Assignment: SQL Tasks for SpecialForce Private Limited - Database Operations (MySQL) : Answer

# Creating Database Table.

mysql> use assignment;

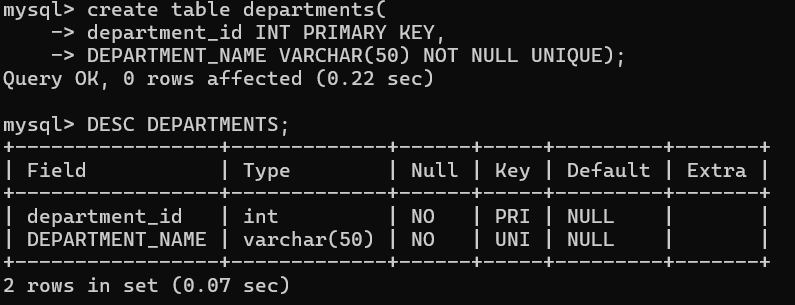
Database changed

mysql> create table departments(

-> department\_id INT PRIMARY KEY,

-> DEPARTMENT\_NAME VARCHAR(50) NOT NULL UNIQUE);

Query OK, 0 rows affected (0.06 sec)



# Creating Employee Table.

# mysql> CREATE TABLE EMPLOYEES (

# -> EMPLOYEE\_ID INT PRIMARY KEY,

# -> FIRST\_NAME VARCHAR(20) NOT NULL,

# -> LAST\_NAME VARCHAR(20) NOT NULL,

# -> EMAIL VARCHAR(50) NOT NULL UNIQUE,

# -> HIRE\_DATE DATE NOT NULL,

# -> SALARY INT NOT NULL,

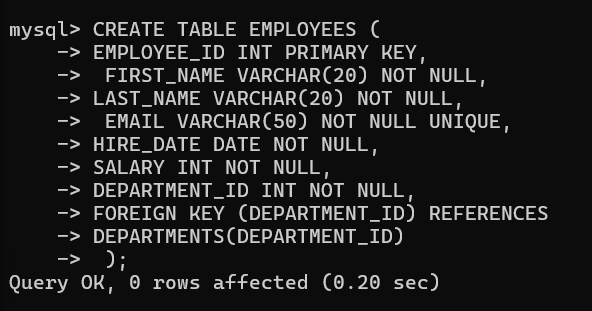
# -> DEPARTMENT\_ID INT NOT NULL,

# -> FOREIGN KEY (DEPARTMENT\_ID) REFERENCES

# -> DEPARTMENTS(DEPARTMENT\_ID)

# -> );

# Query OK, 0 rows affected (0.20 sec)



# Creating Project Table

mysql> CREATE TABLE Projects (

-> project\_id INT PRIMARY KEY,

-> project\_name VARCHAR(50) NOT NULL UNIQUE,

-> start\_date DATE,

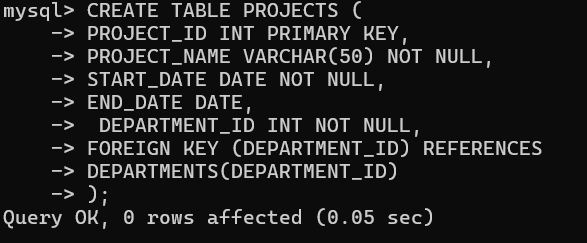
-> end\_date DATE,

-> department\_id INT NOT NULL,

-> FOREIGN KEY (department\_id) REFERENCES Departments(department\_id)

-> );

Query OK, 0 rows affected (0.05 sec)



# Inserting Data into Departments Table and display it .

mysql> INSERT INTO

-> DEPARTMENTS(DEPARTMENT\_ID,

-> DEPARTMENT\_NAME)

-> VALUES

-> (1, 'IT'),

-> (2, 'HR'),

-> (3, 'Sales'),

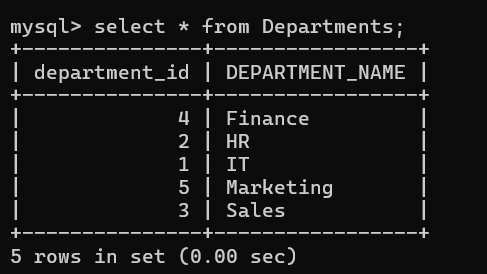
-> (4, 'Finance'),

-> (5, 'Marketing');

Query OK, 5 rows affected (0.06 sec)

Records: 5 Duplicates: 0 Warnings: 0

mysql> select \* from Departments;



# Inserting Data into Employees Table and display it .

mysql> INSERT INTO Employees (employee\_id, first\_name, last\_name, email, hire\_date, salary, department\_id) VALUES

-> (101, 'Ravi', 'Sharma', ['ravi.sharma@specialforce.com',](mailto:%27ravi.sharma@specialforce.com) '2017-05-15', 55000, 1),

-> (102, 'Neha', 'Kapoor', ['neha.kapoor@specialforce.com',](mailto:%27neha.kapoor@specialforce.com) '2019-03-23', 48000, 2),

-> (103, 'Jyoti', 'Verma', ['jyoti.verma@specialforce.com',](mailto:%27jyoti.verma@specialforce.com) '2020-11-02', 60000, 1),

-> (104, 'Anil', 'Patil', ['anil.patil@specialforce.com',](mailto:%27anil.patil@specialforce.com) '2018-09-18', 70000, 3),

-> (105, 'Pooja', 'Singh', ['pooja.singh@specialforce.com',](mailto:%27pooja.singh@specialforce.com) '2021-06-10', 40000, 4),

-> (106, 'Sanjay', 'Iyer', ['sanjay.iyer@specialforce.com',](mailto:%27sanjay.iyer@specialforce.com) '2018-01-22', 75000, 3),

-> (107, 'Jatin', 'Reddy', ['jatin.reddy@specialforce.com',](mailto:%27jatin.reddy@specialforce.com) '2021-12-12', 85000, 2),

-> (108, 'Shreya', 'Mehta', ['shreya.mehta@specialforce.com',](mailto:%27shreya.mehta@specialforce.com) '2022-04-19', 30000, 5),

-> (109, 'Rajesh', 'Gupta', ['rajesh.gupta@specialforce.com',](mailto:%27rajesh.gupta@specialforce.com) '2020-08-11', 90000, 1),

-> (110, 'Kavita', 'Nair', ['kavita.nair@specialforce.com',](mailto:%27kavita.nair@specialforce.com) '2021-02-07', 50000, 2); Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0

mysql> select \* from Employees;



# Inserting Data into Projects Table and display it .

mysql> INSERT INTO Projects (project\_id, project\_name, start\_date, end\_date, department\_id) VALUES

-> (201, 'Project Phoenix', '2021-01-15', '2022-07-30', 1),

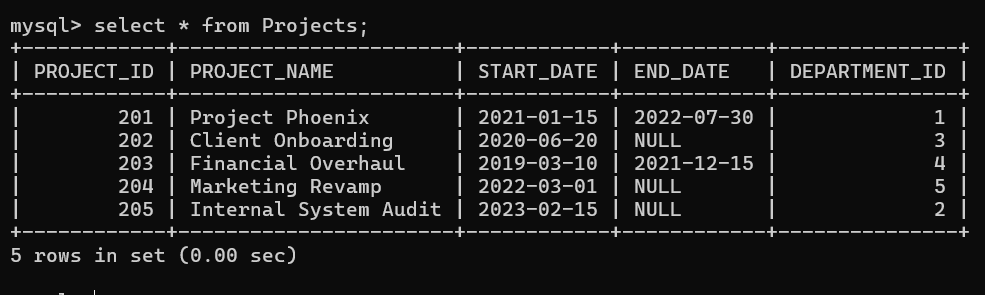
-> (202, 'Client Onboarding', '2020-06-20', NULL, 3),

-> (203, 'Financial Overhaul', '2019-03-10', '2021-12-15', 4),

-> (204, 'Marketing Revamp', '2022-03-01', NULL, 5),

-> (205, 'Internal System Audit', '2023-02-15', NULL, 2); Query OK, 5 rows affected (0.01 sec)

Records: 5 Duplicates: 0 Warnings: 0



Queries to Perform:

 Query 1: Write a query to retrieve the first name, last name, and department name of all employees. If an employee does not belong to any department, the department name should be NULL.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name,

-> Departments.department\_name

-> FROM

-> EMPLOYEES

-> LEFT JOIN DEPARMENTS ON

-> Employees.department\_id = Departments.department\_id; ERROR 1146 (42S02): Table 'assignment.deparments' doesn't exist mysql> SELECT

-> E.first\_name,

-> E.last\_name,

-> D.department\_name

-> FROM

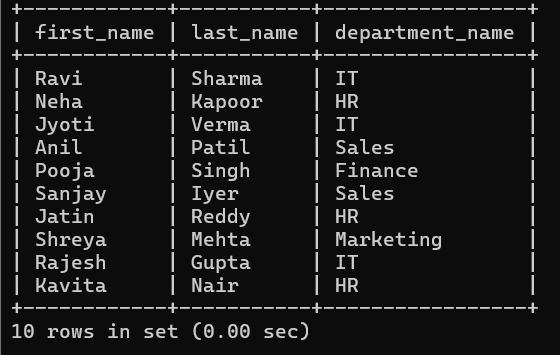
-> Employees E

-> LEFT JOIN

-> Departments D

-> ON

-> E.department\_id = D.department\_id;

****

 Query 2: Write a query to find all employees in the IT department who earn more than ₹50,000.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name,

-> Employees.salary,

-> Departments.department\_name

-> FROM

-> Employees

-> JOIN

-> Departments

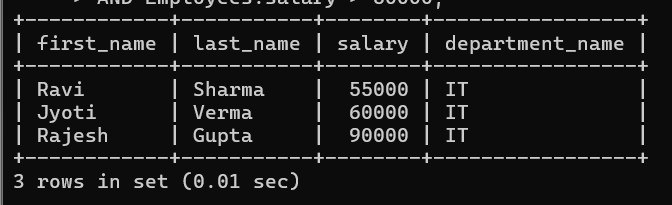
-> ON

-> Employees.department\_id = Departments.department\_id

-> WHERE

-> Departments.department\_name = 'IT'

-> AND Employees.salary > 50000;

****

 Query 3: Write a query to list the first name, last name, and email of all employees whose first name starts with 'J' and whose email contains specialforce.com.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name,

-> Employees.email

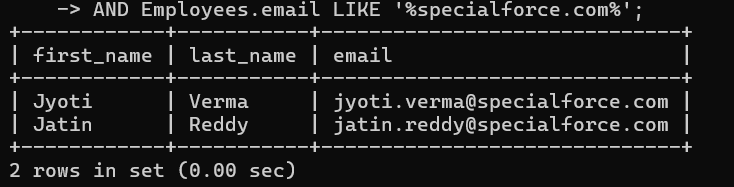
-> FROM

-> Employees

-> WHERE

-> Employees.first\_name LIKE 'J%'

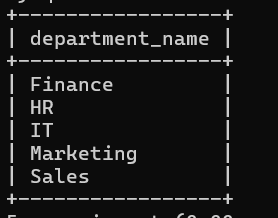
-> AND Employees.email LIKE '%specialforce.com%';



 Query 4: Write a query to find all the distinct department names in the Departments table.



mysql> SELECT DISTINCT Departments.department\_name FROM Departments;



 Query 5: Write a query to calculate the total salary expenditure of each department.



mysql> SELECT

-> Departments.department\_name,

-> SUM(Employees.salary)

-> FROM

-> Employees

-> JOIN

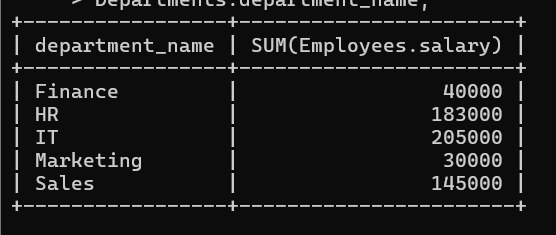
-> Departments

-> ON

-> Employees.department\_id = Departments.department\_id

-> GROUP BY

-> Departments.department\_name;



Query 6: Write a query to find the average salary of employees in the Finance department.



mysql> SELECT AVG(Employees.salary)

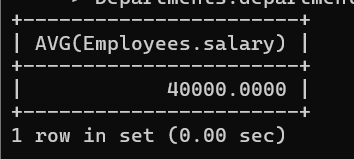
-> FROM Employees

-> join Departments ON

-> Employees.department\_id = Departments.department\_id

-> WHERE

-> Departments.department\_name = 'Finance';



 Query 7: Write a query to find the minimum and maximum salaries of employees in the Sales department.



mysql> SELECT

-> MIN(Employees.salary),

-> MAX(Employees.salary)

-> FROM

-> Employees

-> JOIN

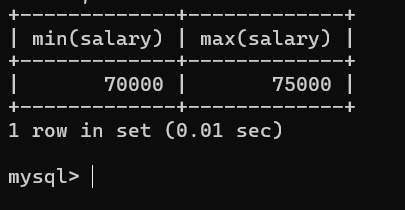
-> Departments

-> ON

-> Employees.department\_id = Departments.department\_id

-> WHERE

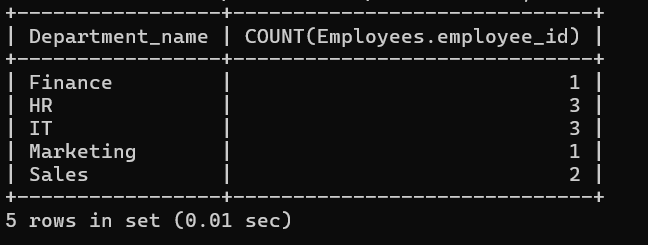
-> Departments.department\_name = 'Sales';



 Query 8: Write a query to count the number of employees in each department.



mysql> SELECT Department\_name,COUNT(Employees.employee\_id) FROM Employees JOIN Departments ON Employees.department\_id = Departments.department\_id GROUP BY Departments.department\_name;



 Query 9: Write a query to find all employees who were hired between January 1, 2018, and December 31, 2020. Sort the result by hire date in ascending order.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name,

-> Employees.hire\_date

-> FROM

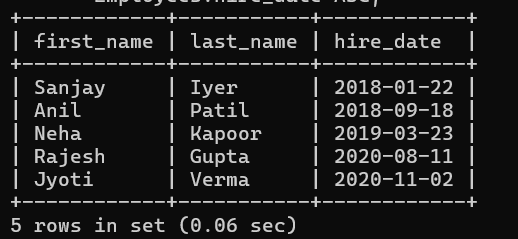
-> Employees

-> WHERE

-> Employees.hire\_date BETWEEN '2018-01-01' AND '2020-12-31'

-> ORDER BY

-> Employees.hire\_date ASC;



 Query 10: Write a query to list all employees who do not have an email address.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name

-> FROM

-> Employees

-> WHERE

-> Employees.email IS NULL;

Empty set (0.00 sec)

 Query 11: Write a query to find all employees who work in HR, Finance, or IT departments.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name,

-> Departments.department\_name

-> FROM

-> Employees

-> JOIN

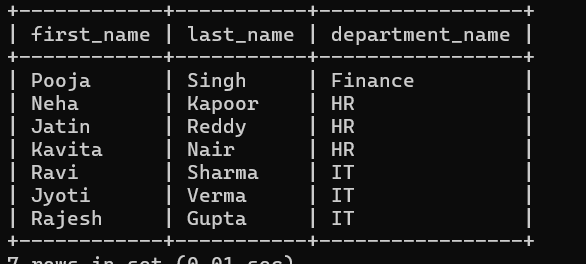
-> Departments

-> ON

-> Employees.department\_id = Departments.department\_id

-> WHERE

-> Departments.department\_name IN ('HR', 'Finance', 'IT');

 Query 12: Write a query to list the first name, last name, and salary of

employees earning between ₹30,000 and ₹70,000. Sort the results by salary in descending order.



mysql> SELECT

-> Employees.first\_name,

-> Employees.last\_name,

-> Employees.salary

-> FROM

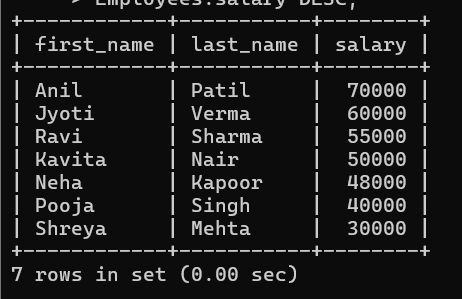
-> Employees

-> WHERE

-> Employees.salary BETWEEN 30000 AND 70000

-> ORDER BY

-> Employees.salary DESC;



**Transaction Management Tasks:**

Use transaction control statements to manage the salary updates as follows:

 Task 1: Increase HR Salaries:

Write a query to increase the salaries of all employees in the HR department by 5%. Start a transaction before applying the changes.



mysql> UPDATE Employees

-> SET salary = salary \* 1.05

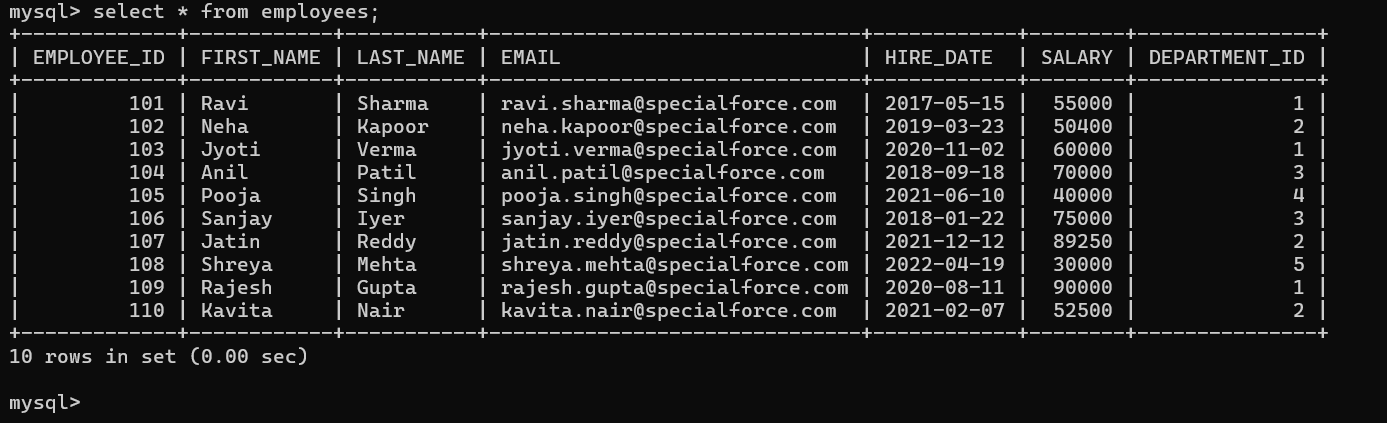
-> WHERE department\_id = (SELECT

-> department\_id FROM Departments WHERE department\_name = 'HR');

Query OK, 3 rows affected (0.01 sec)

Rows matched: 3 Changed: 3 Warnings: 0

mysql> select \* from employees;



 Task 2: Savepoint Before Sales Increase:

Set a savepoint before increasing the salaries of employees in the Sales department by 3%.



mysql>

mysql> UPDATE Employees

-> SET salary = salary \* 1.03

-> WHERE department\_id = (

-> SELECT department\_id FROM Departments WHERE department\_name = 'Sales' );

Query OK, 2 rows affected (0.00 sec)

Rows matched: 2 Changed: 2 Warnings: 0

 Task 3: Rollback Sales Salary Increase:

Rollback to the savepoint created before the Sales salary increase.



mysql> ROLLBACK TO SAVEPOINT before\_sales\_update;

Query OK, 0 rows affected (0.01 sec)

mysql> select \* from Employees;



 Task 4: Commit the Transaction:

After rolling back the Sales increase, commit the changes made to the HR department salaries.



mysql> COMMIT;

Query OK, 0 rows affected (0.01 sec)

 Query 13: Write a query to join the Employees and Departments tables to list employees and their department names. Make sure all employees are included, even if they don’t belong to any department.

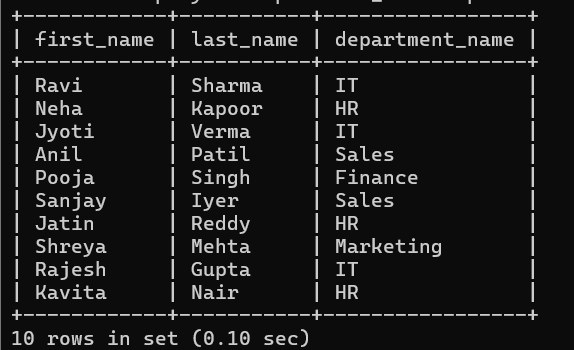


mysql> SELECT Employees.first\_name, Employees.last\_name, Departments.department\_name

-> FROM Employees

-> LEFT JOIN Departments

-> ON Employees.department\_id = Departments.department\_id;

 Query 14: Write a query to list employees who are working on projects that started after January 1, 2023.



mysql> SELECT Employees.first\_name, Employees.last\_name, Projects.project\_name, Projects.start\_date

-> FROM Employees

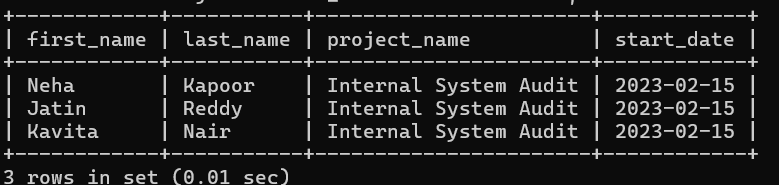
-> JOIN Departments

-> ON Employees.department\_id = Departments.department\_id

-> JOIN Projects

-> ON Departments.department\_id = Projects.department\_id

-> WHERE Projects.start\_date > '2023-01-01';

****

 Query 15: Write a query to list all departments, even those without any employees assigned.

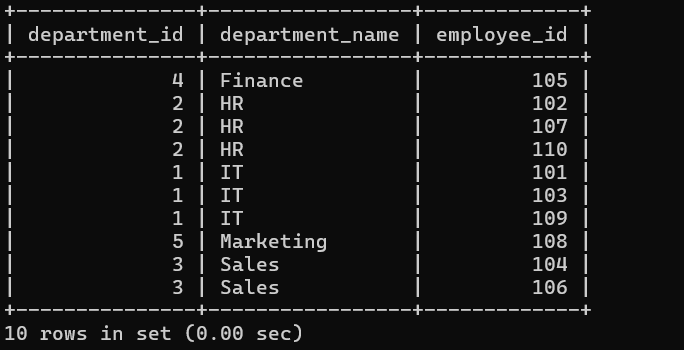


mysql> SELECT Departments.department\_id, Departments.department\_name, Employees.employee\_id

-> FROM Departments

-> LEFT JOIN Employees

-> ON Departments.department\_id = Employees.department\_id;

****

 Query 16: Write a query to find the employee with the highest salary in each department.



mysql> SELECT employee\_id, first\_name, last\_name, department\_id, salary

-> FROM Employees

-> WHERE(department\_id, salary) IN

-> (SELECT department\_id, MAX(salary)

-> FROM Employees

-> GROUP BY department\_id);



 Query 17: Write a query to remove all data from the Employees table but keep the structure intact.



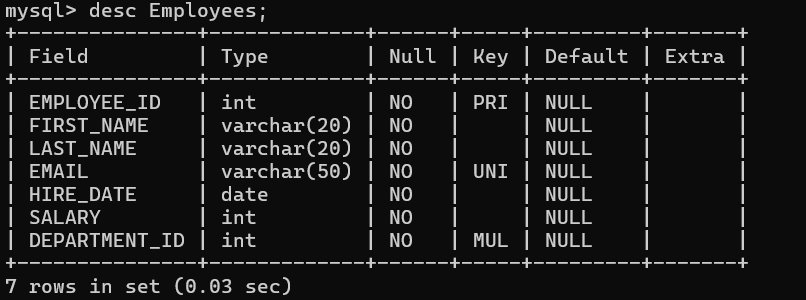
mysql> TRUNCATE TABLE Employees;

Query OK, 0 rows affected (0.07 sec)

mysql> select \* from Employees;

Empty set (0.01 sec)

mysql> desc Employees;



 Query 18: Write a query to drop the Projects table from the database.



mysql> DROP TABLE projects;

Query OK, 0 rows affected (0.04 sec)

mysql> desc projects;

ERROR 1146 (42S02): Table 'assignment.projects' doesn't exist

 Query 19: SpecialForce Private Limited realized they need to store the phone numbers of employees. Write a query to add a new column phone\_number (VARCHAR(15)) to the Employees table using the ALTER statement.



mysql> ALTER TABLE Employees ADD phone\_number VARCHAR(15);

Query OK, 0 rows affected (0.03 sec) Records: 0 Duplicates: 0 Warnings: 0

* Query 20: The company also decided to track the budget for each project. Write a query to add a column budget (DECIMAL(10,2)) to the Projects table.



mysql> ALTER TABLE Projects ADD budget DECIMAL(10,2);

ERROR 1146 (42S02): Table 'assignment.projects' doesn't exist

Query 21: Write a query to find the 2nd largest salary from the Employees table using:



* **A subquery.**
* **The LIMIT clause.**



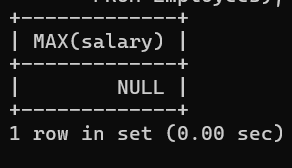
mysql> SELECT MAX(salary)

-> FROM Employees

-> WHERE salary <

-> (SELECT MAX(salary)

-> FROM Employees);



 Query 22: Write a query to find the 3rd largest salary from the Employees table using:

* **A subquery.**
* **The LIMIT clause.**



mysql> SELECT salary

-> FROM Employees

-> ORDER BY salary DESC

-> LIMIT 1 OFFSET 1;

Empty set (0.00 sec)

* Query 23: Write a query to drop the Projects table.

 mysql> DROP TABLE Projects;

* Query 24: Write a query to truncate the Employees table.

 mysql> TRUNCATE TABLE Employees;

Query OK, 0 rows affected (0.06 sec)