Create Tables

```
mysql> CREATE TABLE DEPARTMENT (
    ->     dept_id INT PRIMARY KEY,
    ->     dept_name VARCHAR(30) UNIQUE,
    ->     location VARCHAR(20)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql>
mysql> CREATE TABLE EMPLOYEE (
    ->     emp_id INT PRIMARY KEY,
    ->     emp_name VARCHAR(30) NOT NULL,
    ->     dept_id INT,
    ->     salary INT CHECK (salary > 0),
    ->     city VARCHAR(20),
    ->     doj DATE,
    ->     FOREIGN KEY (dept_id) REFERENCES DEPARTMENT(dept_id)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> CREATE TABLE PROJECT (
    ->     proj_id INT PRIMARY KEY,
    ->     proj_name VARCHAR(30),
    ->     dept_id INT,
    ->     FOREIGN KEY (dept_id) REFERENCES DEPARTMENT(dept_id)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> CREATE TABLE WORKS_ON (
    ->     emp_id INT,
    ->     proj_id INT,
    ->     hours INT CHECK (hours > 0),
    ->     FOREIGN KEY (emp_id) REFERENCES EMPLOYEE(emp_id),
    ->     FOREIGN KEY (proj_id) REFERENCES PROJECT(proj_id)
    -> );
Query OK, 0 rows affected (0.04 sec)
```

ALL tables.

```
+--------+----------+---------+--------+---------+------------+
| emp_id | emp_name | dept_id | salary | city    | doj        |
+--------+----------+---------+--------+---------+------------+
|    101 | Amit     |       1 |  50000 | Delhi   | 2022-01-10 |
|    102 | Ravi     |       2 |  70000 | Mumbai  | 2021-03-15 |
|    103 | Neha     |       1 |  45000 | Delhi   | 2023-07-12 |
|    104 | Priya    |       3 |  40000 | Lucknow | 2020-09-01 |
|    105 | Arjun    |       2 |  80000 | Mumbai  | 2019-11-21 |
|    106 | Karan    |       2 |  60000 | Pune    | 2022-05-18 |
+--------+----------+---------+--------+---------+------------+
6 rows in set (0.00 sec)

mysql> select *from DEPARTMENT;
+---------+-----------+----------+
| dept_id | dept_name | location |
+---------+-----------+----------+
|       1 | Sales     | Delhi    |
|       2 | IT        | Mumbai   |
|       3 | HR        | Lucknow  |
+---------+-----------+----------+
3 rows in set (0.00 sec)

mysql> select *from PROJECT;
+---------+-------------+---------+
| proj_id | proj_name   | dept_id |
+---------+-------------+---------+
|     201 | Website     |       2 |
|     202 | Marketing   |       1 |
|     203 | Recruitment |       3 |
|     204 | App Dev     |       2 |
+---------+-------------+---------+
4 rows in set (0.00 sec)

mysql> select *from WORKS_ON;
+--------+---------+-------+
| emp_id | proj_id | hours |
+--------+---------+-------+
|    101 |     202 |     5 |
|    102 |     201 |     6 |
|    102 |     204 |     4 |
|    103 |     202 |     3 |
|    104 |     203 |     7 |
|    105 |     201 |     8 |
|    105 |     204 |     5 |
|    106 |     201 |     6 |
|    106 |     204 |     4 |
|    101 |     201 |     2 |
+--------+---------+-------+
10 rows in set (0.00 sec)
```

LEVEL 1 – BASIC WITH JOINS

1. Display employee name, department name and salary.
2. Display all projects along with their department name.
3. Display employee name, project name and hours worked.
4. List employees working in a particular department (of your choice).

```
mysql> SELECT e.emp_name, d.dept_name, e.salary
    -> FROM EMPLOYEE e
    -> JOIN DEPARTMENT d ON e.dept_id=d.dept_id;
+----------+-----------+--------+
| emp_name | dept_name | salary |
+----------+-----------+--------+
| Priya    | HR        |  40000 |
| Ravi     | IT        |  70000 |
| Arjun    | IT        |  80000 |
| Karan    | IT        |  60000 |
| Amit     | Sales     |  50000 |
| Neha     | Sales     |  45000 |
+----------+-----------+--------+
6 rows in set (0.00 sec)

mysql> SELECT p.proj_name, d.dept_name
    -> FROM PROJECT p
    -> JOIN DEPARTMENT d ON p.dept_id=d.dept_id;
+------------+-----------+
| proj_name  | dept_name |
+------------+-----------+
| Recruitment | HR        |
| Website    | IT        |
| App Dev    | IT        |
| Marketing  | Sales     |
+------------+-----------+
4 rows in set (0.00 sec)

mysql> SELECT e.emp_name, p.proj_name, w.hours
    -> FROM WORKS_ON w
    -> JOIN EMPLOYEE e ON w.emp_id=e.emp_id
    -> JOIN PROJECT p ON w.proj_id=p.proj_id;
+----------+------------+-------+
| emp_name | proj_name  | hours |
+----------+------------+-------+
| Ravi     | Website    |     6 |
| Arjun    | Website    |     8 |
| Karan    | Website    |     6 |
| Amit     | Website    |     2 |
| Amit     | Marketing  |     5 |
| Neha     | Marketing  |     3 |
| Priya    | Recruitment |     7 |
| Ravi     | App Dev    |     4 |
| Arjun    | App Dev    |     5 |
| Karan    | App Dev    |     4 |
+----------+------------+-------+
10 rows in set (0.00 sec)

mysql> SELECT emp_name
    -> FROM EMPLOYEE
    -> WHERE dept_id=1;
+----------+
| emp_name |
+----------+
| Amit     |
| Neha     |
+----------+
2 rows in set (0.01 sec)

mysql>
```

LEVEL 2 – AGGREGATE + JOIN

5. Find total salary paid in each department.
6. Find the number of employees in each department.
7. Display project name and total hours worked on each project.
8. Find average salary department-wise.

```
mysql> SELECT d.dept_name, SUM(e.salary) AS total_salary
    -> FROM EMPLOYEE e
    -> JOIN DEPARTMENT d ON e.dept_id=d.dept_id
    -> GROUP BY d.dept_name;
+-----------+--------------+
| dept_name | total_salary |
+-----------+--------------+
| HR        |        40000 |
| IT        |       210000 |
| Sales     |        95000 |
+-----------+--------------+
3 rows in set (0.01 sec)

mysql> SELECT d.dept_name, COUNT(e.emp_id)
    -> FROM EMPLOYEE e
    -> JOIN DEPARTMENT d ON e.dept_id=d.dept_id
    -> GROUP BY d.dept_name;
+-----------+-----------------+
| dept_name | COUNT(e.emp_id) |
+-----------+-----------------+
| HR        |               1 |
| IT        |               3 |
| Sales     |               2 |
+-----------+-----------------+
3 rows in set (0.00 sec)

mysql> SELECT p.proj_name, SUM(w.hours)
    -> FROM WORKS_ON w
    -> JOIN PROJECT p ON w.proj_id=p.proj_id
    -> GROUP BY p.proj_name;
+-------------+--------------+
| proj_name   | SUM(w.hours) |
+-------------+--------------+
| Website     |           22 |
| Marketing   |            8 |
| Recruitment |            7 |
| App Dev     |           13 |
+-------------+--------------+
4 rows in set (0.01 sec)

mysql> SELECT d.dept_name, AVG(e.salary)
    -> FROM EMPLOYEE e
    -> JOIN DEPARTMENT d ON e.dept_id=d.dept_id
    -> GROUP BY d.dept_name;
+-----------+---------------+
| dept_name | AVG(e.salary) |
+-----------+---------------+
| HR        |    40000.0000 |
| IT        |    70000.0000 |
| Sales     |    47500.0000 |
+-----------+---------------+
3 rows in set (0.00 sec)
```

LEVEL 3

9. Display details of employees who earn more than the average salary of all employees.

10. Display the employee(s) who earn the maximum salary.

11. Display employees who work in the same department as a given employee (use
subquery).

12. Display employees who are working on at least one project.

13. Display employees who are NOT working on any project.

14. Display department name where the highest salary employee works.

```
mysql> SELECT * FROM EMPLOYEE
    -> WHERE salary > (SELECT AVG(salary) FROM EMPLOYEE);
+--------+----------+---------+--------+--------+------------+
| emp_id | emp_name | dept_id | salary | city   | doj        |
+--------+----------+---------+--------+--------+------------+
|    102 | Ravi     |       2 |  70000 | Mumbai | 2021-03-15 |
|    105 | Arjun    |       2 |  80000 | Mumbai | 2019-11-21 |
|    106 | Karan    |       2 |  60000 | Pune   | 2022-05-18 |
+--------+----------+---------+--------+--------+------------+
3 rows in set (0.01 sec)

mysql> SELECT * FROM EMPLOYEE
    -> WHERE salary = (SELECT MAX(salary) FROM EMPLOYEE);
+--------+----------+---------+--------+--------+------------+
| emp_id | emp_name | dept_id | salary | city   | doj        |
+--------+----------+---------+--------+--------+------------+
|    105 | Arjun    |       2 |  80000 | Mumbai | 2019-11-21 |
+--------+----------+---------+--------+--------+------------+
1 row in set (0.00 sec)

mysql> SELECT emp_name FROM EMPLOYEE
    -> WHERE dept_id = (
    ->     SELECT dept_id FROM EMPLOYEE WHERE emp_id=101
    -> )
    -> AND emp_id !=101;
+----------+
| emp_name |
+----------+
| Neha     |
+----------+
1 row in set (0.01 sec)

mysql> SELECT DISTINCT e.emp_name
    -> FROM EMPLOYEE e
    -> JOIN WORKS_ON w ON e.emp_id=w.emp_id;
+----------+
| emp_name |
+----------+
| Amit     |
| Ravi     |
| Neha     |
| Priya    |
| Arjun    |
| Karan    |
+----------+
6 rows in set (0.00 sec)

mysql> SELECT emp_name FROM EMPLOYEE
    -> WHERE emp_id NOT IN (SELECT emp_id FROM WORKS_ON);
Empty set (0.01 sec)

mysql> SELECT d.dept_name
    -> FROM DEPARTMENT d
    -> JOIN EMPLOYEE e ON d.dept_id=e.dept_id
    -> WHERE e.salary = (SELECT MAX(salary) FROM EMPLOYEE);
+-----------+
| dept_name |
+-----------+
| IT        |
+-----------+
1 row in set (0.00 sec)
```

LEVEL-4

15. Display employees whose salary is greater than the average salary of their own department.

16. Find departments where more than 2 employees are working.

17. Display the project having maximum total hours.

18. Display employee name who works on more than one project.

19. Find the department which has the highest total salary.

20. Display employees who work on all projects of their department.

```
mysql> SELECT emp_name
    -> FROM EMPLOYEE e
    -> WHERE salary > (
    ->     SELECT AVG(salary)
    ->     FROM EMPLOYEE
    ->     WHERE dept_id=e.dept_id
    -> );
+----------+
| emp_name |
+----------+
| Amit     |
| Arjun    |
+----------+
2 rows in set (0.01 sec)

mysql> SELECT dept_id
    -> FROM EMPLOYEE
    -> GROUP BY dept_id
    -> HAVING COUNT(emp_id)>2;
+---------+
| dept_id |
+---------+
|       2 |
+---------+
1 row in set (0.00 sec)

mysql> SELECT proj_name
    -> FROM PROJECT
    -> WHERE proj_id = (
    ->     SELECT proj_id
    ->     FROM WORKS_ON
    ->     GROUP BY proj_id
    ->     ORDER BY SUM(hours) DESC
    ->     LIMIT 1
    -> );
+-----------+
| proj_name |
+-----------+
| Website   |
+-----------+
1 row in set (0.00 sec)
```

```
mysql> SELECT emp_id
    -> FROM WORKS_ON
    -> GROUP BY emp_id
    -> HAVING COUNT(proj_id)>1;
+--------+
| emp_id |
+--------+
|    101 |
|    102 |
|    105 |
|    106 |
+--------+
4 rows in set (0.00 sec)

mysql> SELECT d.dept_name
    -> FROM EMPLOYEE e
    -> JOIN DEPARTMENT d ON e.dept_id=d.dept_id
    -> GROUP BY d.dept_name
    -> ORDER BY SUM(e.salary) DESC
    -> LIMIT 1;
+-----------+
| dept_name |
+-----------+
| IT        |
+-----------+
1 row in set (0.01 sec)

mysql> SELECT e.emp_name
    -> FROM EMPLOYEE e
    -> WHERE NOT EXISTS (
    ->     SELECT p.proj_id
    ->     FROM PROJECT p
    ->     WHERE p.dept_id=e.dept_id
    ->     AND p.proj_id NOT IN (
    ->         SELECT proj_id FROM WORKS_ON w WHERE w.emp_id=e.emp_id
    ->     )
    -> );
+----------+
| emp_name |
+----------+
| Amit     |
| Ravi     |
| Neha     |
| Priya    |
| Arjun    |
| Karan    |
+----------+
```

LEVEL-5

21. Display employee names who work on the same project as employees with emp_id =
101 (use subquery).

22. Display second highest salary employee.

23. Find employees who earn more than every employee in ‘Sales’ department.

24. Display project names which have no employees assigned.

25. Find department names where average salary is greater than company average salary.

```
mysql> SELECT DISTINCT emp_name
    -> FROM EMPLOYEE
    -> WHERE emp_id IN (
    ->     SELECT emp_id FROM WORKS_ON
    ->     WHERE proj_id IN (
    ->         SELECT proj_id FROM WORKS_ON WHERE emp_id=101
    ->     )
    -> )
    -> AND emp_id !=101;
+----------+
| emp_name |
+----------+
| Neha     |
| Ravi     |
| Arjun    |
| Karan    |
+----------+
4 rows in set (0.00 sec)

mysql> SELECT * FROM EMPLOYEE
    -> ORDER BY salary DESC
    -> LIMIT 1 OFFSET 1;
+--------+----------+---------+--------+--------+------------+
| emp_id | emp_name | dept_id | salary | city   | doj        |
+--------+----------+---------+--------+--------+------------+
|    102 | Ravi     |       2 |  70000 | Mumbai | 2021-03-15 |
+--------+----------+---------+--------+--------+------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM EMPLOYEE
    -> WHERE salary > ALL (
    ->     SELECT salary FROM EMPLOYEE
    ->     WHERE dept_id = (
    ->         SELECT dept_id FROM DEPARTMENT WHERE dept_name='Sales'
    ->     )
    -> );
+--------+----------+---------+--------+--------+------------+
| emp_id | emp_name | dept_id | salary | city   | doj        |
+--------+----------+---------+--------+--------+------------+
|    102 | Ravi     |       2 |  70000 | Mumbai | 2021-03-15 |
|    105 | Arjun    |       2 |  80000 | Mumbai | 2019-11-21 |
|    106 | Karan    |       2 |  60000 | Pune   | 2022-05-18 |
+--------+----------+---------+--------+--------+------------+
3 rows in set (0.00 sec)

mysql> SELECT proj_name
    -> FROM PROJECT
    -> WHERE proj_id NOT IN (SELECT proj_id FROM WORKS_ON);
Empty set (0.00 sec)

mysql> SELECT dept_name
    -> FROM DEPARTMENT d
    -> JOIN EMPLOYEE e ON d.dept_id=e.dept_id
    -> GROUP BY d.dept_name
    -> HAVING AVG(e.salary) >
    ->     (SELECT AVG(salary) FROM EMPLOYEE);
+-----------+
| dept_name |
+-----------+
| IT        |
+-----------+
```

PART E – UPDATE / DELETE

26. Increase the salary of employees in a particular department by 15%.

27. Delete employees who are not working on any project.

28. Change the department of an employee and reflect in all related tables.

```
mysql> UPDATE EMPLOYEE
    -> SET salary = salary * 1.15
    -> WHERE dept_id=2;
Query OK, 3 rows affected (0.01 sec)
Rows matched: 3  Changed: 3  Warnings: 0

mysql> DELETE FROM EMPLOYEE
    -> WHERE emp_id NOT IN (SELECT emp_id FROM WORKS_ON);
Query OK, 0 rows affected (0.01 sec)

mysql> UPDATE EMPLOYEE
    -> SET dept_id=3
    -> WHERE emp_id=106;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT *FROM EMPLOYEE;
+--------+----------+---------+--------+---------+------------+
| emp_id | emp_name | dept_id | salary | city    | doj        |
+--------+----------+---------+--------+---------+------------+
|    101 | Amit     |       1 |  50000 | Delhi   | 2022-01-10 |
|    102 | Ravi     |       2 |  80500 | Mumbai  | 2021-03-15 |
|    103 | Neha     |       1 |  45000 | Delhi   | 2023-07-12 |
|    104 | Priya    |       3 |  40000 | Lucknow | 2020-09-01 |
|    105 | Arjun    |       2 |  92000 | Mumbai  | 2019-11-21 |
|    106 | Karan    |       3 |  69000 | Pune    | 2022-05-18 |
+--------+----------+---------+--------+---------+------------+
6 rows in set (0.00 sec)

mysql> SELECT *FROM ;
```

PART F – CONSTRAINT CHECKING

29. Try to insert an invalid record violating foreign key and show the error.

30. Try to insert salary as negative and explain why it fails.

```
mysql> INSERT INTO EMPLOYEE VALUES
    -> (110,'Test',5,30000,'Delhi','2024-01-01');
ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`company_db`.`employee`, CONSTRAINT `employee_ib
fk_1` FOREIGN KEY (`dept_id`) REFERENCES `department` (`dept_id`))
mysql> INSERT INTO EMPLOYEE VALUES
    -> (111,'Test2',1,-5000,'Delhi','2024-01-01');
ERROR 3819 (HY000): Check constraint 'employee_chk_1' is violated.
mysql> select *from EMPLOYEE;
+--------+----------+---------+--------+---------+------------+
| emp_id | emp_name | dept_id | salary | city    | doj        |
+--------+----------+---------+--------+---------+------------+
|    101 | Amit     |       1 |  50000 | Delhi   | 2022-01-10 |
|    102 | Ravi     |       2 |  80500 | Mumbai  | 2021-03-15 |
|    103 | Neha     |       1 |  45000 | Delhi   | 2023-07-12 |
|    104 | Priya    |       3 |  40000 | Lucknow | 2020-09-01 |
|    105 | Arjun    |       2 |  92000 | Mumbai  | 2019-11-21 |
|    106 | Karan    |       3 |  69000 | Pune    | 2022-05-18 |
+--------+----------+---------+--------+---------+------------+
6 rows in set (0.00 sec)

mysql>
```