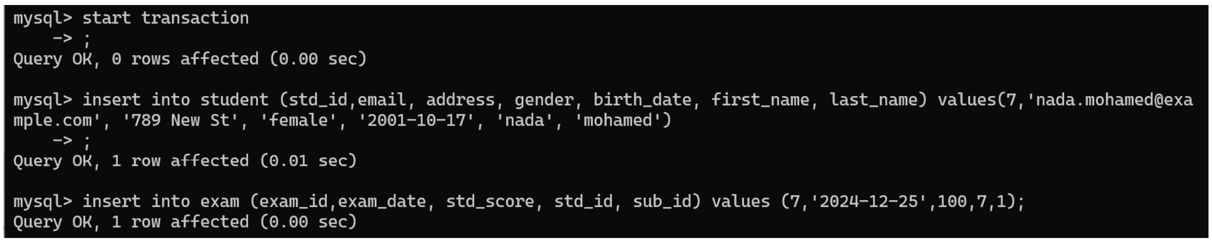
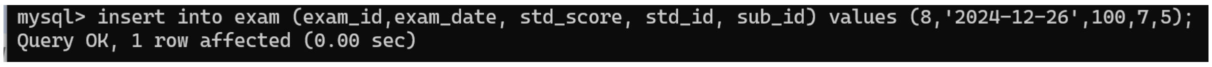
1. Insert new student and his score in exam in different subjects as transaction.

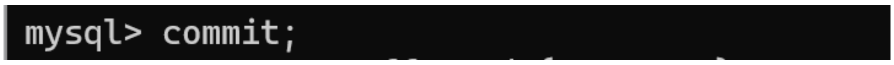
start transaction;
insert into student (std_id,email, address, gender, birth_date, first_name, last_name) values(7,'nada.mohamed@example.com', '789 New St', 'female', '2001-10-17', 'nada', 'mohamed');
 insert into exam (exam_id,exam_date, std_score, std_id, sub_id) values (7,'2024-12-25',100,7,1);
insert into exam (exam_id,exam_date, std_score, std_id, sub_id) values (8,'2024-12-26',100,7,5);
commit;



```
mysql> start transaction
    -> ;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into student (std_id,email, address, gender, birth_date, first_name, last_name) values(7,'nada.mohamed@exa
mple.com', '789 New St', 'female', '2001-10-17', 'nada', 'mohamed')
    -> ;
Query OK, 1 row affected (0.01 sec)

mysql> insert into exam (exam_id,exam_date, std_score, std_id, sub_id) values (7,'2024-12-25',100,7,1);
Query OK, 1 row affected (0.00 sec)
```

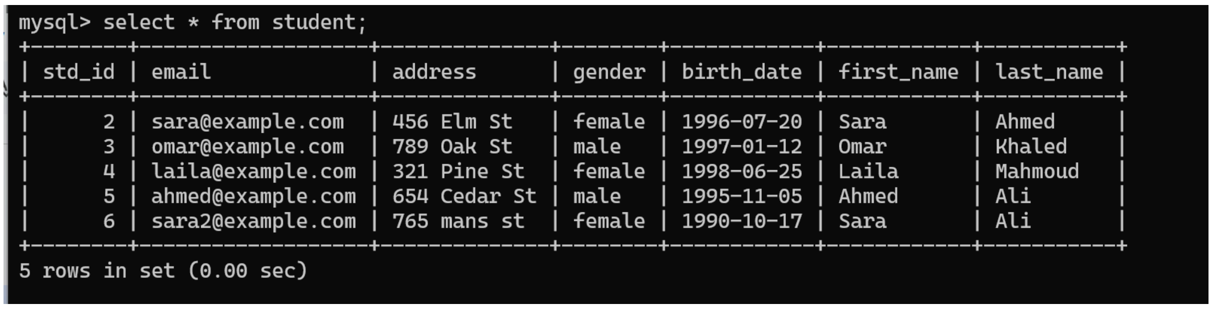


```
mysql> insert into exam (exam_id,exam_date, std_score, std_id, sub_id) values (8,'2024-12-26',100,7,5);
Query OK, 1 row affected (0.00 sec)
```

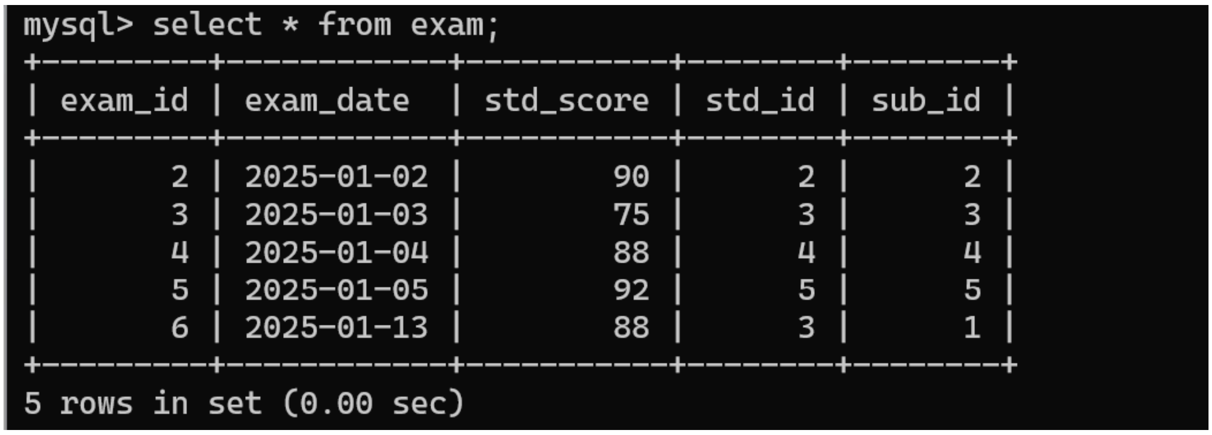


```
mysql> commit;
```

Before commit in other terminal :

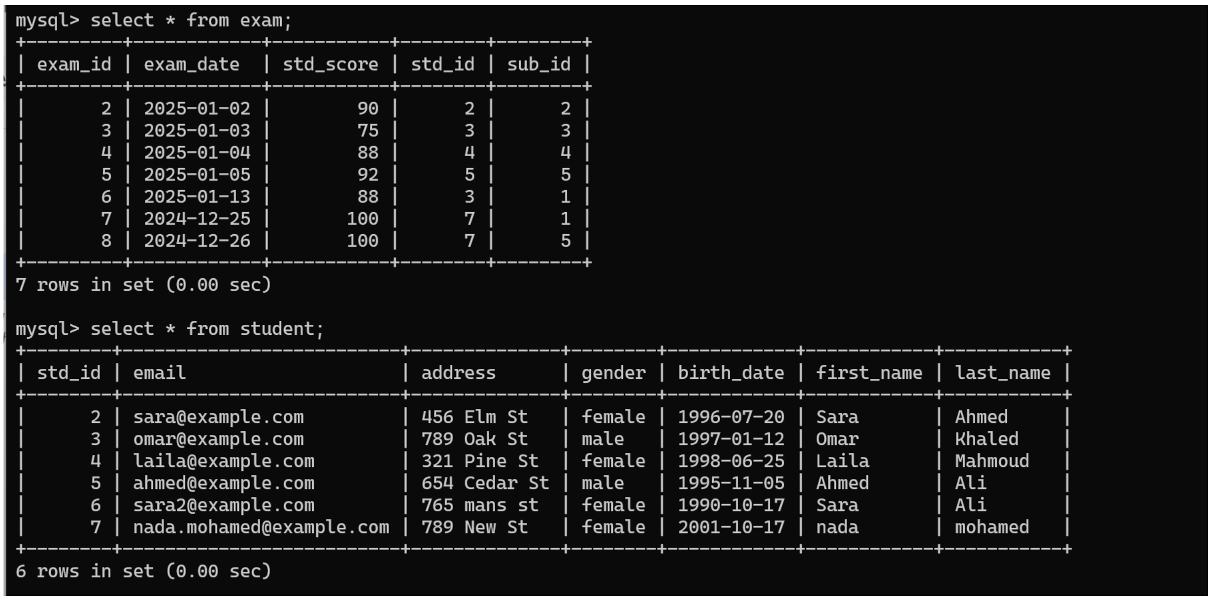


```
mysql> select * from student;
+--------+-------------------+--------------+--------+------------+------------+-----------+
| std_id | email             | address      | gender | birth_date | first_name | last_name |
+--------+-------------------+--------------+--------+------------+------------+-----------+
|      2 | sara@example.com  | 456 Elm St   | female | 1996-07-20 | Sara       | Ahmed     |
|      3 | omar@example.com  | 789 Oak St   | male   | 1997-01-12 | Omar       | Khaled    |
|      4 | laila@example.com | 321 Pine St  | female | 1998-06-25 | Laila      | Mahmoud   |
|      5 | ahmed@example.com | 654 Cedar St | male   | 1995-11-05 | Ahmed      | Ali       |
|      6 | sara2@example.com | 765 mans st  | female | 1990-10-17 | Sara       | Ali       |
+--------+-------------------+--------------+--------+------------+------------+-----------+
5 rows in set (0.00 sec)
```



```
mysql> select * from exam;
+---------+------------+-----------+--------+--------+
| exam_id | exam_date  | std_score | std_id | sub_id |
+---------+------------+-----------+--------+--------+
|       2 | 2025-01-02 |        90 |      2 |      2 |
|       3 | 2025-01-03 |        75 |      3 |      3 |
|       4 | 2025-01-04 |        88 |      4 |      4 |
|       5 | 2025-01-05 |        92 |      5 |      5 |
|       6 | 2025-01-13 |        88 |      3 |      1 |
+---------+------------+-----------+--------+--------+
5 rows in set (0.00 sec)
```

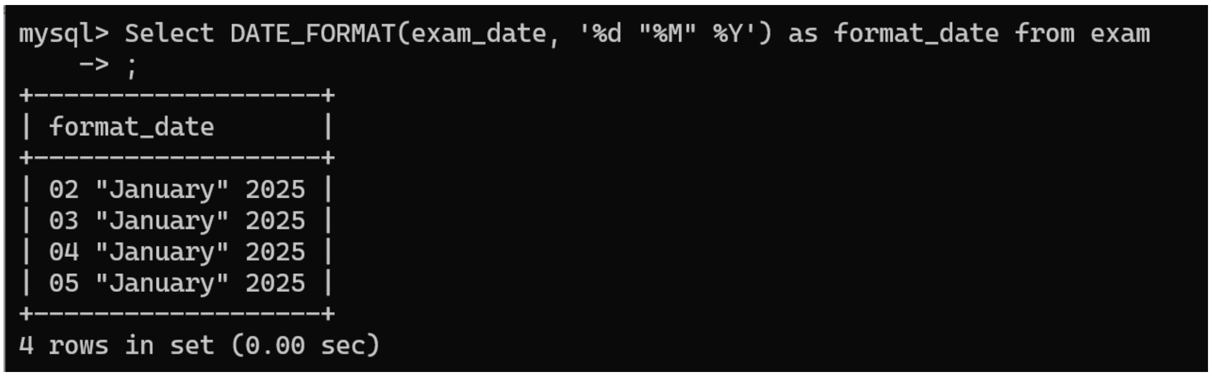
After commit :



```
mysql> select * from exam;
+---------+------------+-----------+--------+--------+
| exam_id | exam_date  | std_score | std_id | sub_id |
+---------+------------+-----------+--------+--------+
|       2 | 2025-01-02 |        90 |      2 |      2 |
|       3 | 2025-01-03 |        75 |      3 |      3 |
|       4 | 2025-01-04 |        88 |      4 |      4 |
|       5 | 2025-01-05 |        92 |      5 |      5 |
|       6 | 2025-01-13 |        88 |      3 |      1 |
|       7 | 2024-12-25 |       100 |      7 |      1 |
|       8 | 2024-12-26 |       100 |      7 |      5 |
+---------+------------+-----------+--------+--------+
7 rows in set (0.00 sec)

mysql> select * from student;
+--------+---------------------------+-------------+--------+------------+------------+-----------+
| std_id | email                     | address     | gender | birth_date | first_name | last_name |
+--------+---------------------------+-------------+--------+------------+------------+-----------+
|      2 | sara@example.com          | 456 Elm St  | female | 1996-07-20 | Sara       | Ahmed     |
|      3 | omar@example.com          | 789 Oak St  | male   | 1997-01-12 | Omar       | Khaled    |
|      4 | laila@example.com         | 321 Pine St | female | 1998-06-25 | Laila      | Mahmoud   |
|      5 | ahmed@example.com         | 654 Cedar St| male   | 1995-11-05 | Ahmed      | Ali       |
|      6 | sara2@example.com         | 765 mans st | female | 1990-10-17 | Sara       | Ali       |
|      7 | nada.mohamed@example.com  | 789 New St  | female | 2001-10-17 | nada       | mohamed   |
+--------+---------------------------+-------------+--------+------------+------------+-----------+
6 rows in set (0.00 sec)
```

2. Display the date of exam as the following: day 'month name' year.

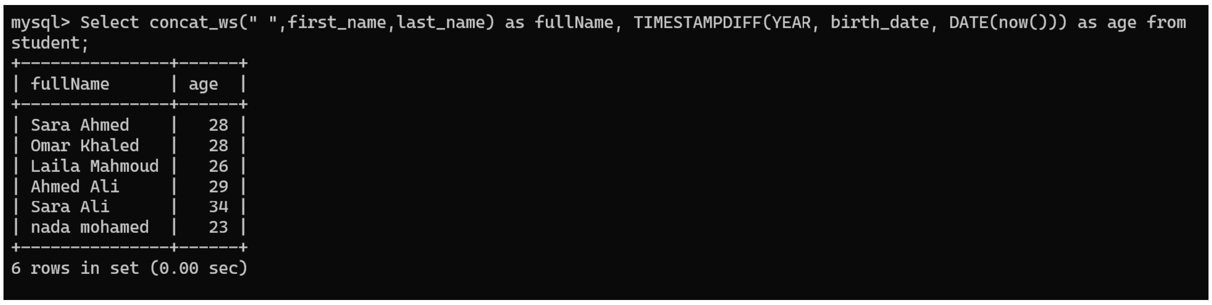
Select DATE_FORMAT(exam_date, '%d "%M" %Y') as format_date from exam



```
mysql> Select DATE_FORMAT(exam_date, '%d "%M" %Y') as format_date from exam
    -> ;
+--------------------+
| format_date        |
+--------------------+
| 02 "January" 2025 |
| 03 "January" 2025 |
| 04 "January" 2025 |
| 05 "January" 2025 |
+--------------------+
4 rows in set (0.00 sec)
```

3. Display name and age of each students

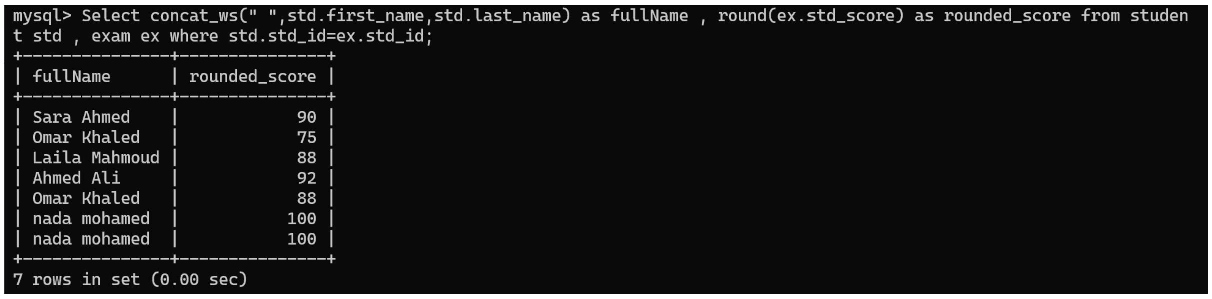
Select concat_ws(“ ”,first_name,last_name) as fullName, TIMESTAMPDIFF(YEAR, birth_date, DATE(NOW())) as age from student;



```
mysql> Select concat_ws(" ",first_name,last_name) as fullName, TIMESTAMPDIFF(YEAR, birth_date, DATE(now())) as age from
student;
+---------------+------+
| fullName      | age  |
+---------------+------+
| Sara Ahmed    |   28 |
| Omar Khaled   |   28 |
| Laila Mahmoud |   26 |
| Ahmed Ali     |   29 |
| Sara Ali      |   34 |
| nada mohamed  |   23 |
+---------------+------+
6 rows in set (0.00 sec)
```

4. Display the name of students with their Rounded score in each Exam

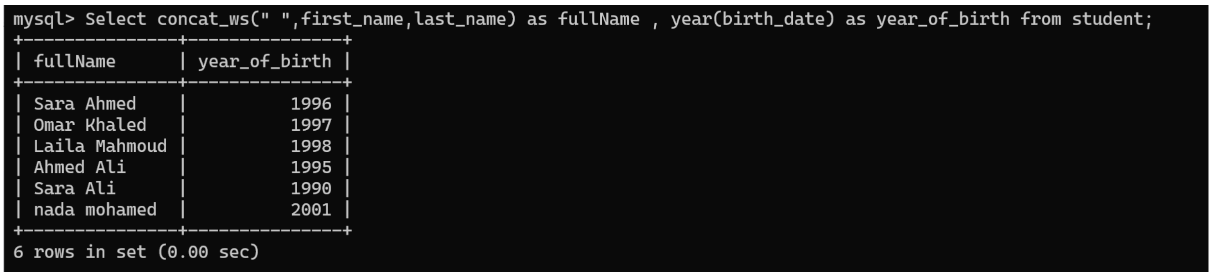
Select concat_ws(‘’ ‘’,std.first_name,std.last_name) as fullName , round(ex.std_score) as rounded_score from student std , exam ex where std.std_id=ex.std_id;



```
mysql> Select concat_ws(" ",std.first_name,std.last_name) as fullName , round(ex.std_score) as rounded_score from studen
t std , exam ex where std.std_id=ex.std_id;
+---------------+---------------+
| fullName      | rounded_score |
+---------------+---------------+
| Sara Ahmed    |            90 |
| Omar Khaled   |            75 |
| Laila Mahmoud |            88 |
| Ahmed Ali     |            92 |
| Omar Khaled   |            88 |
| nada mohamed  |           100 |
| nada mohamed  |           100 |
+---------------+---------------+
7 rows in set (0.00 sec)
```

5. Display the name of students with the year of Birthdate

Select concat_ws(“ ”,first_name,last_name) as fullName , year(birth_date) as year_of_birth from student;

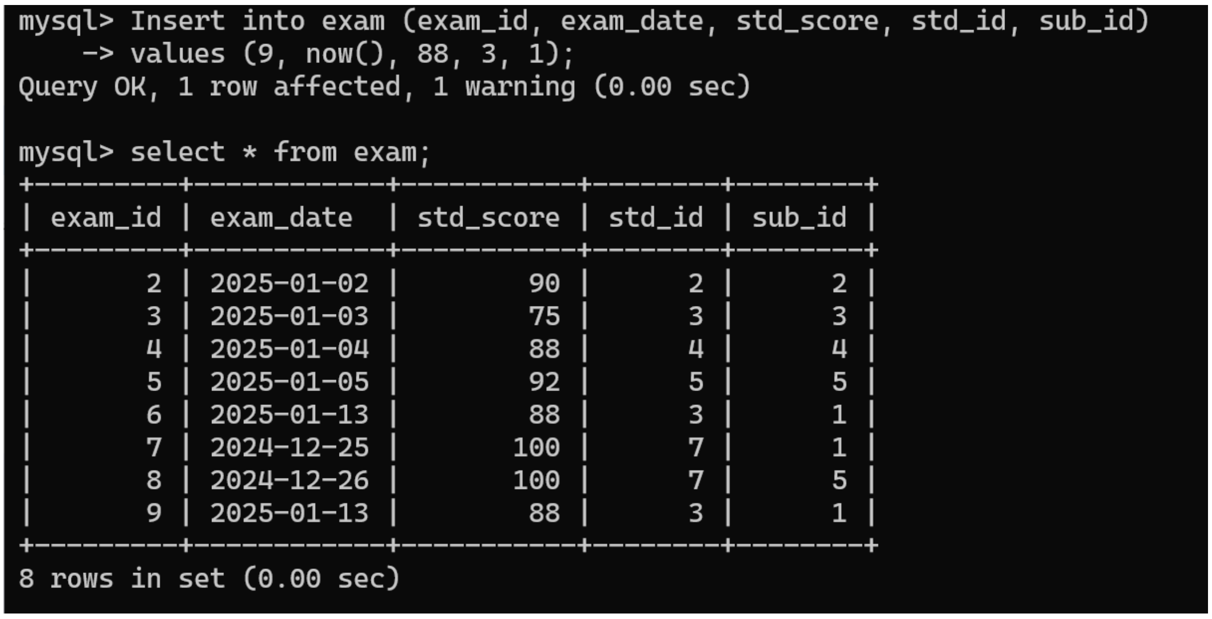


```
mysql> Select concat_ws(" ",first_name,last_name) as fullName , year(birth_date) as year_of_birth from student;
+---------------+---------------+
| fullName      | year_of_birth |
+---------------+---------------+
| Sara Ahmed    |          1996 |
| Omar Khaled   |          1997 |
| Laila Mahmoud |          1998 |
| Ahmed Ali     |          1995 |
| Sara Ali      |          1990 |
| nada mohamed  |          2001 |
+---------------+---------------+
6 rows in set (0.00 sec)
```

6. Add new exam result, in date column use NOW

Insert into exam (exam_id, exam_date, std_score, std_id, sub_id)
values (6, now(), 88, 3, 1);

The col type is date so the result only date 13-1-2024



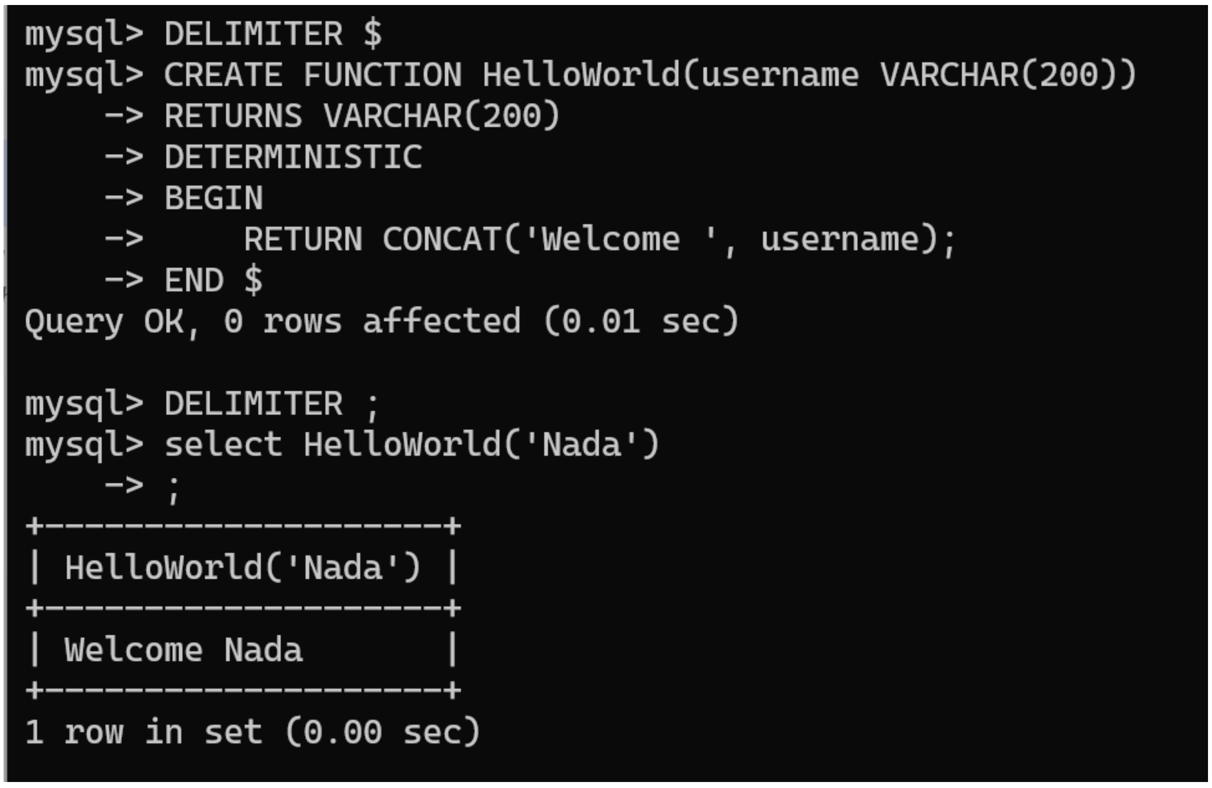
```
mysql> Insert into exam (exam_id, exam_date, std_score, std_id, sub_id)
    -> values (9, now(), 88, 3, 1);
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> select * from exam;
+---------+------------+-----------+--------+--------+
| exam_id | exam_date  | std_score | std_id | sub_id |
+---------+------------+-----------+--------+--------+
|       2 | 2025-01-02 |        90 |      2 |      2 |
|       3 | 2025-01-03 |        75 |      3 |      3 |
|       4 | 2025-01-04 |        88 |      4 |      4 |
|       5 | 2025-01-05 |        92 |      5 |      5 |
|       6 | 2025-01-13 |        88 |      3 |      1 |
|       7 | 2024-12-25 |       100 |      7 |      1 |
|       8 | 2024-12-26 |       100 |      7 |      5 |
|       9 | 2025-01-13 |        88 |      3 |      1 |
+---------+------------+-----------+--------+--------+
8 rows in set (0.00 sec)
```

7. Create Hello world function which take username and return welcome message to user using his name

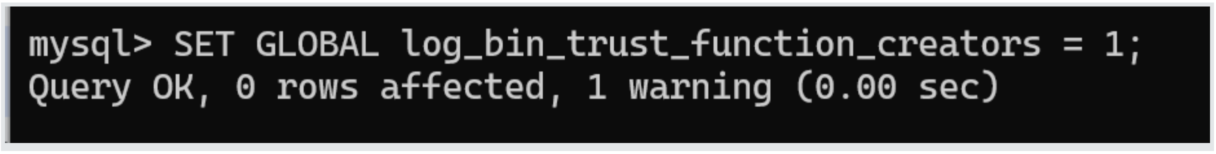
We face a problem say:
This function has none of DETERMINISTIC, NO SQL, or READS SQL DATA in its declaration and binary logging is enabled (you *might* want to use the less safe log_bin_trust_function_creators variable)

So we can write (DETERMINISTIC) in any stored function we create

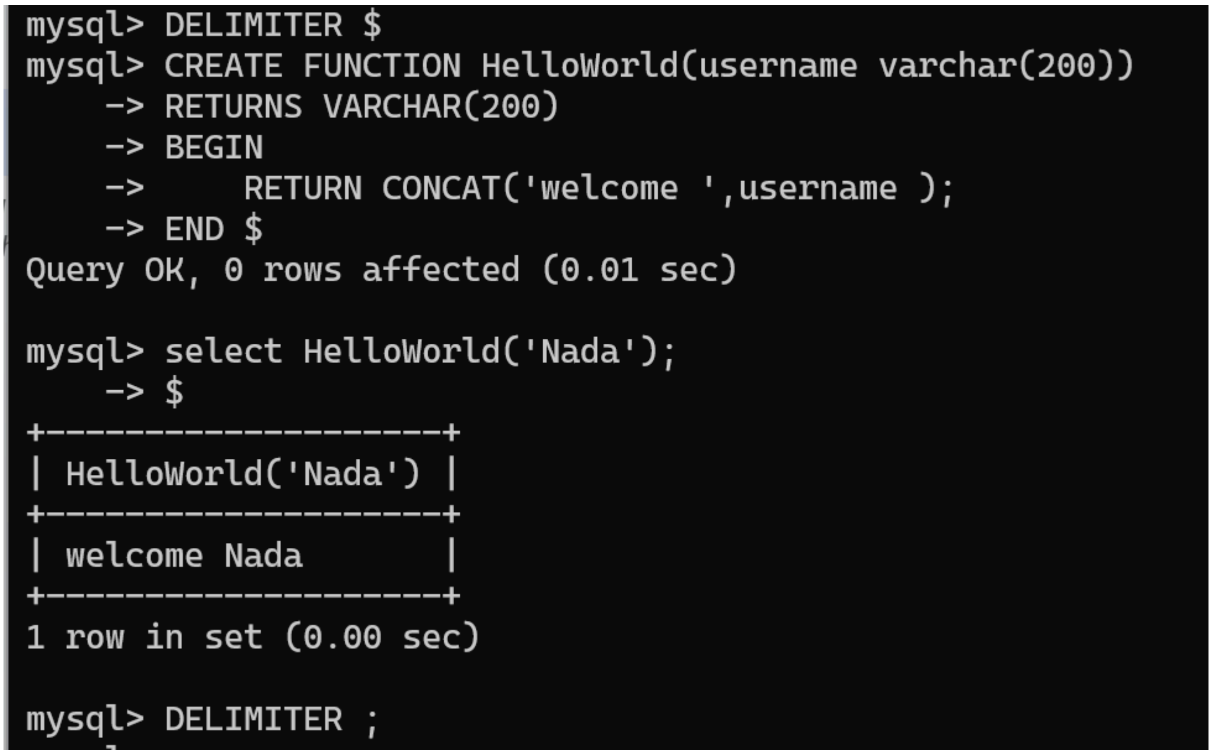
```
DELIMITER $
CREATE FUNCTION HelloWorld(username varchar(200))
RETURNS VARCHAR(200)
DETERMINISTIC
BEGIN
   RETURN CONCAT('welcome ',username );
END $
DELIMITER ;
```


```
mysql> DELIMITER $
mysql> CREATE FUNCTION HelloWorld(username VARCHAR(200))
    -> RETURNS VARCHAR(200)
    -> DETERMINISTIC
    -> BEGIN
    ->     RETURN CONCAT('Welcome ', username);
    -> END $
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> select HelloWorld('Nada')
    -> ;
+--------------------+
| HelloWorld('Nada') |
+--------------------+
| Welcome Nada       |
+--------------------+
1 row in set (0.00 sec)
```

Or
Run this first :
`SET GLOBAL log_bin_trust_function_creators = 1;`


```
mysql> SET GLOBAL log_bin_trust_function_creators = 1;
Query OK, 0 rows affected, 1 warning (0.00 sec)
```

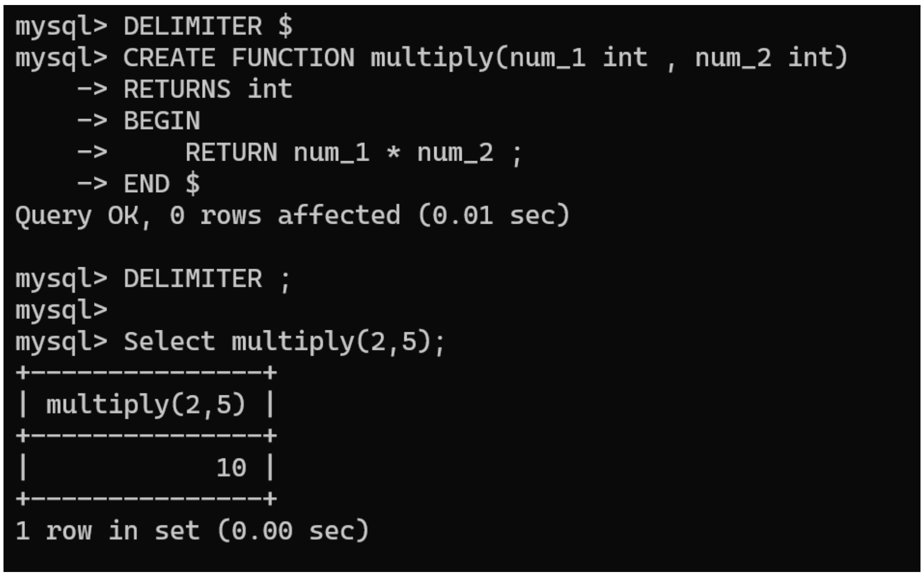
```
DELIMITER $
CREATE FUNCTION HelloWorld(username varchar(200))
RETURNS VARCHAR(200)
BEGIN
   RETURN CONCAT(‘welcome ’,username );
END $
DELIMITER ;

Select HelloWorld(‘Nada’);
```



```
mysql> DELIMITER $
mysql> CREATE FUNCTION HelloWorld(username varchar(200))
    -> RETURNS VARCHAR(200)
    -> BEGIN
    ->     RETURN CONCAT('welcome ',username );
    -> END $
Query OK, 0 rows affected (0.01 sec)

mysql> select HelloWorld('Nada');
    -> $
+--------------------+
| HelloWorld('Nada') |
+--------------------+
| welcome Nada       |
+--------------------+
1 row in set (0.00 sec)

mysql> DELIMITER ;
```

8. Create multiply function which take two number and return the multiply of them.

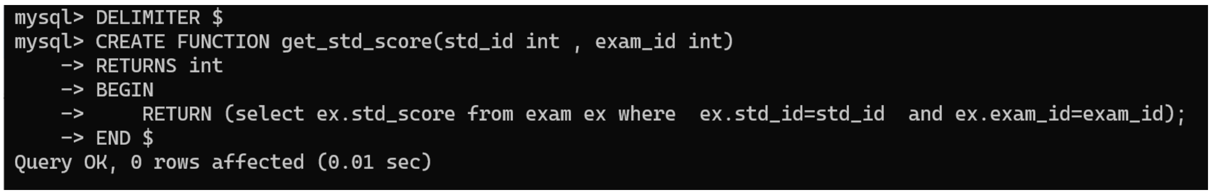
```
DELIMITER $
CREATE FUNCTION multiply(num_1 int , num_2 int)
RETURNS int
BEGIN
   RETURN num_1 * num_2 ;
END $
DELIMITER ;

Select multiply(2,5);
```



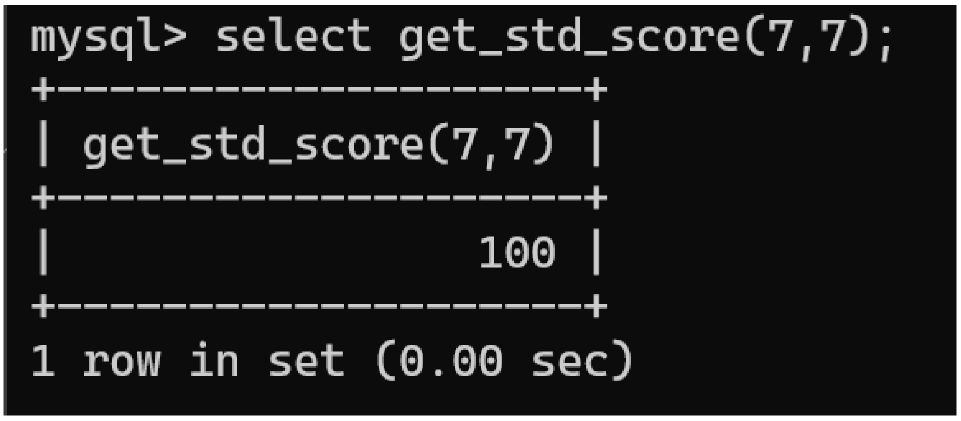
```
mysql> DELIMITER $
mysql> CREATE FUNCTION multiply(num_1 int , num_2 int)
    -> RETURNS int
    -> BEGIN
    ->     RETURN num_1 * num_2 ;
    -> END $
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql>
mysql> Select multiply(2,5);
+---------------+
| multiply(2,5) |
+---------------+
|            10 |
+---------------+
1 row in set (0.00 sec)
```

9. Create function which takes student id and Exam id and return score the student in Exam.

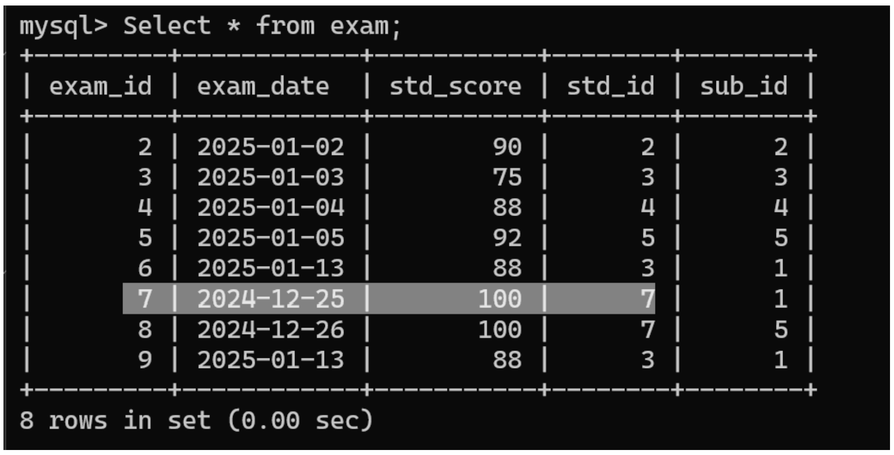
```
DELIMITER $
CREATE FUNCTION get_std_score(std_id int , exam_id int)
RETURNS int
BEGIN
   RETURN (select ex.std_score from exam ex where  ex.std_id=std_id  and
ex.exam_id=exam_id);
END $
DELIMITER ;

select get_std_score(7,7);
```



```
mysql> DELIMITER $
mysql> CREATE FUNCTION get_std_score(std_id int , exam_id int)
    -> RETURNS int
    -> BEGIN
    ->     RETURN (select ex.std_score from exam ex where  ex.std_id=std_id  and ex.exam_id=exam_id);
    -> END $
Query OK, 0 rows affected (0.01 sec)
```



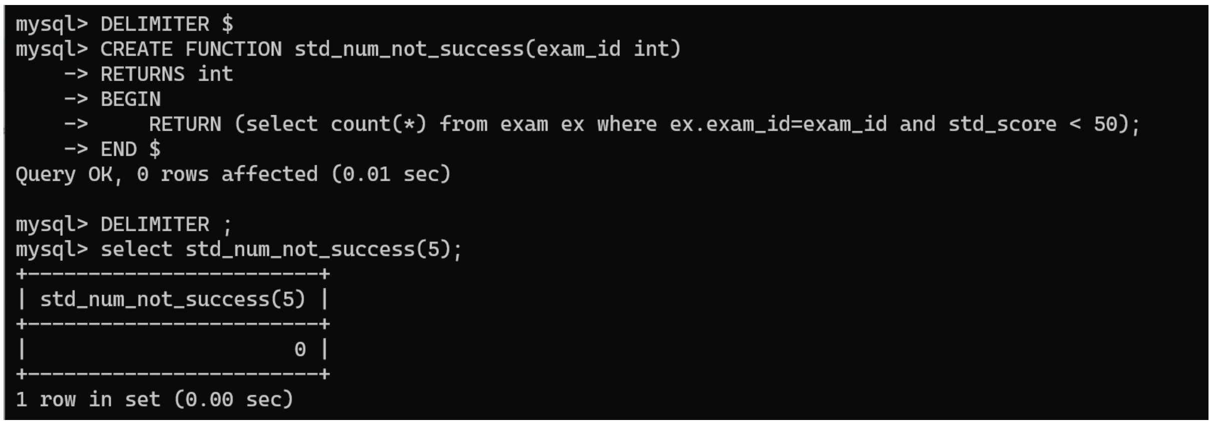
```
mysql> select get_std_score(7,7);
+--------------------+
| get_std_score(7,7) |
+--------------------+
|                100 |
+--------------------+
1 row in set (0.00 sec)
```



```
mysql> Select * from exam;
+---------+------------+-----------+--------+--------+
| exam_id | exam_date  | std_score | std_id | sub_id |
+---------+------------+-----------+--------+--------+
|       2 | 2025-01-02 |        90 |      2 |      2 |
|       3 | 2025-01-03 |        75 |      3 |      3 |
|       4 | 2025-01-04 |        88 |      4 |      4 |
|       5 | 2025-01-05 |        92 |      5 |      5 |
|       6 | 2025-01-13 |        88 |      3 |      1 |
|       7 | 2024-12-25 |       100 |      7 |      1 |
|       8 | 2024-12-26 |       100 |      7 |      5 |
|       9 | 2025-01-13 |        88 |      3 |      1 |
+---------+------------+-----------+--------+--------+
8 rows in set (0.00 sec)
```

10. Create a function which takes Exam id and return the number of students who failed in an Exam (Score less than 50).

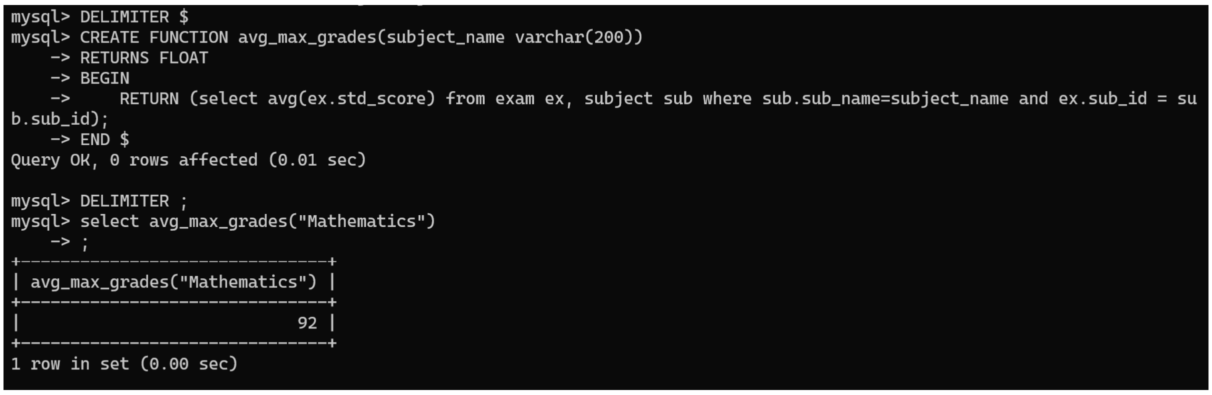
```
DELIMITER $
CREATE FUNCTION std_num_not_success(exam_id int)
RETURNS int
BEGIN
   RETURN (select count(*) from exam ex where ex.exam_id=exam_id and
std_score < 50);
END $
DELIMITER ;

select std_num_not_success(5);
```



```
mysql> DELIMITER $
mysql> CREATE FUNCTION std_num_not_success(exam_id int)
    -> RETURNS int
    -> BEGIN
    ->     RETURN (select count(*) from exam ex where ex.exam_id=exam_id and std_score < 50);
    -> END $
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> select std_num_not_success(5);
+------------------------+
| std_num_not_success(5) |
+------------------------+
|                      0 |
+------------------------+
1 row in set (0.00 sec)
```

11. Create a function which takes the subject name and returns the average of grades for the subject.

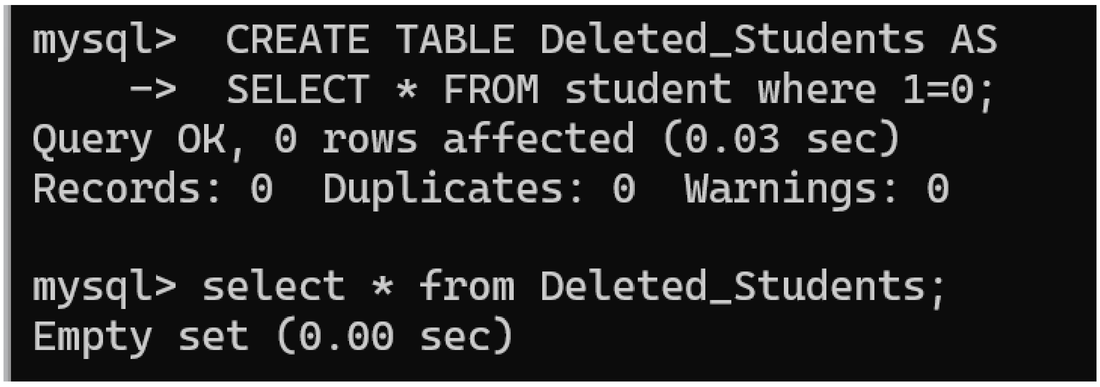
```
DELIMITER $
CREATE FUNCTION avg_max_grades(subject_name varchar(200))
RETURNS FLOAT
BEGIN
   RETURN (select avg(ex.std_score) from exam ex, subject sub where
sub.sub_name=subject_name and ex.sub_id = sub.sub_id);
END $
DELIMITER ;
```



```
mysql> DELIMITER $
mysql> CREATE FUNCTION avg_max_grades(subject_name varchar(200))
    -> RETURNS FLOAT
    -> BEGIN
    ->     RETURN (select avg(ex.std_score) from exam ex, subject sub where sub.sub_name=subject_name and ex.sub_id = su
b.sub_id);
    -> END $
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> select avg_max_grades("Mathematics")
    -> ;
+-------------------------------+
| avg_max_grades("Mathematics") |
+-------------------------------+
|                            92 |
+-------------------------------+
1 row in set (0.00 sec)
```

12. Create Table called Deleted_Students which will hold the deleted students info(same columns as in student tables)

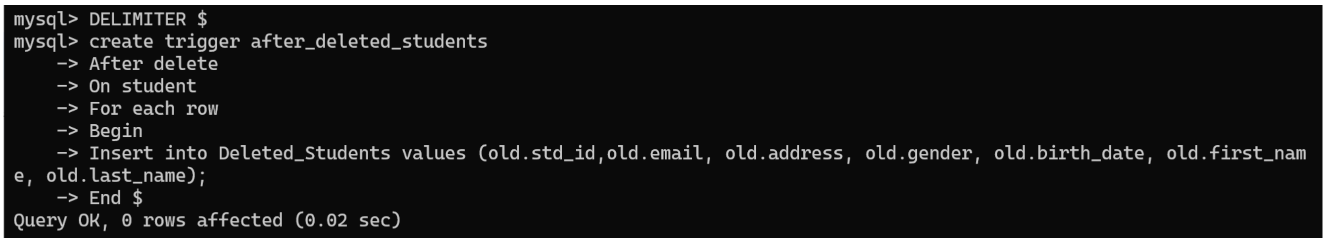
CREATE TABLE Deleted_Students AS SELECT * FROM student where 1=0;
(create with the same structure but without same date)
Or
CREATE TABLE Deleted_Students LIKE student;



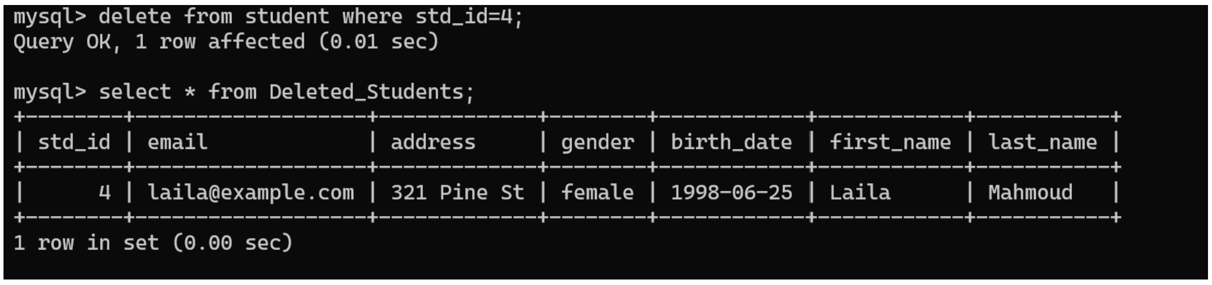
```
mysql>  CREATE TABLE Deleted_Students AS
    ->  SELECT * FROM student where 1=0;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from Deleted_Students;
Empty set (0.00 sec)
```

13. Create trigger to save the deleted student from Student table to Deleted_Students.

```
DELIMITER $
create trigger after_deleted_students
After delete
On student
For each row
Begin
        Insert into Deleted_Students values (old.std_id,old.email, old.address,
        old.gender, old.birth_date, old.first_name, old.last_name);
End $
DELIMITER ;
```



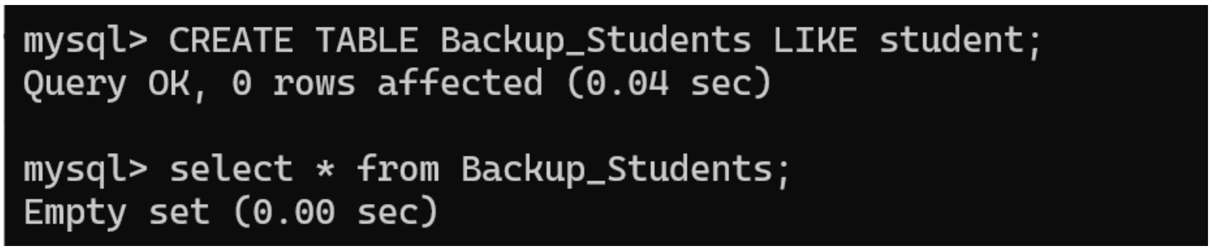
```
mysql> DELIMITER $
mysql> create trigger after_deleted_students
    -> After delete
    -> On student
    -> For each row
    -> Begin
    -> Insert into Deleted_Students values (old.std_id,old.email, old.address, old.gender, old.birth_date, old.first_nam
e, old.last_name);
    -> End $
Query OK, 0 rows affected (0.02 sec)
```



```
mysql> delete from student where std_id=4;
Query OK, 1 row affected (0.01 sec)

mysql> select * from Deleted_Students;
+--------+-------------------+-------------+--------+------------+------------+-----------+
| std_id | email             | address     | gender | birth_date | first_name | last_name |
+--------+-------------------+-------------+--------+------------+------------+-----------+
|      4 | laila@example.com | 321 Pine St | female | 1998-06-25 | Laila      | Mahmoud   |
+--------+-------------------+-------------+--------+------------+------------+-----------+
1 row in set (0.00 sec)
```

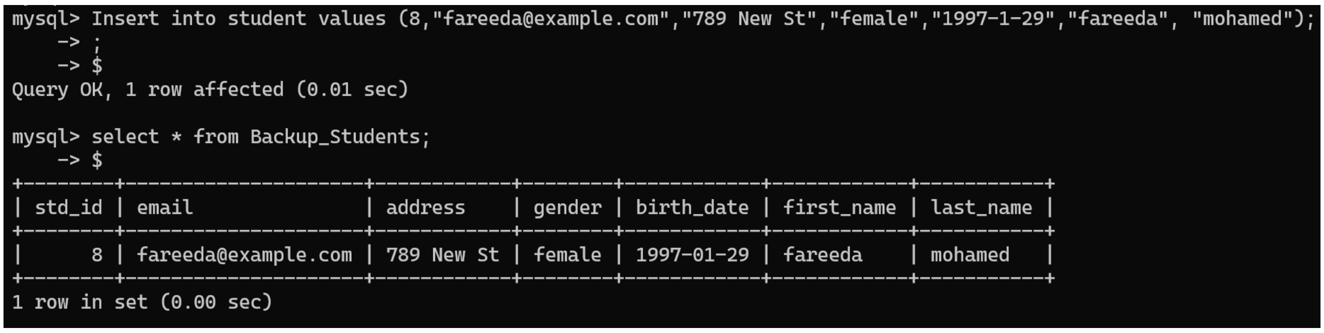
14. Create a trigger to save the newly added students to Student table to Backup_Students.

Create table Backup_Students like student



```
mysql> CREATE TABLE Backup_Students LIKE student;
Query OK, 0 rows affected (0.04 sec)

mysql> select * from Backup_Students;
Empty set (0.00 sec)
```

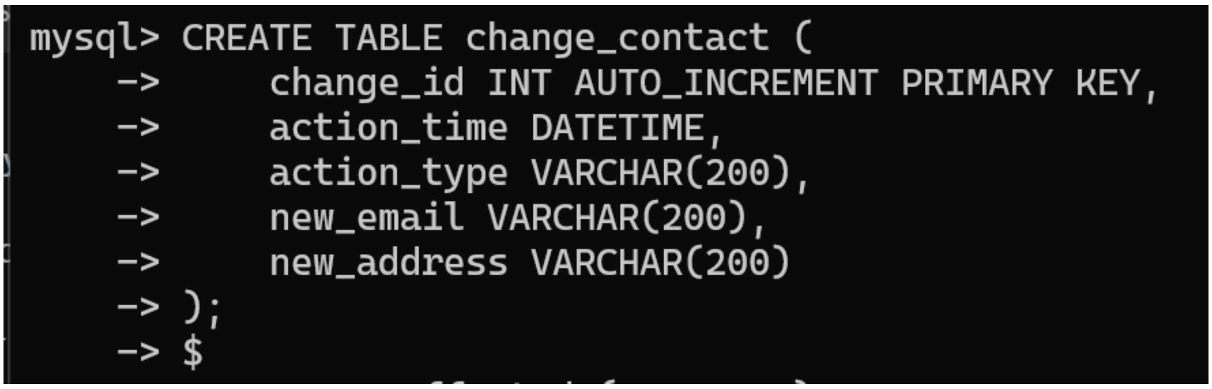
```
DELIMITER $
create trigger after_added_students
After insert
On student
For each row
Begin
        Insert into Backup_Students values (new.std_id,new.email, new.address,
        new.gender, new.birth_date, new.first_name, new.last_name);
End $
```



```
mysql> Insert into student values (8,"fareeda@example.com","789 New St","female","1997-1-29","fareeda", "mohamed");
    -> ;
    -> $
Query OK, 1 row affected (0.01 sec)

mysql> select * from Backup_Students;
    -> $
+--------+---------------------+------------+--------+------------+------------+-----------+
| std_id | email               | address    | gender | birth_date | first_name | last_name |
+--------+---------------------+------------+--------+------------+------------+-----------+
|      8 | fareeda@example.com | 789 New St | female | 1997-01-29 | fareeda    | mohamed   |
+--------+---------------------+------------+--------+------------+------------+-----------+
1 row in set (0.00 sec)
```

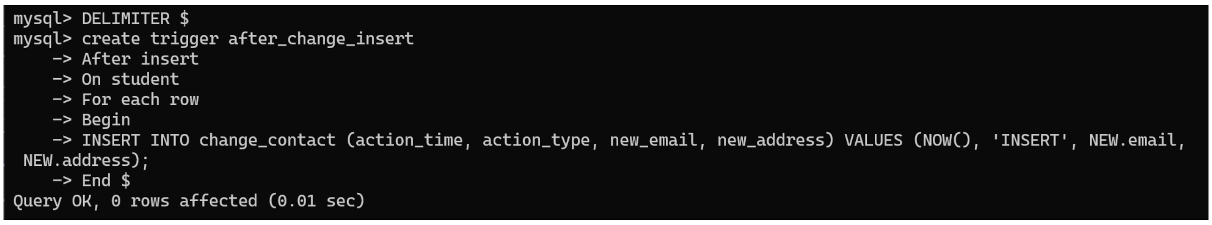
15. Create a trigger to keep track of the changes of the contact info table (add/update rows); it will log the time of action and description of action to another table.
If we mean the contact info which exist in student table and two triggers:

CREATE TABLE change_contact (change_id INT AUTO_INCREMENT PRIMARY KEY, action_time DATETIME, action_type VARCHAR(200), new_email VARCHAR(200), new_address VARCHAR(200));



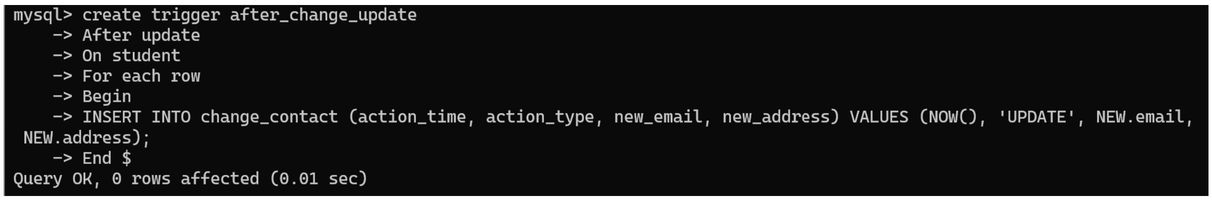
```
mysql> CREATE TABLE change_contact (
    ->     change_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     action_time DATETIME,
    ->     action_type VARCHAR(200),
    ->     new_email VARCHAR(200),
    ->     new_address VARCHAR(200)
    -> );
    -> $
```

DELIMITER $
create trigger after_change_insert
After insert
On student
For each row
Begin
    INSERT INTO change_contact (action_time, action_type, new_email, new_address) VALUES (NOW(), 'INSERT', NEW.email, NEW.address);
End $

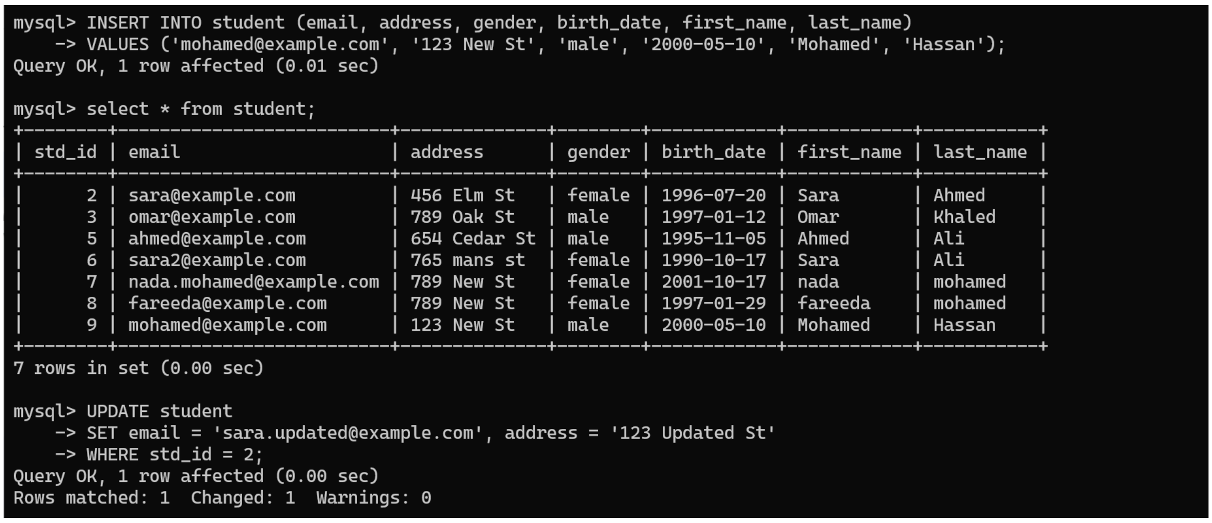


```
mysql> DELIMITER $
mysql> create trigger after_change_insert
    -> After insert
    -> On student
    -> For each row
    -> Begin
    -> INSERT INTO change_contact (action_time, action_type, new_email, new_address) VALUES (NOW(), 'INSERT', NEW.email,
 NEW.address);
    -> End $
Query OK, 0 rows affected (0.01 sec)
```

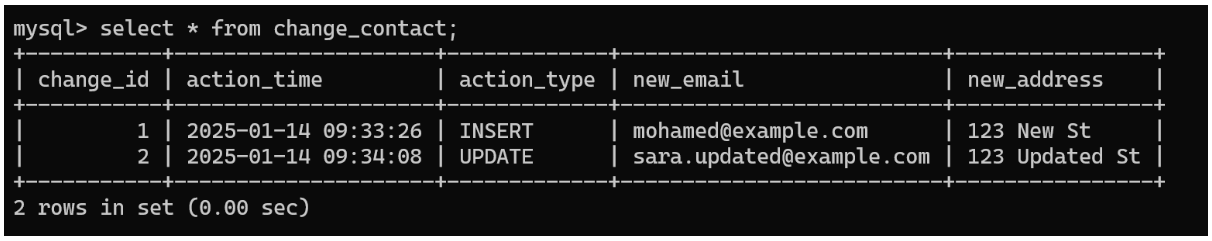
create trigger after_change_update
After update
On student
For each row
Begin
    INSERT INTO change_contact (action_time, action_type, new_email, new_address) VALUES (NOW(), 'UPDATE', NEW.email, NEW.address);
End $
DELIMITER ;



```
mysql> create trigger after_change_update
    -> After update
    -> On student
    -> For each row
    -> Begin
    -> INSERT INTO change_contact (action_time, action_type, new_email, new_address) VALUES (NOW(), 'UPDATE', NEW.email,
 NEW.address);
    -> End $
Query OK, 0 rows affected (0.01 sec)
```



```
mysql> INSERT INTO student (email, address, gender, birth_date, first_name, last_name)
    -> VALUES ('mohamed@example.com', '123 New St', 'male', '2000-05-10', 'Mohamed', 'Hassan');
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+--------+-------------------------+--------------+--------+------------+------------+-----------+
| std_id | email                   | address      | gender | birth_date | first_name | last_name |
+--------+-------------------------+--------------+--------+------------+------------+-----------+
|      2 | sara@example.com        | 456 Elm St   | female | 1996-07-20 | Sara       | Ahmed     |
|      3 | omar@example.com        | 789 Oak St   | male   | 1997-01-12 | Omar       | Khaled    |
|      5 | ahmed@example.com       | 654 Cedar St | male   | 1995-11-05 | Ahmed      | Ali       |
|      6 | sara2@example.com       | 765 mans st  | female | 1990-10-17 | Sara       | Ali       |
|      7 | nada.mohamed@example.com | 789 New St  | female | 2001-10-17 | nada       | mohamed   |
|      8 | fareeda@example.com     | 789 New St   | female | 1997-01-29 | fareeda    | mohamed   |
|      9 | mohamed@example.com     | 123 New St   | male   | 2000-05-10 | Mohamed    | Hassan    |
+--------+-------------------------+--------------+--------+------------+------------+-----------+
7 rows in set (0.00 sec)

mysql> UPDATE student
    -> SET email = 'sara.updated@example.com', address = '123 Updated St'
    -> WHERE std_id = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```



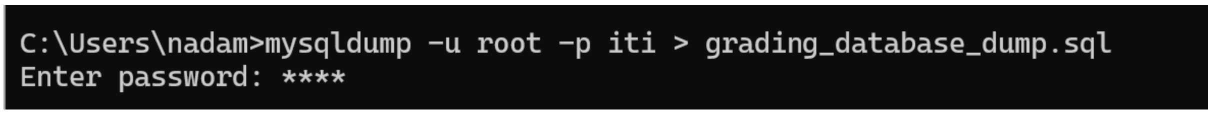
```
mysql> select * from change_contact;
+-----------+---------------------+-------------+--------------------------+----------------+
| change_id | action_time         | action_type | new_email                | new_address    |
+-----------+---------------------+-------------+--------------------------+----------------+
|         1 | 2025-01-14 09:33:26 | INSERT      | mohamed@example.com      | 123 New St     |
|         2 | 2025-01-14 09:34:08 | UPDATE      | sara.updated@example.com | 123 Updated St |
+-----------+---------------------+-------------+--------------------------+----------------+
2 rows in set (0.00 sec)
```

One trigger:

```
DELIMITER $
create trigger after_change_insert_update
After insert or update
On student
For each row
Begin
        Declare action_type varchar(200);
        IF (OLD.std_id IS NULL) THEN
              SET action_type = 'INSERT';
        ELSE
             SET action_type = 'UPDATE';
         END;

        INSERT INTO change_contact (action_time, action_type, new_email,
        new_address) VALUES (NOW(),action_type , NEW.email, NEW.address);
End $
DELIMITER ;
```

16. Dump your database (Grading Database) into SQL file.

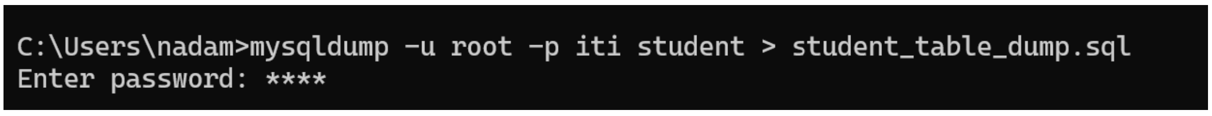
mysqldump -u root -p iti > grading_database_dump.sql



```
C:\Users\nadam>mysqldump -u root -p iti > grading_database_dump.sql
Enter password: ****
```

17. Dump Students table into file.

mysqldump -u root -p iti student > student_table_dump.sql

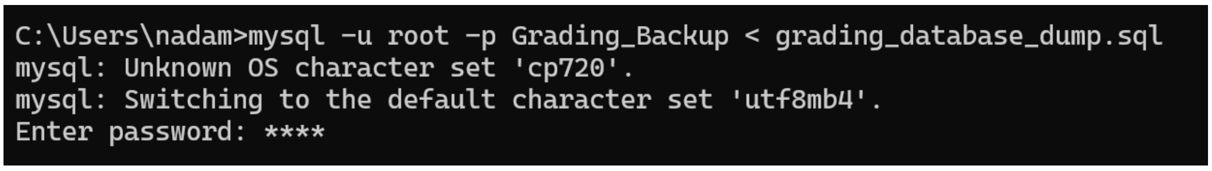


```
C:\Users\nadam>mysqldump -u root -p iti student > student_table_dump.sql
Enter password: ****
```

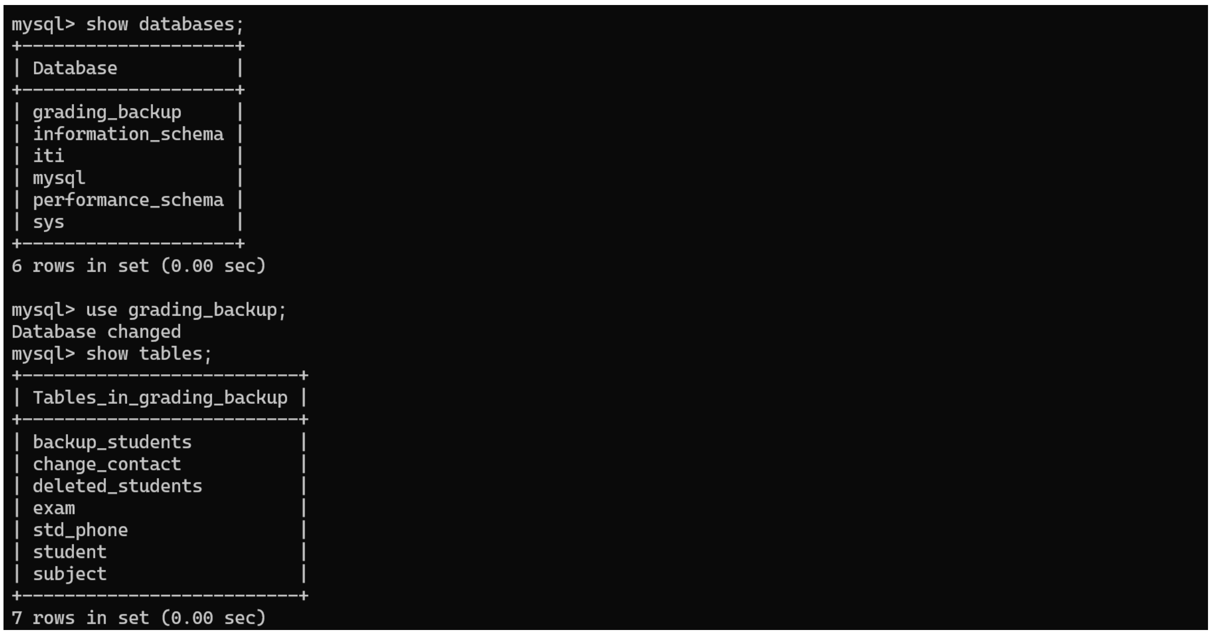
18. Import SQL file into your backup database (Grading_Backup Database)

CREATE DATABASE Grading_Backup;

mysql -u root -p Grading_Backup < grading_database_dump.sql;



```
C:\Users\nadam>mysql -u root -p Grading_Backup < grading_database_dump.sql
mysql: Unknown OS character set 'cp720'.
mysql: Switching to the default character set 'utf8mb4'.
Enter password: ****
```



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| grading_backup     |
| information_schema |
| iti                |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.00 sec)

mysql> use grading_backup;
Database changed
mysql> show tables;
+--------------------------+
| Tables_in_grading_backup |
+--------------------------+
| backup_students          |
| change_contact           |
| deleted_students         |
| exam                     |
| std_phone                |
| student                  |
| subject                  |
+--------------------------+
7 rows in set (0.00 sec)
```