```sql
-- Creating the database named employee
CREATE DATABASE employee;

-- To set the database employee as the default schema
USE employee;

-- 3. 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_table
ORDER BY DEPT;

-- 4.1 Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER,
DEPARTMENT, and EMP_RATING if the EMP_RATING is less than two
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM emp_record_table
WHERE EMP_RATING < 2;

-- 4.2 Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER,
DEPARTMENT, and EMP_RATING if the EMP_RATING is greater than four
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM emp_record_table
WHERE EMP_RATING > 4;

-- 4.3 Write a query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER,
DEPARTMENT, and EMP_RATING if the EMP_RATING is between two and four
SELECT EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM emp_record_table
WHERE EMP_RATING BETWEEN 2 AND 4;

-- 5. 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_table
WHERE DEPT = 'FINANCE';

-- 6. Write   a   query   to   list   only   those   employees   who
have   someone reporting to them. Also, show the number of reporters
(including the President)
SELECT y.EMP_ID, y.FIRST_NAME AS 'Manager Name', COUNT(x.FIRST_NAME) AS
'Reportees'
FROM emp_record_table x, emp_record_table y
WHERE x.MANAGER_ID = y.EMP_ID
GROUP BY y.EMP_ID;

SELECT @@sql_mode;

SET @@sql_mode = SYS.LIST_DROP(@@sql_mode, 'ONLY_FULL_GROUP_BY');

SELECT y.EMP_ID, y.FIRST_NAME AS 'Manager Name', COUNT(x.FIRST_NAME) AS
'Reportees'
FROM emp_record_table x, emp_record_table y
```

```
WHERE x.MANAGER_ID = y.EMP_ID
GROUP BY y.EMP_ID;

-- 7. 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,DEPT
FROM emp_record_table
WHERE DEPT = 'HEALTHCARE'
UNION
SELECT
      EMP_ID,FIRST_NAME,DEPT
FROM emp_record_table
WHERE DEPT = 'FINANCE';

-- 8. 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,
    EMP_RATING,  -- Employee rating of each Employee.
    MAX(EMP_RATING) OVER (PARTITION BY DEPT) AS 'Max Employee
Rating/Department'  -- Maximum employee rating per department.
FROM emp_record_table;

-- 9. 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
      EMP_ID,
    FIRST_NAME,
    LAST_NAME,
    DEPT,
    ROLE,
    SALARY,
    MIN(SALARY) OVER(PARTITION BY ROLE) AS 'Minimum Salary',
    MAX(SALARY) OVER(PARTITION BY ROLE) AS 'Maximum Salary'
    FROM emp_record_table;

-- 10. 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 'Employee Experience Rank',
    DENSE_RANK() OVER(ORDER BY EXP DESC) AS 'Employee Experience Dense
Rank'
FROM emp_record_table;
```

```
-- 11. 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 emp
      AS
                  SELECT * FROM emp_record_table
        WHERE SALARY > 6000;

SELECT * FROM emp;

-- 12. Write a nested query to find employees with experience of more than
ten years. Take data from the employee record table.
SELECT x.EMP_ID,x.FIRST_NAME,x.EXP
FROM emp_record_table x
WHERE x.EMP_ID IN
                  (
                             SELECT y.EMP_ID
            FROM emp_record_table y
            WHERE y.EXP > 10
                  );

-- 13. 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 sp_emp()
BEGIN
      SELECT * FROM emp_record_table
    WHERE EXP > 3;
END; //
DELIMITER ;

CALL sp_emp();


-- 14. 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 organizationâ€™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 fn_prof_match(e_id VARCHAR(4)) RETURNS VARCHAR(50)
DETERMINISTIC
BEGIN
```

```
      -- Declaration of variables
                          DECLARE exist_exp INT DEFAULT NULL;
                          DECLARE exist_role VARCHAR(24) DEFAULT NULL;
                          DECLARE set_role VARCHAR(24) DEFAULT NULL;

      -- Selecting experience and role from data_science_team table for
the given employee id.
                          SELECT EXP, ROLE INTO exist_exp, exist_role
                          FROM data_science_team
                          WHERE EMP_ID = e_id;

      -- Evaluating the existing experience and assigning role as per
organization's set standard
                          IF exist_exp <=2 THEN
                               SET set_role = "JUNIOR DATA SCIENTIST";
                          ELSEIF exist_exp > 2 AND exist_exp <= 5 THEN
                               SET set_role = "ASSOCIATE DATA
SCIENTIST";
                          ELSEIF exist_exp > 5 AND exist_exp <= 10 THEN
                               SET set_role = "SENIOR DATA SCIENTIST";
                          ELSEIF exist_exp > 10 AND exist_exp <= 12
THEN
                               SET set_role = "LEAD DATA SCIENTIST";
                          ELSEIF exist_exp > 12 AND exist_exp <= 16
THEN
                               SET set_role = "MANAGER";
                          END IF;

      -- Compare set role to existing role and return the result based on
profile matching
                          IF exist_role = set_role THEN
                               RETURN "Congratulations! Profile
matches set standard.";
                          ELSE
                               RETURN "Sorry! Profile doesn't match
set standard.";
                          END IF;
END; //
DELIMITER ;

SHOW FUNCTION STATUS WHERE db = 'employee';

SELECT fn_prof_match('E005');   -- Mention the EMP_ID available in the
database
SELECT fn_prof_match('E007');     -- Mention the EMP_ID not available in
the database


-- 15. Create an index to improve the cost and performance of the query to
find the employee whose FIRST_NAME is â€˜Ericâ€™ in the employee table
after checking the execution plan
SELECT * FROM emp_record_table WHERE FIRST_NAME = 'Eric';      -- Before
creating index
```

```
CREATE INDEX id_emp_fn ON emp_record_table(FIRST_NAME(7));          --
creating an index
SELECT * FROM emp_record_table WHERE FIRST_NAME = 'Eric';      -- After
creating index

-- 16. 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,SALARY,EMP_RATING,(0.05*salary*emp_rating) AS
Bonus
FROM emp_record_table;

-- 17. 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,SALARY,COUNTRY,CONTINENT,
    AVG(SALARY) OVER(PARTITION BY COUNTRY) AS 'Average Country Based
Salary',
    AVG(SALARY) OVER(PARTITION BY CONTINENT) AS 'Average Continent Based
Salary'
FROM emp_record_table;
```