```sql
-- ScienceQtech Employee Performance Mapping Project

-- DESCRIPTION:
/* ScienceQtech is a startup that works in the Data Science field. ScienceQtech has worked on fraud detection, market basket, self-driving cars, supply chain, algorithmic early
detection of lung cancer, customer sentiment, and the drug discovery field. With the annual appraisal cycle around the corner, the HR department has asked you
(Junior Database Administrator) to generate reports on employee details, their performance, and on the project that the employees have undertaken, to analyze the employee database
and extract specific data based on different requirements.*/

-- Objective:
/*To facilitate a better understanding, managers have provided ratings for each employee which will help the HR department to finalize the employee performance mapping.
As a DBA, you should find the maximum salary of the employees and ensure that all jobs are meeting the organization's profile standard. You also need to calculate bonuses to find
extra cost for expenses. This will raise the overall performance of the organization by ensuring that all required employees receive training.*/

-- The task to be performed:
-- Task01: 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 IF NOT EXISTS employee;
USE employee;
-- Select schemas window employee database then Right click on Table - select Table data import wizard - select file path - Import data

DESCRIBE employee.data_science_team;
DESCRIBE employee.emp_record_table;
DESCRIBE employee.proj_table;

SELECT * FROM employee.data_science_team;
SELECT * FROM employee.emp_record_table;
SELECT * FROM employee.proj_table;

SET FOREIGN_KEY_CHECKS=0;
SET GLOBAL FOREIGN_KEY_CHECKS=0;

ALTER TABLE employee.data_science_team ADD FOREIGN KEY (EMP_ID) REFERENCES employee.emp_record_table (EMP_ID);
ALTER TABLE employee.emp_record_table ADD FOREIGN KEY (PROJ_ID) REFERENCES employee.proj_table (PROJECT_id);

-- Task02: Create an ER diagram for the given employee database.
-- screenshot attached in pdf.

/* Task03: 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 employee.emp_record_table ;

-- Task04: Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPARTMENT, and EMP_RATING if the EMP_RATING is:
-- a) less than two
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING FROM employee.emp_record_table WHERE EMP_RATING < 2;
-- b) greater than four
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING FROM employee.emp_record_table WHERE EMP_RATING > 4;
-- c) between two and four
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING FROM employee.emp_record_table WHERE EMP_RATING BETWEEN 2 AND 4;

/* Task05: 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, DEPT FROM employee.emp_record_table WHERE DEPT = "FINANCE";

-- Task06: Write a query to list only those employees who have someone reporting to them. Also, show the number of reporters (including the President).
SELECT employee. EMP_ID, CONCAT(employee.FIRST_NAME," ",employee.LAST_NAME) AS EMPLOYEE_NAME , manager.MANAGER_ID,
                         CONCAT(manager.FIRST_NAME," ",manager.LAST_NAME) AS MANAGER_NAME , manager.ROLE
                         FROM employee.emp_record_table employee
                         JOIN employee.emp_record_table manager
                         ON employee.MANAGER_ID = manager.EMP_ID;

-- Task07: 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 AS DEPARTMENT FROM employee.emp_record_table WHERE DEPT = "HEALTHCARE" UNION
SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT AS DEPARTMENT FROM employee.emp_record_table WHERE DEPT = "FINANCE";

/* Task08: 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_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, MAX(EMP_RATING) FROM employee.emp_record_table GROUP BY DEPT;

-- Task09: 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 ROLE, MIN(SALARY) AS MINIMUM_SALARY, MAX(SALARY) AS MAXIMUM_SALARY FROM employee.emp_record_table GROUP BY ROLE;

-- Task10: 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, ROLE, DEPT, EXP, RANK() OVER (ORDER BY EXP DESC) AS RANKING FROM employee.emp_record_table;

-- Task11: 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.
SELECT*FROM employee.emp_record_table;
CREATE VIEW EMPLOYEE_SALARY_VIEW AS SELECT EMP_ID, FIRST_NAME, LAST_NAME, COUNTRY, SALARY FROM employee.emp_record_table WHERE SALARY > 6000;
SELECT*FROM EMPLOYEE_SALARY_VIEW;

-- Task12: Write a nested query to find employees with experience of more than ten years. Take data from the employee record table.
SELECT EMP_ID, FIRST_NAME, LAST_NAME, EXP FROM employee.emp_record_table WHERE EMP_ID IN (SELECT EMP_ID FROM emp_record_table WHERE EXP > 10);
-- Another Way
SELECT e.EMP_ID, e.FIRST_NAME, e.LAST_NAME, e.EXP, (SELECT COUNT(DISTINCT p.EMP_ID) FROM employee.emp_record_table p) AS EXP1 FROM employee.emp_record_table e WHERE e.EXP>10;

-- Task13: 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 $$
CREATE PROCEDURE Get_Employee_Exp()
BEGIN
SELECT*FROM employee.emp_record_table WHERE EXP > 3;
END $$
CALL Get_Employee_Exp;

/* Task14: 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 $$
CREATE FUNCTION EMPLOYEE_JOB_PROFILE(EXP int)
RETURNS VARCHAR(100) DETERMINISTIC
BEGIN
DECLARE EMPLOYEE_JOB_PROFILE VARCHAR(100);
IF EXP <= 2 THEN SET EMPLOYEE_JOB_PROFILE = 'JUNIOR DATA SCIENTIST';
ELSEIF EXP BETWEEN 2 AND 5 THEN SET EMPLOYEE_JOB_PROFILE = 'ASSOCIATE DATA SCIENTIST';
ELSEIF EXP BETWEEN 5 AND 10 THEN SET EMPLOYEE_JOB_PROFILE ='SENIOR DATA SCIENTIST';
ELSEIF EXP BETWEEN 10 AND 12 THEN SET EMPLOYEE_JOB_PROFILE ='LEAD DATA SCIENTIST';
ELSEIF EXP BETWEEN 12 AND 16 THEN SET EMPLOYEE_JOB_PROFILE ='MANAGER';
END IF;
RETURN (EMPLOYEE_JOB_PROFILE);
END $$
SELECT EMP_ID, FIRST_NAME, EXP, EMPLOYEE_JOB_PROFILE(EXP) FROM employee.emp_record_table;

-- Task15: 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 employee.emp_record_table(FIRST_NAME);
EXPLAIN SELECT EMP_ID, FIRST_NAME, LAST_NAME FROM employee.emp_record_table WHERE FIRST_NAME = "ERIC";

-- Task16: 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, ROLE, DEPT, EXP, SALARY, EMP_RATING, (SALARY * 5/100)*(EMP_RATING) AS BONUS FROM employee.emp_record_table;

-- Task17: Write a query to calculate the average salary distribution based on the continent and country. Take data from the employee record table.
SELECT EMP_ID, FIRST_NAME, LAST_NAME, COUNTRY, CONTINENT, AVG(SALARY) AS AVERAGE_SALARY FROM employee.emp_record_table GROUP BY CONTINENT, COUNTRY;
```