Create a database named *employee*, then import data_science_team.csv proj_table.csv and emp_record_table.csv into the employee database from the given resources.

CREATE DATABASE employee;

USE employee;

Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, and DEPARTMENT from the employee record table, and make a list of employees and details of their department.

SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT FROM emp_record;

Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:

- **less than two**
- **greater than four**
- **between two and four**

SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING FROM emp_record WHERE EMP_RATING < 2;

SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING FROM emp_record WHERE EMP_RATING > 4;

SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING FROM emp_record WHERE EMP_RATING BETWEEN 2 AND 4;

Write a query to concatenate the FIRST_NAME and the LAST_NAME of employees in the *Finance* department from the employee table and then give the resultant column alias as NAME.

SELECT CONCAT(FIRST_NAME, LAST_NAME) AS NAME FROM emp_record WHERE DEPT = 'FINANCE';

Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President).

SELECT MANAGER, COUNT(*) AS NUM_REPORTERS FROM emp_record GROUP BY MANAGER ORDER BY NUM_REPORTERS DESC;

Write a query to list down all the employees from the healthcare and finance departments using union. Take data from the employee record table.

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT FROM emp_record WHERE
DEPT = 'HEALTHCARE'
UNION
SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT FROM emp_record WHERE
DEPT = 'FINANCE';
```

Write a query to list down employee details such as EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPARTMENT, and EMP_RATING grouped by dept. Also include the respective employee rating along with the max emp rating for the department.

```
SELECT emp.EMP_ID, emp.FIRST_NAME, emp.LAST_NAME, emp.ROLE, emp.DEPT,
dept_max_rating.max_rating AS MAX_EMP_RATING_FOR_DEPT FROM
emp_record AS emp
INNER JOIN(
SELECT DEPT, MAX(EMP_RATING) AS max_rating FROM emp_record GROUP BY
DEPT) AS dept_max_rating ON emp.DEPT = dept_max_rating.DEPT ORDER BY
MAX_EMP_RATING_FOR_DEPT DESC
```

Write a query to calculate the minimum and the maximum salary of the employees in each role. Take data from the employee record table.

```
SELECT MIN(SALARY) AS MIN_SALARY, MAX(SALARY) AS MAX_SALARY, ROLE
FROM emp_record GROUP BY ROLE;
```

Write a query to assign ranks to each employee based on their experience. Take data from the employee record table.

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, EXP, RANK() OVER (ORDER BY EXP
DESC) AS EXP_RANK FROM emp_record;
```

Write a query to create a view that displays employees in various countries whose salary is more than six thousand. Take data from the employee record table.

```
CREATE VIEW high_salary_emp_view AS (SELECT * FROM emp_record WHERE
SALARY > 6000);
```

Write a nested query to find employees with experience of more than ten years. Take data from the employee record table.

```
SELECT * FROM emp_record WHERE EMP_ID IN (SELECT EMP_ID FROM
emp_record WHERE EXP > 10);
```

Write a query to create a stored procedure to retrieve the details of the employees whose experience is more than three years. Take data from the employee record table.

```
DELIMITER //

CREATE PROCEDURE get_emp_exp_more_than_three()
BEGIN
    SELECT * FROM emp_record WHERE EXP > 3;
END//

DELIMITER ;

CALL get_emp_exp_more_than_three()
```

Write a query using stored functions in the project table to check whether the job profile assigned to each employee in the data science team matches the organization's set standard.

> **The standard being:**
> **For an employee with experience less than or equal to 2 years assign 'JUNIOR DATA SCIENTIST',**
> **For an employee with the experience of 2 to 5 years assign 'ASSOCIATE DATA SCIENTIST',**
> **For an employee with the experience of 5 to 10 years assign 'SENIOR DATA SCIENTIST',**
> **For an employee with the experience of 10 to 12 years assign 'LEAD DATA SCIENTIST',**
> **For an employee with the experience of 12 to 16 years assign 'MANAGER'.**

```
DELIMITER //

CREATE FUNCTION GetJobProfile(experience INT)
RETURNS VARCHAR(100)
DETERMINISTIC
```

```
BEGIN
    DECLARE job_profile VARCHAR(100);

    IF experience <= 2 THEN
        SET job_profile = 'JUNIOR DATA SCIENTIST';
    ELSEIF experience <= 5 THEN
        SET job_profile = 'ASSOCIATE DATA SCIENTIST';
    ELSEIF experience <= 10 THEN
        SET job_profile = 'SENIOR DATA SCIENTIST';
    ELSEIF experience <= 12 THEN
        SET job_profile = 'LEAD DATA SCIENTIST';
    ELSE
        SET job_profile = 'MANAGER';
    END IF;

    RETURN job_profile;
END //

DELIMITER ;

SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT, EXP, GetJobProfile(EXP) AS
JOB_PROFILE FROM data_science_team;
```

Create an index to improve the cost and performance of the query to find the employee whose FIRST_NAME is 'Eric' in the employee table after checking the execution plan.

```
CREATE INDEX idx_first_name ON emp_record(FIRST_NAME);

EXPLAIN SELECT * FROM emp_record WHERE FIRST_NAME = 'Eric';
```

Write a query to calculate the bonus for all the employees, based on their ratings and salaries (Use the formula: 5% of salary * employee rating).

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT, ROLE, SALARY * 0.05 *
EMP_RATING AS BONUS FROM emp_record;
```

Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table.

```
SELECT CONTINENT, COUNTRY, AVG(SALARY) AS AVERAGE_SALARY FROM
emp_record GROUP BY CONTINENT, COUNTRY;
```