```
CREATE DATABASE DB;

USE DB;

CREATE TABLE Employee (
EmpID int NOT NULL,
EmpName Varchar(50),
Gender Char,
Salary int,
City Char(20) );

INSERT INTO Employee
VALUES (1, 'Arjun', 'M', 75000, 'Pune'),
(2, 'Ekadanta', 'M', 125000, 'Bangalore'),
(3, 'Lalita', 'F', 150000 , 'Mathura'),
(4, 'Madhav', 'M', 250000 , 'Delhi'),
(5, 'Visakha', 'F', 120000 , 'Mathura');

SELECT * FROM Employee;

CREATE TABLE EmployeeDetail (
EmpID int NOT NULL,
Project Varchar(50),
EmpPosition Char(20),
DOJ date );

INSERT INTO EmployeeDetail (EmpID, Project, EmpPosition, DOJ)
VALUES
(1, 'P1', 'Executive', STR_TO_DATE('26-01-2019', '%d-%m-%Y')),
(2, 'P2', 'Executive', STR_TO_DATE('04-05-2020', '%d-%m-%Y')),
(3, 'P1', 'Lead', STR_TO_DATE('21-10-2021', '%d-%m-%Y')),
(4, 'P3', 'Manager', STR_TO_DATE('29-11-2019', '%d-%m-%Y')),
(5, 'P2', 'Manager', STR_TO_DATE('01-08-2020', '%d-%m-%Y'));

SELECT * FROM EmployeeDetail;

#Q1: Find the list of employees whose salary ranges between 2L to 3L.
SELECT *
FROM Employee
WHERE Salary Between 200000 AND 300000;

#Q2: Write a query to retrieve the list of employees from the same
city.
SELECT E1.EmpID,E1.EmpName,E1.City
FROM Employee E1 , Employee E2
WHERE E1.City = E2.City AND E1.EmpId != E2.EmpId;

#Q3: Query to find the cumilative sum of Employee's salary
SELECT EmpID, Salary, SUM(Salary) OVER (ORDER BY EmpID) AS
CumulativeSum
FROM Employee;

#Q4 : What’s the male and female employees ratio.
SELECT
    SUM(CASE WHEN gender = 'M' THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS
male_percentage,
    SUM(CASE WHEN gender = 'F' THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS
female_percentage
```

```
FROM Employee;

#Q5 : Write a Query th fetch 50% records from the employee table
SELECT * FROM Employee
WHERE EmpID <= ( SELECT COUNT(EmpID)/2 FROM Employee);

#Q6 : Query to fetch the employee's salary but replace the Last 2
digit's with 'XX'
SELECT
    Salary,
    CONCAT(LEFT(CAST(Salary AS CHAR), CHAR_LENGTH(Salary)-2), 'XX') AS
masked_salary
FROM Employee;

#Q7 : Write a query to fetch even and odd rows from employee table
SELECT *
FROM Employee
WHERE MOD(EmpID, 2) = 1;

SELECT *
FROM Employee
WHERE MOD(EmpID, 2) = 0;

#Q8: Write a query to find all the Employee names whose name:
#• Begin with 'A'
SELECT * FROM Employee WHERE EmpName LIKE 'A%';
#• Contains 'A' alphabet at second place
SELECT * FROM Employee WHERE EmpName LIKE '_a%';
#• Contains 'Y' alphabet at second last place
SELECT * FROM Employee WHERE EmpName LIKE '%y_';
#• Ends with 'L' and contains 4 alphabets
SELECT * FROM Employee WHERE EmpName LIKE '____l';
#• Begins with 'V' and ends with 'A'
SELECT * FROM Employee WHERE EmpName LIKE 'V%a';

#09 : Write a query to find all the Employee names whose name:
#(a) Starting with vowels(a,e,i,o,u) without Duplicates
SELECT DISTINCT EmpName
FROM Employee
WHERE LOWER(EmpName) REGEXP '^[aeiou]';

#(b) Ending with vowels(a,e,i,o,u) without Duplicates
SELECT DISTINCT EmpName
FROM Employee
WHERE LOWER(EmpName) REGEXP '[aeiou]$';

#(c) Starting & Ending with vowels(a,e,i,o,u) without Duplicates
SELECT DISTINCT EmpName
FROM Employee
WHERE LOWER(EmpName) REGEXP '^[aeiou].*[aeiou]$';

#Q10 : Find the Nth Highest Salary from employee table with and
# without using the top/Limit Keywords

-- Replace N with the rank you want (e.g., 2 for 2nd highest)
SELECT Salary
FROM Employee E1
WHERE ( N - 1) = (
    SELECT COUNT(DISTINCT E2.Salary)
    FROM Employee E2
    WHERE E2.Salary > E1.Salary
);

-- Using Limit {3rd highest (N = 3) -> OFFSET 2}
 SELECT DISTINCT Salary
FROM Employee
ORDER BY Salary DESC
LIMIT 1 OFFSET 2;

-- Top 2nd highest salary in SQL Server
SELECT Salary
FROM (
  SELECT DISTINCT Salary
  FROM Employee
  ORDER BY Salary DESC
  LIMIT 2
) AS t
ORDER BY Salary ASC
LIMIT 1;

#Q11 : Write a Query to Find and Remove duplicate records from a table
-- FIND
SELECT EmpID, EmpName, gender, Salary, city,
COUNT(*) AS duplicate_count
FROM Employee
GROUP BY EmpID, EmpName, gender, Salary, city
HAVING COUNT(*) > 1;

-- REMOVE
DELETE FROM Employee
WHERE EmpID IN (
    SELECT EmpID
    FROM (
        SELECT EmpID
        FROM Employee
        GROUP BY EmpID
        HAVING COUNT(*) > 1
    ) AS t
);

#Q12 : Query to retrieve the list of employees working in same project.
WITH CTE AS
      (SELECT e.EmpID, e.EmpName, ed.Project
      FROM Employee AS e
      INNER JOIN EmployeeDetail AS ed
      ON e.EmpID = ed.EmpID)
SELECT c1.EmpName, c2.EmpName, c1.project
FROM CTE c1, CTE c2
WHERE c1.Project = c2.Project AND c1.EmpID != c2.EmpID AND c1.EmpID <
c2.EmpID;

#Q13 : Show the employee with the highest salary for each project
SELECT ed.Project, MAX(e.Salary) AS ProjectSal
FROM Employee AS e
INNER JOIN EmployeeDetail AS ed
ON e.EmpID = ed.EmpID
```

```
GROUP BY Project
ORDER BY ProjectSal DESC;

#Q14 : Query to find the total count of employees joined each year
SELECT EXTRACT(YEAR FROM ed.DOJ) AS JoinYear,
       COUNT(*) AS EmpCount
FROM Employee AS e
INNER JOIN EmployeeDetail AS ed ON e.EmpID = ed.EmpID
GROUP BY JoinYear
ORDER BY JoinYear ASC;

#Q15 : Create 3 groups based on salary col, salary less than 1L is low,
between 1 -
# 2L is medium and above 2L is High

SELECT EmpName, Salary,
      CASE
                 WHEN Salary > 200000 THEN 'High'
                 WHEN Salary >= 100000 AND Salary <= 200000 THEN
'Medium'
                 ELSE 'Low'
      END AS SalaryStatus
FROM Employee;
```