```
create schema employee;
use employee;

-- 3. A query to fetch EMP_ID, FIRST_NAME, LAST_NAME, GENDER, and DEPARTMENT from the employee record table
SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT
FROM
   emp_record_table;


-- 4. EMPLOYEE RATING<2
SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM
   emp_record_table
WHERE
   EMP_RATING < 2;

-- EMPLOYEE RATING>4
SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM
   emp_record_table
WHERE
   EMP_RATING > 4;

-- EMPLOYEE RATING BETWEEN 2 AND 4
SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, GENDER, DEPT, EMP_RATING
FROM
```

```
   emp_record_table
WHERE
   EMP_RATING > 2 AND EMP_RATING < 4;


-- 5. A query to concatenate the FIRST_NAME and the LAST_NAME of employees in the Finance department from the employee table
SELECT
   CONCAT(FIRST_NAME, ' ',LAST_NAME) AS NAME
FROM
   emp_record_table
WHERE
   DEPT = 'FINANCE';



-- 6 A query to list only those employees who have someone reporting to them
SELECT
   t1.EMP_ID, t1.FIRST_NAME, t1.LAST_NAME, t1.ROLE,
   COUNT(t2.EMP_ID) AS 'EMP_COUNT'
FROM
   emp_record_table t1
      INNER JOIN
   emp_record_table t2 ON t1.EMP_ID = t2.MANAGER_ID
                        GROUP BY t1.EMP_ID, t1.FIRST_NAME, t1.LAST_NAME, t1.ROLE
         ORDER BY EMP_COUNT DESC;



-- 7. A query to list down all the employees from the healthcare and finance departments using union
SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, DEPT
```

```
FROM
   emp_record_table T1
WHERE
   DEPT = 'HEALTHCARE'
UNION SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, DEPT
FROM
   emp_record_table T1
WHERE
   DEPT = 'FINANCE';
```

-- 8. A query to list down employee details such as EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPARTMENT, and EMP_RATING grouped by dept

```
SELECT
        EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EMP_RATING, max(EMP_RATING)
OVER
        (PARTITION BY DEPT) AS MAX_EMP_RATING
FROM
        emp_record_table
        ORDER BY DEPT;
```

-- 9. A query to calculate the minimum and the maximum salary of the employees in each role

```
SELECT
        DISTINCT(ROLE), MAX(SALARY) OVER (PARTITION BY ROLE) AS MAX_SALARY, MIN(SALARY) OVER (PARTITION BY ROLE) AS MIN_SALARY
FROM
        emp_record_table;
```

-- A query to assign ranks to each employee based on their experience

```
SELECT EMP_ID, FIRST_NAME, LAST_NAME, ROLE, DEPT, EXP, RANK() OVER(ORDER BY EXP DESC) as OVER_ALL_RANK FROM emp_record_table;
```

```
-- 11. A query to create a view that displays employees in various countries whose salary is more than six thousand
CREATE VIEW emp_salary_view AS
   SELECT
      EMP_ID, FIRST_NAME, LAST_NAME, COUNTRY, SALARY
   FROM
      emp_record_table
   WHERE
      SALARY > 6000;
SELECT
   *
FROM
   emp_salary_view;


-- 12. A nested query to find employees with experience of more than ten years
SELECT
   EMP_ID, FIRST_NAME, LAST_NAME, EXP
FROM
   emp_record_table
WHERE
   EXP IN (SELECT
         EXP
      FROM
         emp_record_table
      WHERE
         EXP > 10)
ORDER BY EXP DESC;
```

-- 13. A query to create a stored procedure to retrieve the details of the employees whose experience is more than three years

```
DELIMITER &&
CREATE PROCEDURE EXP_OVER_3()
BEGIN SELECT*
FROM emp_record_table
WHERE
        EXP>3;
END &&
CALL EXP_OVER_3();
```

-- 14. 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.

```
DELIMITER &&
CREATE FUNCTION Employee_ROLE(
EXP int
)
RETURNS VARCHAR(40)
DETERMINISTIC
BEGIN
DECLARE Employee_ROLE VARCHAR(40);
IF EXP>12 AND 16 THEN
SET Employee_ROLE="MANAGER";
ELSEIF EXP>10 AND 12 THEN
SET Employee_ROLE ="LEAD DATA SCIENTIST";
ELSEIF EXP>5 AND 10 THEN
SET Employee_ROLE ="SENIOR DATA SCIENTIST";
ELSEIF EXP>2 AND 5 THEN
SET Employee_ROLE ="ASSOCIATE DATA SCIENTIST";
ELSEIF EXP<=2 THEN
SET Employee_ROLE ="JUNIOR DATA SCIENTIST";
```

```
END IF;
RETURN (Employee_ROLE);
END &&

SELECT EMP_ID, FIRST_NAME, LAST_NAME, DEPT, EXP, Employee_ROLE(EXP)
FROM data_science_team;

-- 15. 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.
SELECT
   *
FROM
   emp_record_table
WHERE
   FIRST_NAME = 'ERIC';
create index emp_name_index on emp_record_table(FIRST_NAME(50));
EXPLAIN SELECT EMP_ID, FIRST_NAME, LAST_NAME FROM emp_record_table WHERE FIRST_NAME = 'ERIC';

-- 16. 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
   *, ROUND(0.05 * SALARY * EMP_RATING, 0) AS BONUS
FROM
   emp_record_table;

-- 17. A query to calculate the average salary distribution based on the continent and country
SELECT DISTINCT(COUNTRY), CONTINENT,
        ROUND(AVG(SALARY) OVER(PARTITION BY COUNTRY),2) AS AVG_SALARY_IN_COUNTRY,
```

```
   ROUND(AVG(SALARY) OVER(PARTITION BY CONTINENT),2) AS
AVG_SALARY_IN_CONTINENT

FROM

        emp_record_table;
```