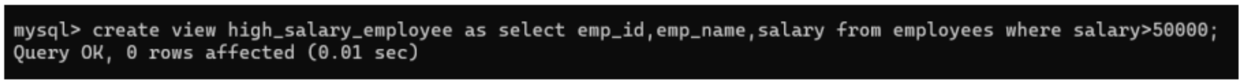
Assignment - 14/10/2024

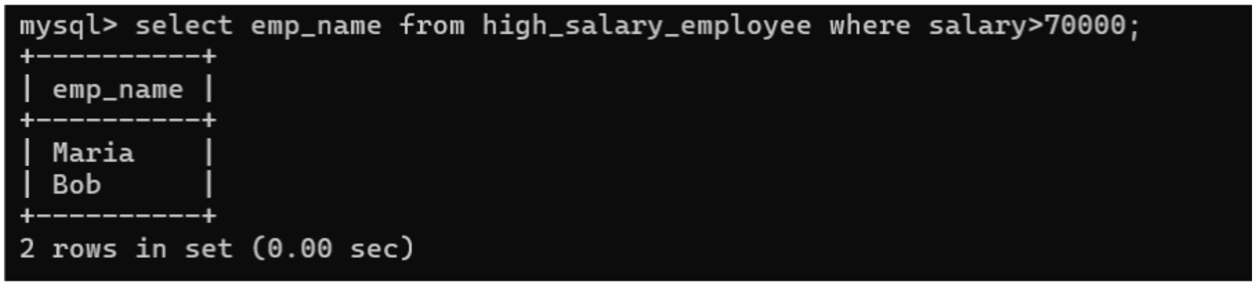
(Create employee and department table as per the requirements)

1. Create a view named high_salary_employees that lists the emp_id, emp_name, and emp_salary of all employees who have a salary greater than 50,000 from the employees table.



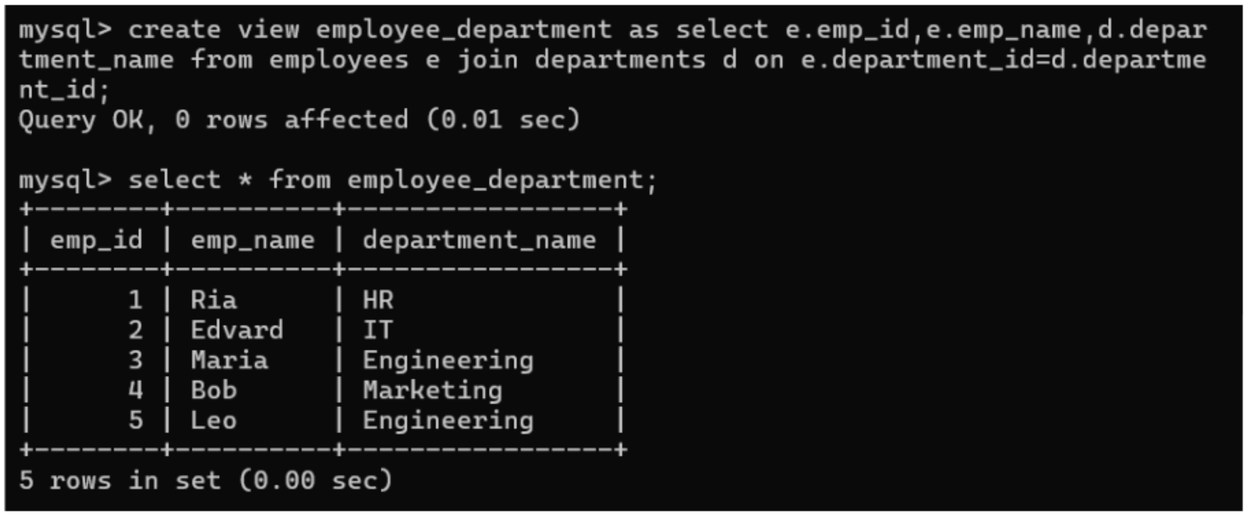
```
mysql> create view high_salary_employee as select emp_id,emp_name,salary from employees where salary>50000;
Query OK, 0 rows affected (0.01 sec)
```

2. Using the view high_salary_employees, write a query to display the names of all employees earning more than 70,000.

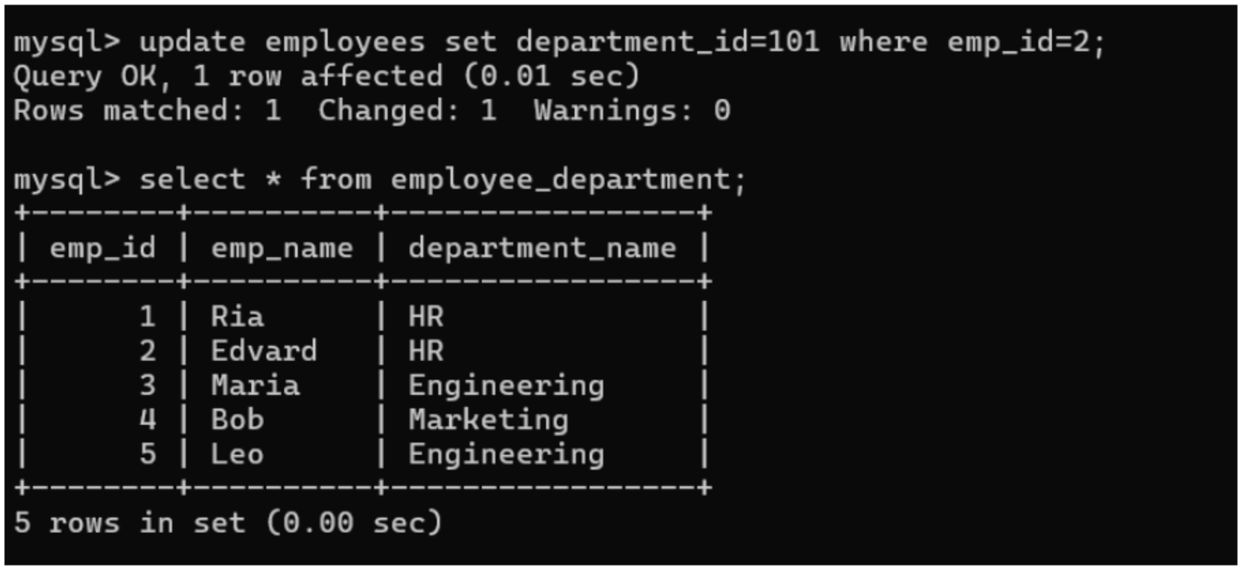


```
mysql> select emp_name from high_salary_employee where salary>70000;
+----------+
| emp_name |
+----------+
| Maria    |
| Bob      |
+----------+
2 rows in set (0.00 sec)
```

3. Create a view called employee_department that joins the employees and departments tables and displays the emp_id, emp_name, and dept_name. Write an UPDATE query to change the dept_name for an employee in the view. Check if the underlying table(s) get updated.

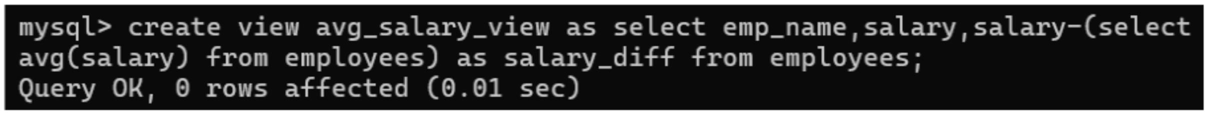


```
mysql> create view employee_department as select e.emp_id,e.emp_name,d.depar
tment_name from employees e join departments d on e.department_id=d.departme
nt_id;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from employee_department;
+--------+----------+-----------------+
| emp_id | emp_name | department_name |
+--------+----------+-----------------+
|      1 | Ria      | HR              |
|      2 | Edvard   | IT              |
|      3 | Maria    | Engineering     |
|      4 | Bob      | Marketing       |
|      5 | Leo      | Engineering     |
+--------+----------+-----------------+
5 rows in set (0.00 sec)
```

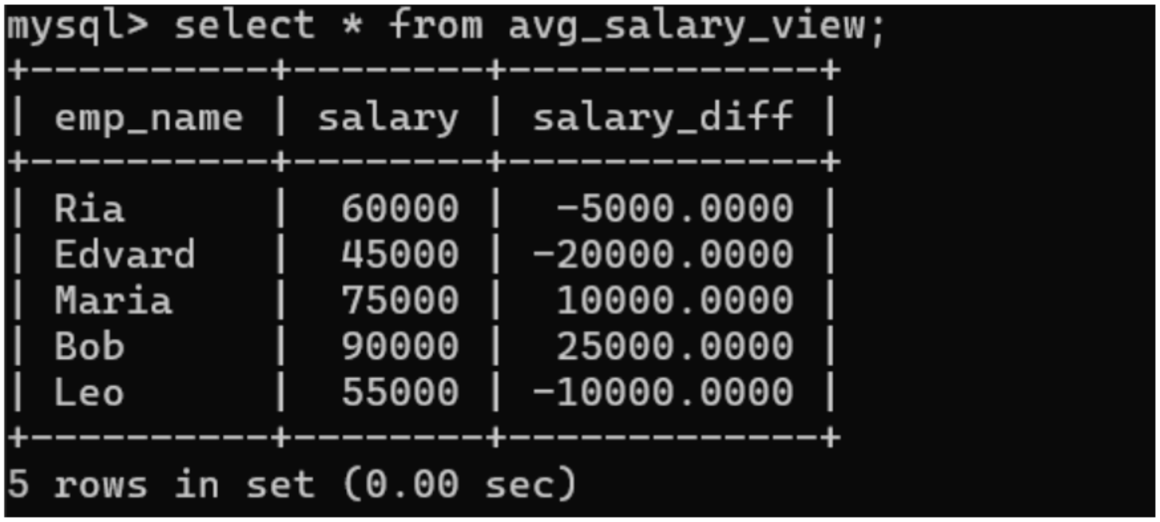


```
mysql> update employees set department_id=101 where emp_id=2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from employee_department;
+--------+----------+-----------------+
| emp_id | emp_name | department_name |
+--------+----------+-----------------+
|      1 | Ria      | HR              |
|      2 | Edvard   | HR              |
|      3 | Maria    | Engineering     |
|      4 | Bob      | Marketing       |
|      5 | Leo      | Engineering     |
+--------+----------+-----------------+
5 rows in set (0.00 sec)
```

4. Create a view called avg_salary_view that shows the emp_name and the difference between their salary and the average salary in the employees table.

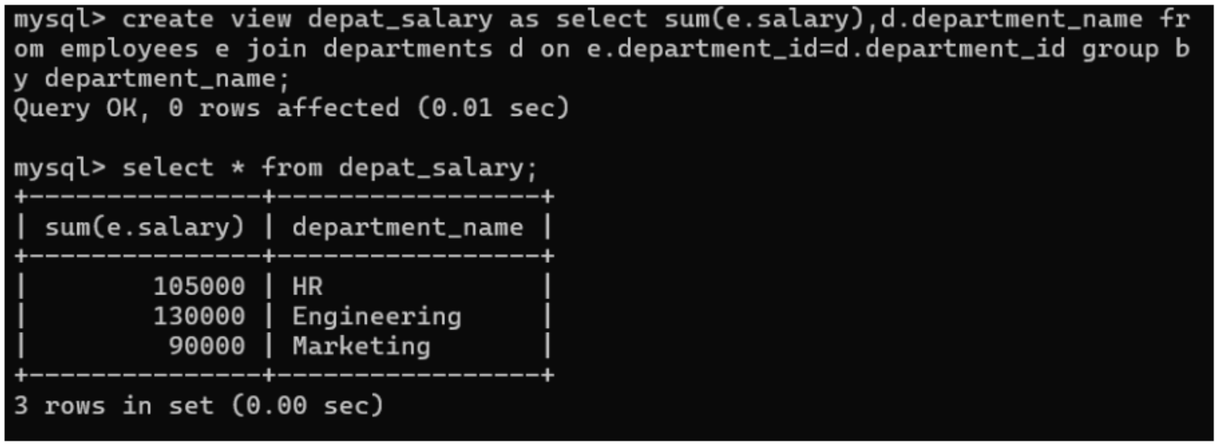


```
mysql> create view avg_salary_view as select emp_name,salary,salary-(select
avg(salary) from employees) as salary_diff from employees;
Query OK, 0 rows affected (0.01 sec)
```



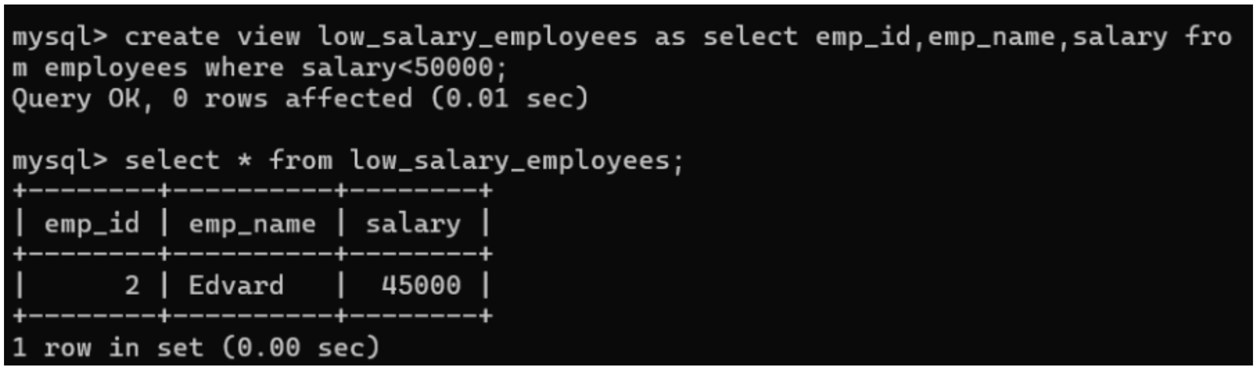
```
mysql> select * from avg_salary_view;
+----------+--------+-------------+
| emp_name | salary | salary_diff |
+----------+--------+-------------+
| Ria      |  60000 |  -5000.0000 |
| Edvard   |  45000 | -20000.0000 |
| Maria    |  75000 |  10000.0000 |
| Bob      |  90000 |  25000.0000 |
| Leo      |  55000 | -10000.0000 |
+----------+--------+-------------+
5 rows in set (0.00 sec)
```

5. Create a view called dept_salary_total that lists each department's name and the total salary for all employees in that department, using a GROUP BY clause.



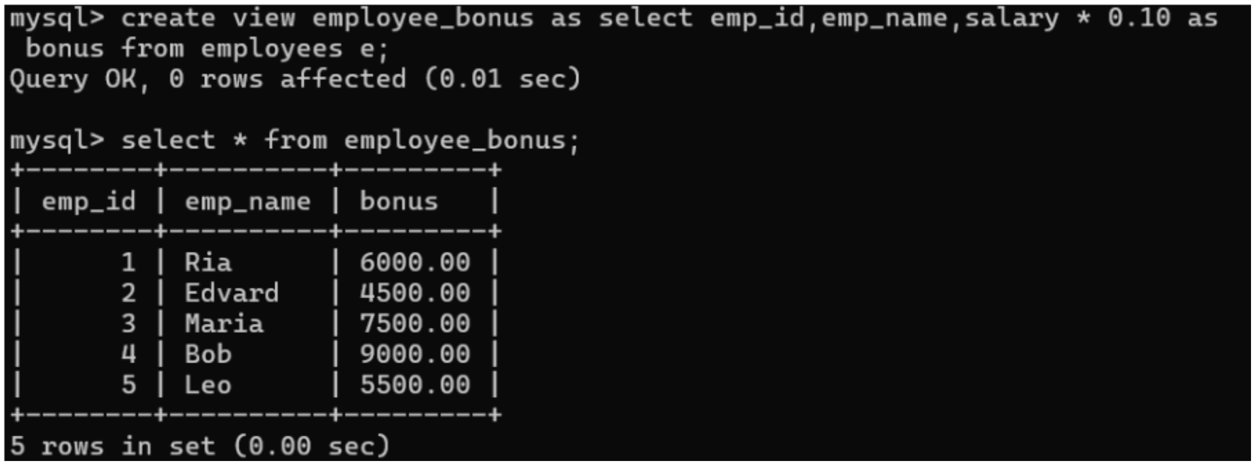
```
mysql> create view depat_salary as select sum(e.salary),d.department_name fr
om employees e join departments d on e.department_id=d.department_id group b
y department_name;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from depat_salary;
+---------------+-----------------+
| sum(e.salary) | department_name |
+---------------+-----------------+
|        105000 | HR              |
|        130000 | Engineering     |
|         90000 | Marketing       |
+---------------+-----------------+
3 rows in set (0.00 sec)
```

6. Create a view called low_salary_employees that lists the emp_id, emp_name, and emp_salary for employees earning less than 30,000. Write a DELETE statement to remove an employee from the view and observe whether the employee is deleted from the original table.



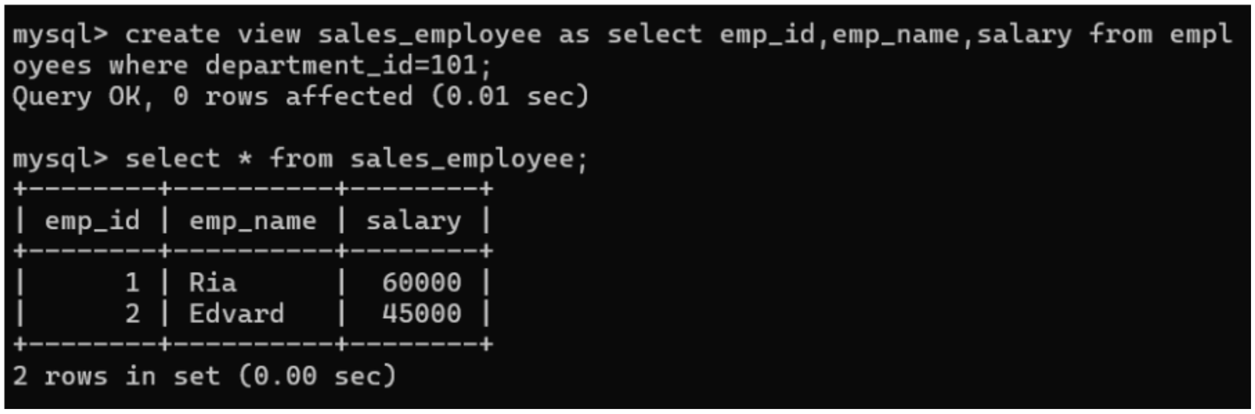
```
mysql> create view low_salary_employees as select emp_id,emp_name,salary fro
m employees where salary<50000;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from low_salary_employees;
+--------+----------+--------+
| emp_id | emp_name | salary |
+--------+----------+--------+
|      2 | Edvard   |  45000 |
+--------+----------+--------+
1 row in set (0.00 sec)
```

7. Create a view called employee_bonus that shows the emp_id, emp_name, and a calculated column bonus (which is 10% of emp_salary) for each employee in the employees table.



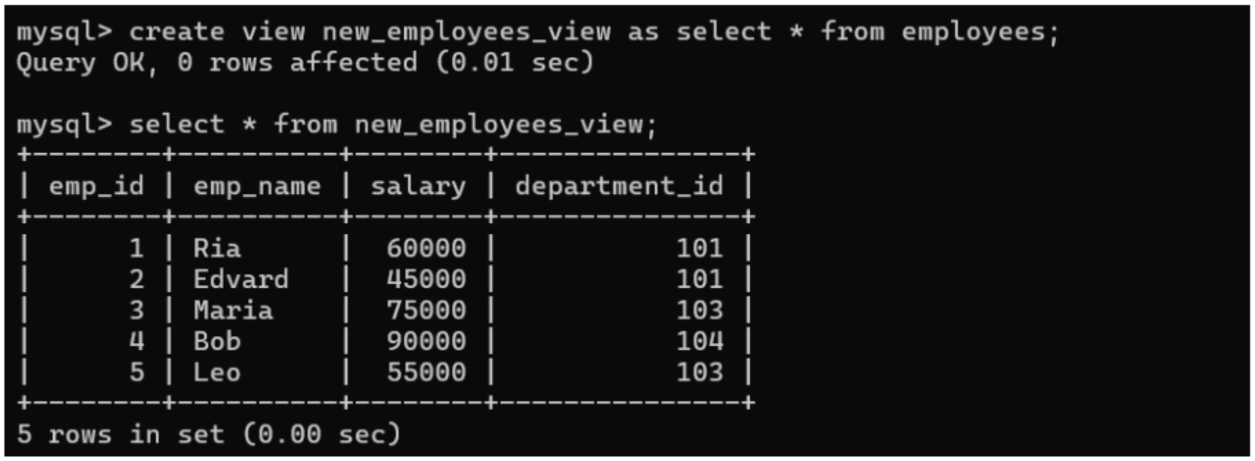
```
mysql> create view employee_bonus as select emp_id,emp_name,salary * 0.10 as
 bonus from employees e;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from employee_bonus;
+--------+----------+---------+
| emp_id | emp_name | bonus   |
+--------+----------+---------+
|      1 | Ria      | 6000.00 |
|      2 | Edvard   | 4500.00 |
|      3 | Maria    | 7500.00 |
|      4 | Bob      | 9000.00 |
|      5 | Leo      | 5500.00 |
+--------+----------+---------+
5 rows in set (0.00 sec)
```

8. Create a view named sales_employees that only includes employees from the Sales department. Ensure the view displays emp_id, emp_name, and emp_salary.



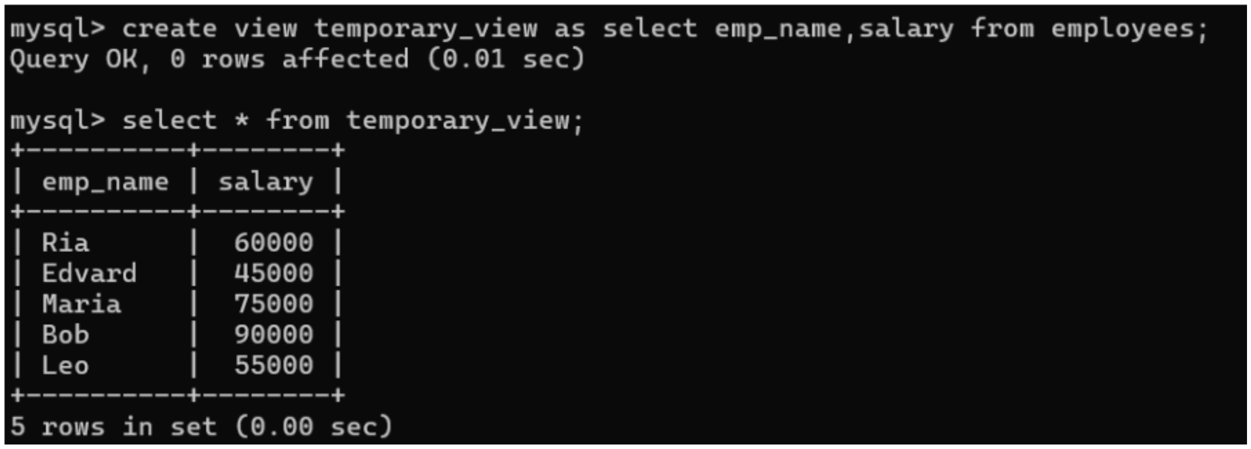
```
mysql> create view sales_employee as select emp_id,emp_name,salary from empl
oyees where department_id=101;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from sales_employee;
+--------+----------+--------+
| emp_id | emp_name | salary |
+--------+----------+--------+
|      1 | Ria      |  60000 |
|      2 | Edvard   |  45000 |
+--------+----------+--------+
2 rows in set (0.00 sec)
```

9. Create a simple view called new_employees_view that includes all columns from the employees table. Try inserting a new employee into the view and check if the data is inserted into the original employees table.



```
mysql> create view new_employees_view as select * from employees;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from new_employees_view;
+--------+----------+--------+---------------+
| emp_id | emp_name | salary | department_id |
+--------+----------+--------+---------------+
|      1 | Ria      |  60000 |           101 |
|      2 | Edvard   |  45000 |           101 |
|      3 | Maria    |  75000 |           103 |
|      4 | Bob      |  90000 |           104 |
|      5 | Leo      |  55000 |           103 |
+--------+----------+--------+---------------+
5 rows in set (0.00 sec)
```

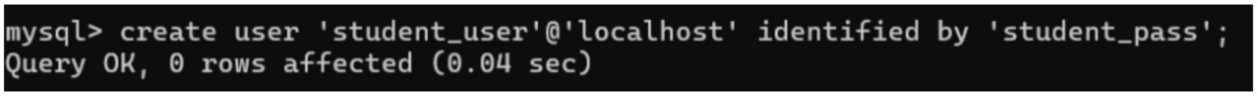
10. Create a view called temporary_view that displays emp_name and emp_salary. After querying the view, write a statement to drop the view from the database.



```
mysql> create view temporary_view as select emp_name,salary from employees;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from temporary_view;
+----------+--------+
| emp_name | salary |
+----------+--------+
| Ria      |  60000 |
| Edvard   |  45000 |
| Maria    |  75000 |
| Bob      |  90000 |
| Leo      |  55000 |
+----------+--------+
5 rows in set (0.00 sec)
```

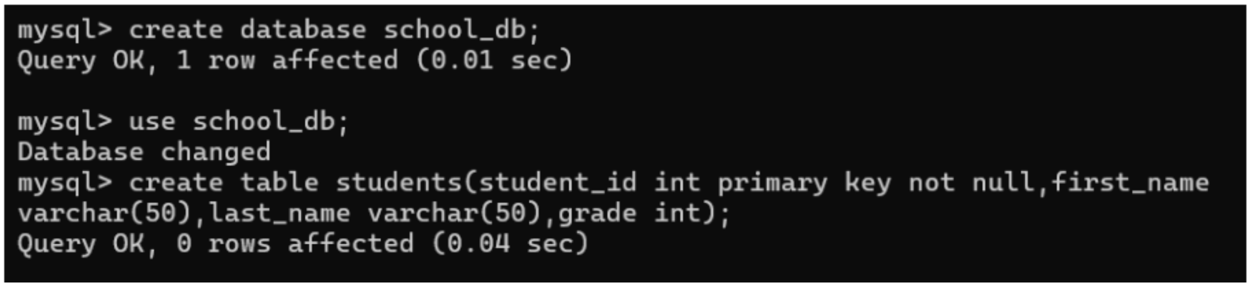
PART 2

1. Creating a New User:
    a. Log in to the MySQL server as the root user.
    b. Create a new user called student_user with the password student_pass who will only
be able to log in from localhost.
CREATE USER 'student_user'@'localhost' IDENTIFIED BY 'student_pass';



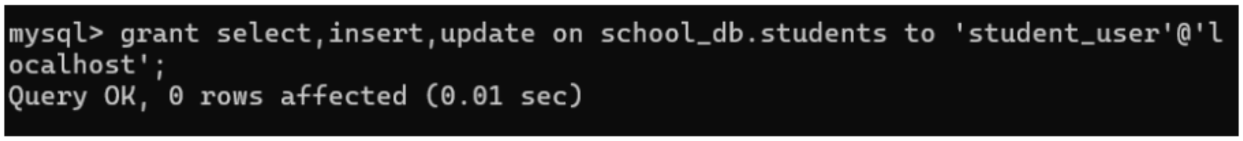
```
mysql> create user 'student_user'@'localhost' identified by 'student_pass';
Query OK, 0 rows affected (0.04 sec)
```

2. Create a Table:
    a. Use or create a database called school_db.
    b. Inside the school_db, create a table called students with the following fields:
        i. student_id (INT, Primary Key, Not Null)
        ii. first_name (VARCHAR(50), Nullable)
        iii. last_name (VARCHAR(50), Nullable)
        iv. grade (INT, Nullable)



```
mysql> create database school_db;
Query OK, 1 row affected (0.01 sec)

mysql> use school_db;
Database changed
mysql> create table students(student_id int primary key not null,first_name
varchar(50),last_name varchar(50),grade int);
Query OK, 0 rows affected (0.04 sec)
```

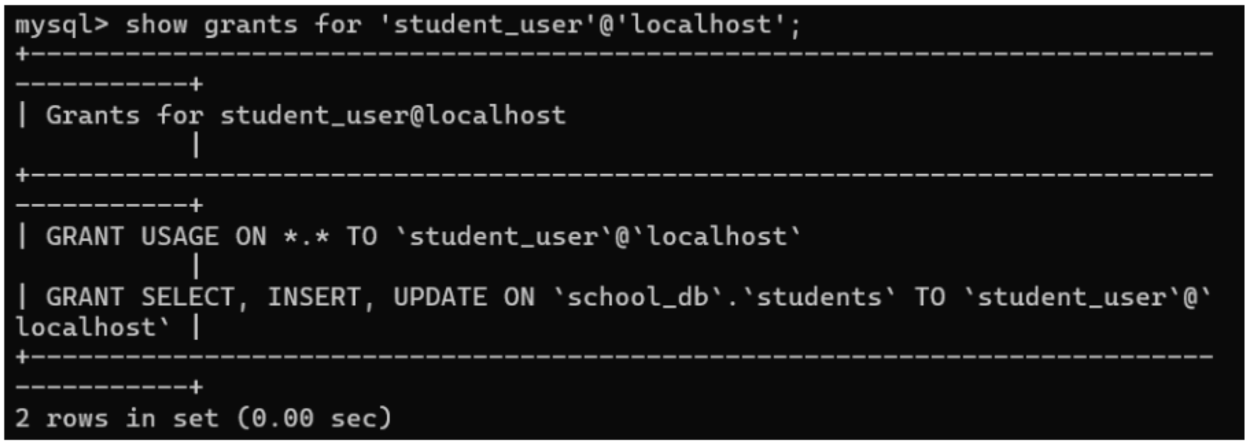
3. Grant Privileges to the User:
    a. Grant SELECT, INSERT, and UPDATE privileges on the students table to the
Student_user.
GRANT SELECT, INSERT, UPDATE ON school_db.students TO 'student_user'@'localhost';



```
mysql> grant select,insert,update on school_db.students to 'student_user'@'l
ocalhost';
Query OK, 0 rows affected (0.01 sec)
```

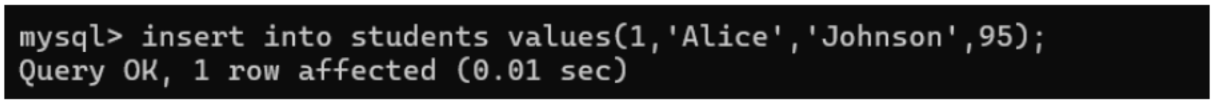
4. Verify the Granted Privileges:
    a. As the root user, check the privileges granted to student_user on the students table.

SHOW GRANTS FOR 'student_user'@'localhost';

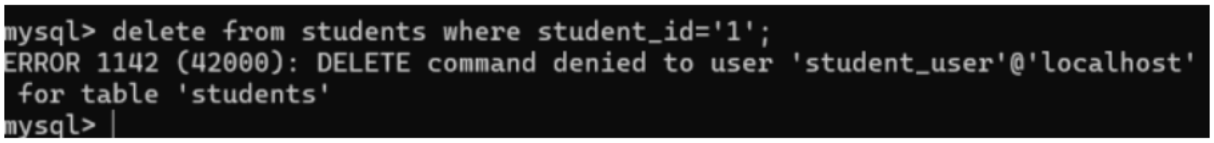


```
mysql> show grants for 'student_user'@'localhost';
+----------------------------------------------------------------------------
-----------+
| Grants for student_user@localhost
           |
+----------------------------------------------------------------------------
-----------+
| GRANT USAGE ON *.* TO `student_user`@`localhost`
           |
| GRANT SELECT, INSERT, UPDATE ON `school_db`.`students` TO `student_user`@`
localhost` |
+----------------------------------------------------------------------------
-----------+
2 rows in set (0.00 sec)
```

5. Test User Access:
    a. Log in as student_user and try inserting a new record into the students table. Use the following data:
        i. student_id: 1
        ii. first_name: 'Alice'
        iii. last_name: 'Johnson'
        iv. grade: 95
    b. Attempt to run a DELETE operation on the students table and observe the result.



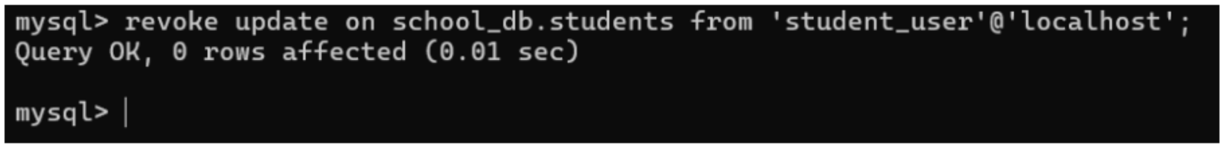
```
mysql> insert into students values(1,'Alice','Johnson',95);
Query OK, 1 row affected (0.01 sec)
```



```
mysql> delete from students where student_id='1';
ERROR 1142 (42000): DELETE command denied to user 'student_user'@'localhost'
 for table 'students'
mysql> |
```

6. Revoke Privileges:
    a. As the root user, revoke the UPDATE privilege from student_user on the students table.

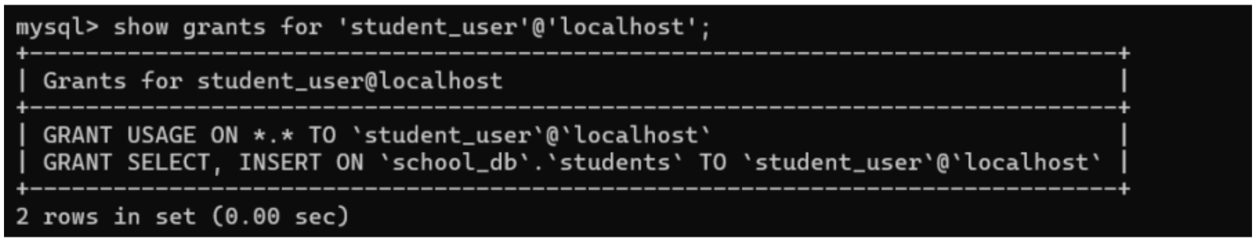
REVOKE UPDATE ON school_db.students FROM 'student_user'@'localhost';



```
mysql> revoke update on school_db.students from 'student_user'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> |
```

7. Verify Revoked Privileges:
    a. As the root user, verify the updated privileges of student_user.

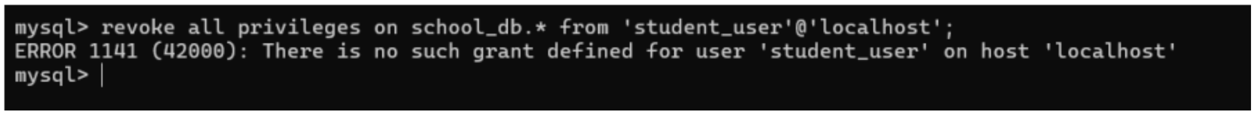
SHOW GRANTS FOR 'student_user'@'localhost';



```
mysql> show grants for 'student_user'@'localhost';
+--------------------------------------------------------------------------+
| Grants for student_user@localhost                                        |
+--------------------------------------------------------------------------+
| GRANT USAGE ON *.* TO `student_user`@`localhost`                         |
| GRANT SELECT, INSERT ON `school_db`.`students` TO `student_user`@`localhost` |
+--------------------------------------------------------------------------+
2 rows in set (0.00 sec)
```

8. Revoking All Privileges:
    a. Finally, revoke all privileges from student_user on the school_db database.

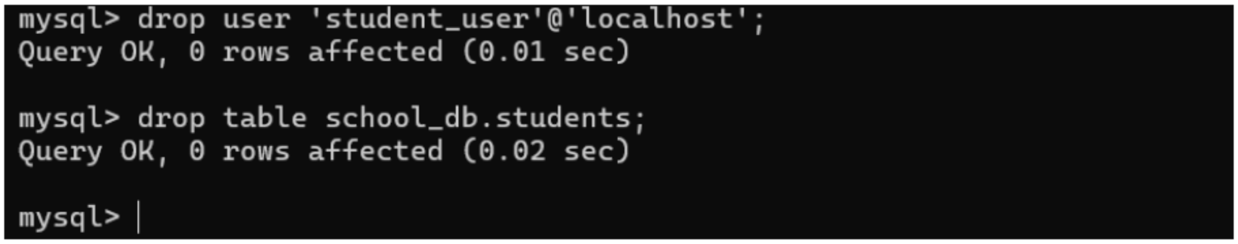
REVOKE ALL PRIVILEGES ON school_db.* FROM 'student_user'@'localhost';



```
mysql> revoke all privileges on school_db.* from 'student_user'@'localhost';
ERROR 1141 (42000): There is no such grant defined for user 'student_user' on host 'localhost'
mysql>
```

9. Cleanup (Optional):
    a. Delete the student_user and students table once you've completed the lab.

DROP USER 'student_user'@'localhost';
DROP TABLE school_db.students;



```
mysql> drop user 'student_user'@'localhost';
Query OK, 0 rows affected (0.01 sec)

mysql> drop table school_db.students;
Query OK, 0 rows affected (0.02 sec)

mysql>
```