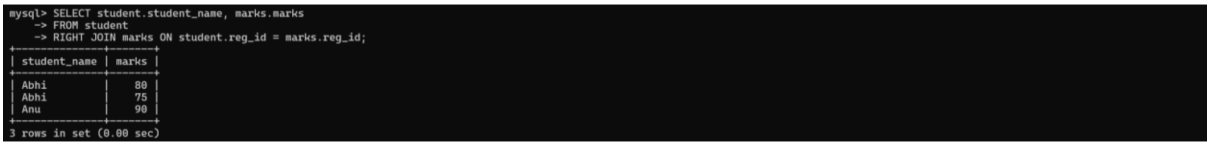
```
SELECT student.student_name, marks.marks
FROM student
LEFT JOIN marks ON student.reg_id = marks.reg_id;
```



```
mysql> SELECT student.student_name, marks.marks
    -> FROM student
    -> LEFT JOIN marks ON student.reg_id = marks.reg_id;
+--------------+-------+
| student_name | marks |
+--------------+-------+
| Abhi         |    75 |
| Abhi         |    80 |
| Anu          |    90 |
+--------------+-------+
3 rows in set (0.00 sec)
```

3. Right join:

```
SELECT student.student_name, marks.marks
FROM student
RIGHT JOIN marks ON student.reg_id = marks.reg_id;
```



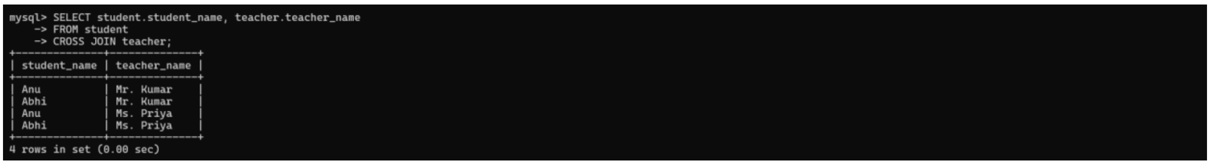
```
mysql> SELECT student.student_name, marks.marks
    -> FROM student
    -> RIGHT JOIN marks ON student.reg_id = marks.reg_id;
+--------------+-------+
| student_name | marks |
+--------------+-------+
| Abhi         |    80 |
| Abhi         |    75 |
| Anu          |    90 |
+--------------+-------+
3 rows in set (0.00 sec)
```

4. <u>Cross join:</u>

   SELECT student.student_name, teacher.teacher_name

   FROM student

   CROSS JOIN teacher;



```
mysql> SELECT student.student_name, teacher.teacher_name
    -> FROM student
    -> CROSS JOIN teacher;
+--------------+--------------+
| student_name | teacher_name |
+--------------+--------------+
| Anu          | Mr. Kumar    |
| Abhi         | Mr. Kumar    |
| Anu          | Ms. Priya    |
| Abhi         | Ms. Priya    |
+--------------+--------------+
4 rows in set (0.00 sec)
```

TASK-4

1. <u>Simple Stored Procedure (Print message):</u>

   DELIMITER //

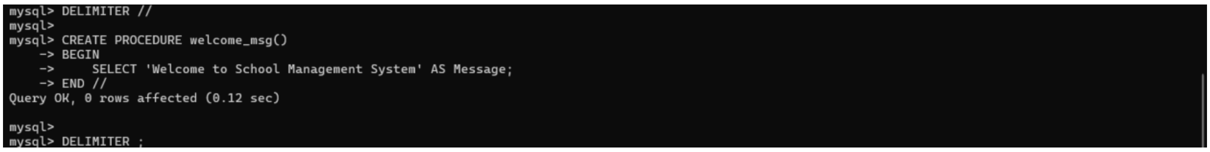
   CREATE PROCEDURE welcome_msg()

   BEGIN

     SELECT 'Welcome to School Management System' AS Message;

   END //

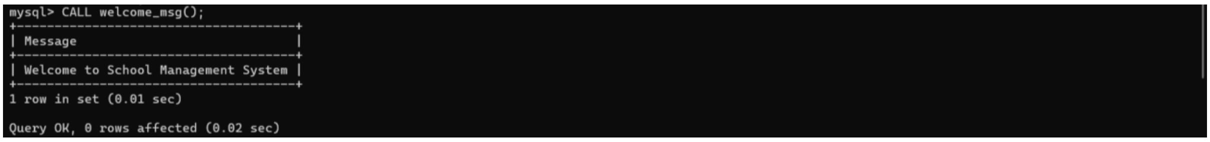
   DELIMITER ;



```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE welcome_msg()
    -> BEGIN
    ->     SELECT 'Welcome to School Management System' AS Message;
    -> END //
Query OK, 0 rows affected (0.12 sec)

mysql>
mysql> DELIMITER ;
```

Call procedure:

CALL welcome_msg();

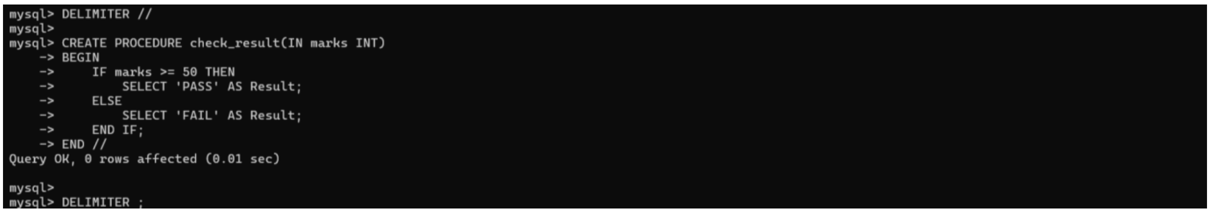


```
mysql> CALL welcome_msg();
+-------------------------------------+
| Message                             |
+-------------------------------------+
| Welcome to School Management System |
+-------------------------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.02 sec)
```

2. <u>Procedure with IF–ELSE (Pass / Fail):</u>

   DELIMITER //

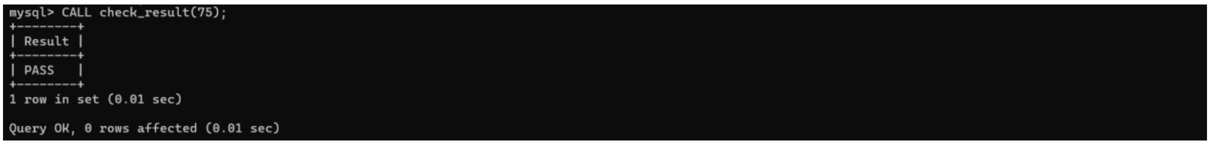
```
CREATE PROCEDURE check_result(IN marks INT)
BEGIN
  IF marks >= 50 THEN
    SELECT 'PASS' AS Result;
  ELSE
    SELECT 'FAIL' AS Result;
  END IF;
END //

DELIMITER ;
```



```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE check_result(IN marks INT)
    -> BEGIN
    ->     IF marks >= 50 THEN
    ->         SELECT 'PASS' AS Result;
    ->     ELSE
    ->         SELECT 'FAIL' AS Result;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
```

Call:

```
CALL check_result(75);
```

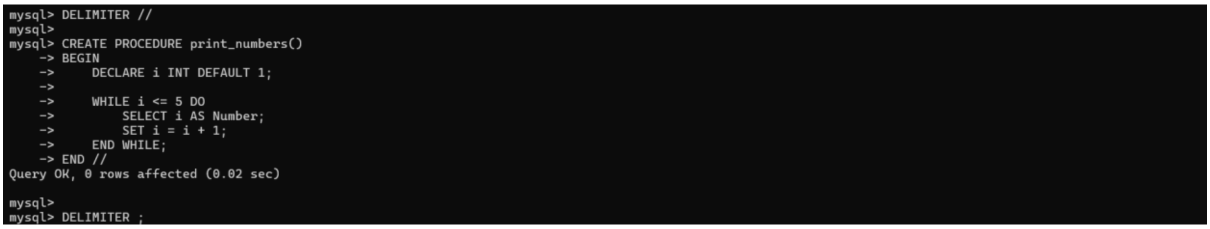


```
mysql> CALL check_result(75);
+--------+
| Result |
+--------+
| PASS   |
+--------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

3. Loop Program (Print numbers 1 to 5):

```
DELIMITER //

CREATE PROCEDURE print_numbers()
BEGIN
  DECLARE i INT DEFAULT 1;

  WHILE i <= 5 DO
    SELECT i AS Number;
    SET i = i + 1;
  END WHILE;
END //
```

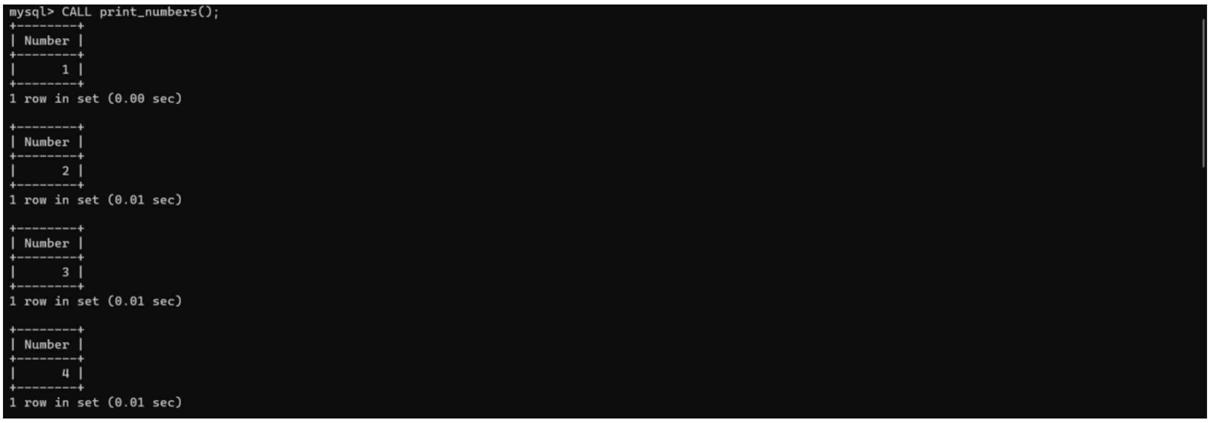
```
DELIMITER ;
```



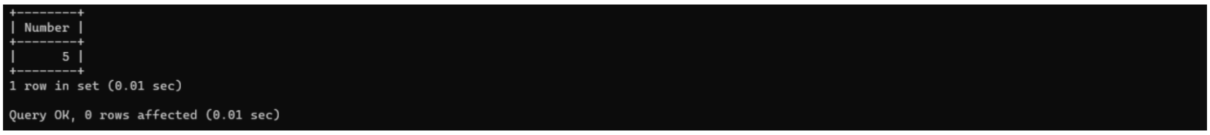
```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE print_numbers()
    -> BEGIN
    ->     DECLARE i INT DEFAULT 1;
    ->
    ->     WHILE i <= 5 DO
    ->         SELECT i AS Number;
    ->         SET i = i + 1;
    ->     END WHILE;
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> DELIMITER ;
```

Call:

```
CALL print_numbers();
```



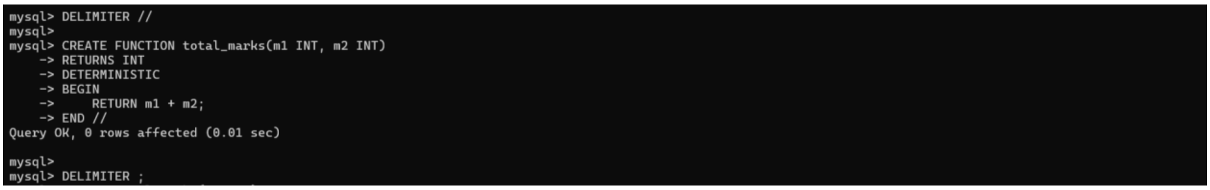
```
mysql> CALL print_numbers();
+--------+
| Number |
+--------+
|      1 |
+--------+
1 row in set (0.00 sec)

+--------+
| Number |
+--------+
|      2 |
+--------+
1 row in set (0.01 sec)

+--------+
| Number |
+--------+
|      3 |
+--------+
1 row in set (0.01 sec)

+--------+
| Number |
+--------+
|      4 |
+--------+
1 row in set (0.01 sec)
```



```
+--------+
| Number |
+--------+
|      5 |
+--------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

4. Function (Return Total Marks):

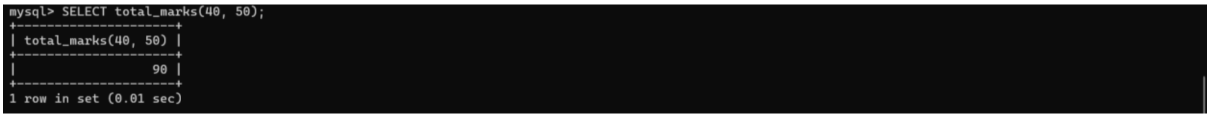
```
DELIMITER //


CREATE FUNCTION total_marks(m1 INT, m2 INT)
RETURNS INT
DETERMINISTIC
BEGIN
  RETURN m1 + m2;
END //


DELIMITER ;
```



```
mysql> DELIMITER //
mysql>
mysql> CREATE FUNCTION total_marks(m1 INT, m2 INT)
    -> RETURNS INT
    -> DETERMINISTIC
    -> BEGIN
    ->     RETURN m1 + m2;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
```

Use function:

SELECT total_marks(40, 50);

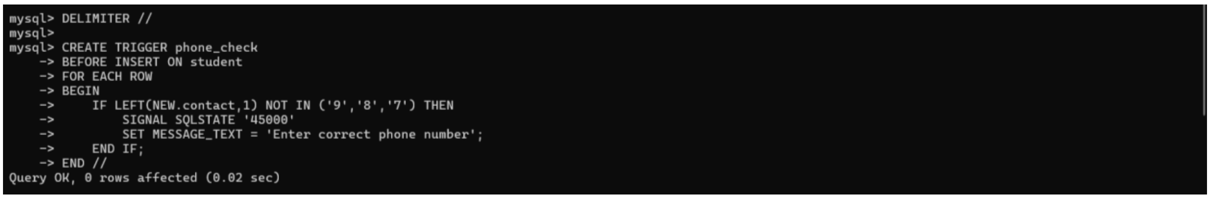


```
mysql> SELECT total_marks(40, 50);
+---------------------+
| total_marks(40, 50) |
+---------------------+
|                  90 |
+---------------------+
1 row in set (0.01 sec)
```

TASK-5

1. <u>Trigger (Phone number validation):</u>

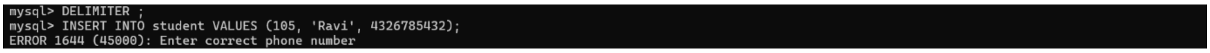
```
DELIMITER //


CREATE TRIGGER phone_check
BEFORE INSERT ON student
FOR EACH ROW
BEGIN
  IF LEFT(NEW.contact,1) NOT IN ('9','8','7') THEN
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Enter correct phone number';
  END IF;
END //


DELIMITER ;
```



```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER phone_check
    -> BEFORE INSERT ON student
    -> FOR EACH ROW
    -> BEGIN
    ->     IF LEFT(NEW.contact,1) NOT IN ('9','8','7') THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Enter correct phone number';
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.02 sec)
```

Test:

INSERT INTO student VALUES (105, 'Ravi', 4326785432);

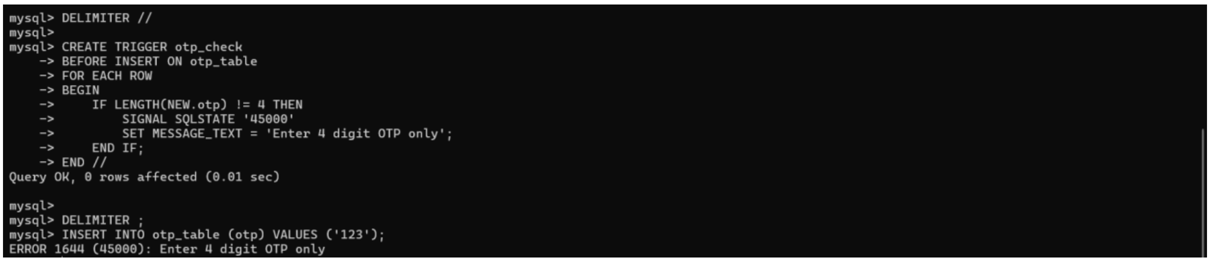


```
mysql> DELIMITER ;
mysql> INSERT INTO student VALUES (105, 'Ravi', 4326785432);
ERROR 1644 (45000): Enter correct phone number
```

2. <u>Trigger (OTP must be exactly 4 digits):</u>

```
DELIMITER //


CREATE TRIGGER otp_check
```

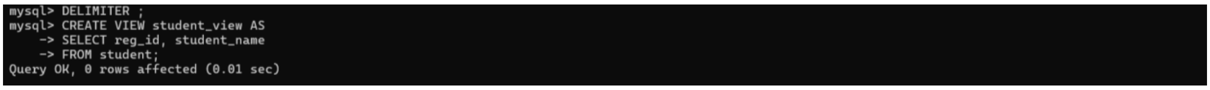
```
BEFORE INSERT ON otp_table

FOR EACH ROW

BEGIN

  IF LENGTH(NEW.otp) != 4 THEN

    SIGNAL SQLSTATE '45000'

    SET MESSAGE_TEXT = 'Enter 4 digit OTP only';

  END IF;

END //


DELIMITER ;
```



```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER otp_check
    -> BEFORE INSERT ON otp_table
    -> FOR EACH ROW
    -> BEGIN
    ->     IF LENGTH(NEW.otp) != 4 THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Enter 4 digit OTP only';
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> INSERT INTO otp_table (otp) VALUES ('123');
ERROR 1644 (45000): Enter 4 digit OTP only
```

3. View (Student basic details):

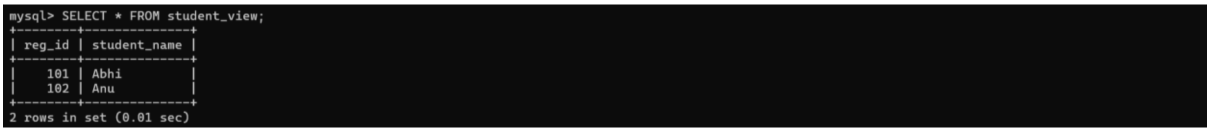
```
CREATE VIEW student_view AS

SELECT reg_id, student_name

FROM student;
```



```
mysql> DELIMITER ;
mysql> CREATE VIEW student_view AS
    -> SELECT reg_id, student_name
    -> FROM student;
Query OK, 0 rows affected (0.01 sec)
```

Use view:

```
SELECT * FROM student_view;
```

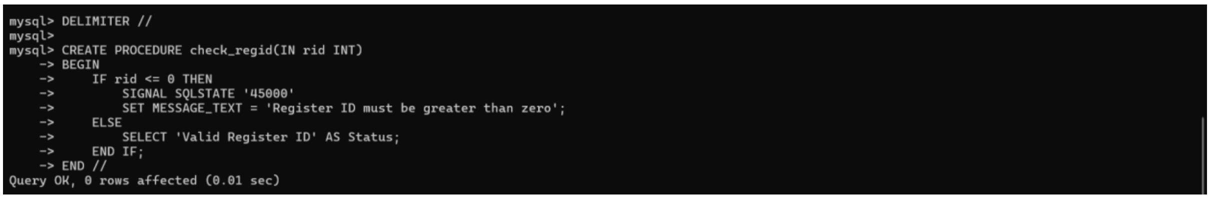


```
mysql> SELECT * FROM student_view;
+--------+--------------+
| reg_id | student_name |
+--------+--------------+
|    101 | Abhi         |
|    102 | Anu          |
+--------+--------------+
2 rows in set (0.01 sec)
```

4. Exception Handling in MySQL (Using SIGNAL):

```
DELIMITER //


CREATE PROCEDURE check_regid(IN rid INT)

BEGIN

  IF rid <= 0 THEN

    SIGNAL SQLSTATE '45000'
```

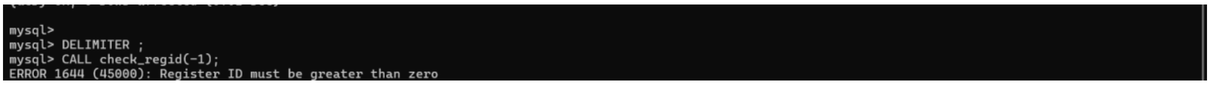
```
        SET MESSAGE_TEXT = 'Register ID must be greater than zero';
    ELSE
        SELECT 'Valid Register ID' AS Status;
    END IF;
END //

DELIMITER ;
```



```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE check_regid(IN rid INT)
    -> BEGIN
    ->     IF rid <= 0 THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Register ID must be greater than zero';
    ->     ELSE
    ->         SELECT 'Valid Register ID' AS Status;
    ->     END IF;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

Call:

CALL check_regid(-1);



```
mysql>
mysql> DELIMITER ;
mysql> CALL check_regid(-1);
ERROR 1644 (45000): Register ID must be greater than zero
```