1. **Find the Second Highest Salary**

CREATE TABLE Employees (

EmpID INT PRIMARY KEY,

Name NVARCHAR(50),

Salary INT

)

INSERT INTO Employees VALUES

(1, 'Alice', 5000),

(2, 'Bob', 7000),

(3, 'Charlie', 8000),

(4, 'David', 5000),

(5, 'Eve', 6000)

Main code –

SELECT MAX(Salary) AS SecondHighestSalary

FROM Employees

WHERE Salary < (SELECT MAX(Salary) FROM Employees)

-------------------------------------------------------------

2)**Display all columns for everyone whose last name contains “May”**

Main code –

select\*from empinfo where last like '%may%'

---------------------------------------------------------------

1. **Find Employees Who Earn More Than the Average Salary**

Main code –

SELECT Name, Salary

FROM Employees

WHERE Salary > (SELECT AVG(Salary) FROM Employees);

-------------------------------------------------------------------------------

1. **Find duplicate records in a table based on a specific column**.

SELECT ColumnName, COUNT(\*)

FROM TableName

GROUP BY ColumnName

HAVING COUNT(\*) > 1;

---------------------------------------------------------------

1. **Write a query to get employees with the top 3 salaries.**

SELECT TOP 3 \*

FROM Employees

ORDER BY Salary DESC;

----------------------------------------------------------------------

**6) Write a query to reverse a string.**

DECLARE @str NVARCHAR(100) = 'Interview';

SELECT REVERSE(@str) AS ReversedString;

----------------------------------------------------------------------------------

1. **Retrieve records between '2024-01-01' and '2024-12-31'.**

SELECT \*

FROM TableName

WHERE DateColumn BETWEEN '2024-01-01' AND '2024-12-31';

----------------------------------------------------------------------------

1. **Write a query to find employees with the longest names.**

SELECT EmployeeName

FROM Employee

WHERE LEN(EmployeeName) = (

SELECT MAX(LEN(EmployeeName))

FROM Employee

)

1. **Write a query to find rows where any column has a NULL value.**

SELECT \*

FROM TableName

WHERE Column1 IS NULL OR Column2 IS NULL OR Column3 IS NULL;

--------------------------------------------------------------------------------------------------------------------------------**10) Write a query to delete rows where salary is less than 20,000.**

DELETE FROM Employee

WHERE Salary < 20000;

1. **Write a query to find employees with salaries between 30,000 and 50,000**

SELECT EmployeeName, Salary

FROM Employee

WHERE Salary BETWEEN 30000 AND 50000;

1. **Categorize employees as Low, Medium, or High based on salary.**

SELECT EmployeeName, Salary,

CASE

WHEN Salary < 30000 THEN 'Low'

WHEN Salary BETWEEN 30000 AND 60000 THEN 'Medium'

ELSE 'High'

END AS SalaryCategory

FROM Employee

-------------------------------------------------------------------------------------------------------------------------------

1. **Write a query to reverse the values in a text column.**

SELECT EmployeeName, REVERSE(EmployeeName) AS ReversedName

FROM Employee;

**13) Write a query to fetch unique values from a column or removing the duplicate values from the column**

SELECT DISTINCT ColumnName

FROM TableName;

1. **Swap the values of two columns, ColumnA and ColumnB.**

UPDATE TableName

SET ColumnA = ColumnB,

ColumnB = ColumnA;

1. **Write a query to fetch rows where a column contains a palindrome.**

SELECT ColumnName

FROM TableName

WHERE ColumnName = REVERSE(ColumnName);

----------------------------------------------------------------------------------------------------------------------

1. **Write a query to fetch the employees who joined the earliest and the latest.**

SELECT TOP 1 EmployeeName, HireDate

FROM Employee

ORDER BY HireDate ASC;

SELECT TOP 1 EmployeeName, HireDate

FROM Employee

ORDER BY HireDate DESC;

---------------------------------------------------------------------------------------------------------------------

1. **Fetch all years from a table where the year is a leap year.**

SELECT DISTINCT YEAR(DateColumn) AS LeapYear

FROM TableName

WHERE (YEAR(DateColumn) % 4 = 0 AND YEAR(DateColumn) % 100 <> 0)

OR (YEAR(DateColumn) % 400 = 0);

**18) Write a query to find customers who appear in both TableA and TableB. &&** **Write a query to find rows in TableA that do not exist in TableB.**

SELECT CustomerID

FROM TableA

INTERSECT

SELECT CustomerID

FROM TableB;

SELECT \*

FROM TableA

EXCEPT

SELECT \*

FROM TableB;

----------------------------------------------------------------------------------------------------------------------------

**19) Write a query to group data into ranges (e.g., age groups).**

-- Create a temporary table for demonstration

CREATE TABLE #People (

Name NVARCHAR(50),

Age INT

);

-- Insert sample data

INSERT INTO #People (Name, Age)

VALUES

('Alice', 15),

('Bob', 23),

('Charlie', 37),

('David', 45),

('Eve', 62),

('Frank', 78);

-- Query to group data into age ranges

SELECT

CASE

WHEN Age BETWEEN 0 AND 17 THEN '0-17'

WHEN Age BETWEEN 18 AND 35 THEN '18-35'

WHEN Age BETWEEN 36 AND 50 THEN '36-50'

WHEN Age BETWEEN 51 AND 65 THEN '51-65'

ELSE '66+'

END AS AgeGroup,

COUNT(\*) AS TotalPeople

FROM #People

GROUP BY

CASE

WHEN Age BETWEEN 0 AND 17 THEN '0-17'

WHEN Age BETWEEN 18 AND 35 THEN '18-35'

WHEN Age BETWEEN 36 AND 50 THEN '36-50'

WHEN Age BETWEEN 51 AND 65 THEN '51-65'

ELSE '66+'

END;

-- Drop the temporary table

DROP TABLE #People;

**20)Write a query to get the current date and time in SQL Server.**

SELECT GETDATE();

**21) Write a query to find the employee who has been with the company the longest**

SELECT TOP 1 EmployeeID, Name, HireDate

FROM Employee

ORDER BY HireDate ASC;

-------------------------------------------------------------------------------------------------------------------------

22) **27. Write a query to return the difference in days between two dates.**

SELECT DATEDIFF(DAY, '2023-01-01', GETDATE()) AS DaysDifference;

23) SELECT

CASE

WHEN (A + B <= C) OR (A + C <= B) OR (B + C <= A) THEN 'Not A Triangle' -- Check for triangle inequality violation

WHEN A = B AND B = C THEN 'Equilateral' -- All sides are equal

WHEN A = B OR B = C OR A = C THEN 'Isosceles' -- Two sides are equal

ELSE 'Scalene' -- All sides are different

END AS Triangle\_Type

FROM TRIANGLES

-------------------------------------------------------------------------------------------------------------------------

**24) Write a query to delete duplicate rows from a table while keeping one**

-- Step 1: Create a table

CREATE TABLE YourTable (

id INT,

column1 VARCHAR(50),

column2 VARCHAR(50),

column3 VARCHAR(50)

);

-- Step 2: Insert sample data with duplicates

INSERT INTO YourTable (id, column1, column2, column3)

VALUES

(1, 'A', 'B', 'C'),

(2, 'A', 'B', 'C'),

(3, 'D', 'E', 'F'),

(4, 'A', 'B', 'C'),

(5, 'D', 'E', 'F');

-- Step 3: Run the DELETE query to remove duplicates

DELETE FROM YourTable

WHERE id NOT IN (

SELECT MIN(id)

FROM YourTable

GROUP BY column1, column2, column3

);

-- Step 4: Check the remaining data

SELECT \* FROM YourTable;

**25) to fetch the top 5 salaries:**

SELECT TOP 5 \*

FROM Employee

ORDER BY Salary DESC;

**26) Write a query to find employees who joined in the last 30 days.**

SELECT \*

FROM Employee

WHERE JoinDate >= DATEADD(DAY, -30, GETDATE());

27) **Write a query to find the number of employees who earn a salary between two given amounts**.

SELECT COUNT(\*)

FROM Employee

WHERE Salary BETWEEN 50000 AND 100000;

**28) Write a query to find the employees who have the same job title.**

SELECT JobTitle, COUNT(\*) AS JobTitleCount

FROM Employee

GROUP BY JobTitle

HAVING COUNT(\*) > 1;

---------------------

**29) Write a query to list the names of employees who have the same first and last name.**

SELECT FirstName, LastName

FROM Employee

WHERE FirstName = LastName;

-----------------------------------------------------------

**30) Write a query to find employees who are older than 40 but have been working for less than 5 years.**

SELECT \*

FROM Employee

WHERE DATEDIFF(YEAR, BirthDate, GETDATE()) > 40

AND DATEDIFF(YEAR, JoinDate, GETDATE()) < 5;

**31) 58. Write a query to calculate the total salary paid for each month.**

SELECT YEAR(JoinDate) AS YearHired, MONTH(JoinDate) AS MonthHired, SUM(Salary) AS TotalSalary

FROM Employee

GROUP BY YEAR(JoinDate), MONTH(JoinDate)

ORDER BY YearHired DESC, MonthHired DESC;

3**2) Write a query to find the department with the highest average salary.**

SELECT DepartmentID, AVG(Salary) AS AverageSalary

FROM Employee

GROUP BY DepartmentID

ORDER BY AverageSalary DESC

LIMIT 1;

**33) Write a query to find the employees whose salary is within the range of 30,000 and 60,000.**

SELECT \*

FROM Employee

WHERE Salary BETWEEN 30000 AND 60000;

**34) Write a query to display employees whose names start with the letter 'J' and end with the letter 'n'**.

SELECT \*

FROM Employee

WHERE Name LIKE 'J%n';

35) **Write a query to find the departments where the total salary is greater than a given amount (e.g., 500,000).**

SELECT DepartmentID

FROM Employee

GROUP BY DepartmentID

HAVING SUM(Salary) > 500000;

**36) Write a query to find the employees who have worked in more than one department.**

SELECT EmployeeID, COUNT(DISTINCT DepartmentID) AS DepartmentCount

FROM Employee

GROUP BY EmployeeID

HAVING COUNT(DISTINCT DepartmentID) > 1**;**

**37)** **Write a query to find employees who have been in the company for more than 10 years but have not been promoted.**

SELECT \*

FROM Employee

WHERE DATEDIFF(YEAR, JoinDate, GETDATE()) > 10

AND PromotionDate IS NULL;

**38) Write a query to return employees who were hired in the first quarter of 2023.**

SELECT \*

FROM Employee

WHERE JoinDate BETWEEN '2023-01-01' AND '2023-03-31';

39**) Write a query to return employees who earn more than the average salary in their department but less than the highest salary.**

SELECT E.EmployeeID, E.Name, E.Salary, E.DepartmentID

FROM Employee E

WHERE E.Salary > (SELECT AVG(Salary) FROM Employee WHERE DepartmentID = E.DepartmentID)

AND E.Salary < (SELECT MAX(Salary) FROM Employee WHERE DepartmentID = E.DepartmentID);

40) **Write a query to calculate the percentage of employees who earn above the average salary in the company.**

SELECT

(SELECT COUNT(\*) FROM Employee WHERE Salary > (SELECT AVG(Salary) FROM Employee)) \* 100.0 / COUNT(\*) AS PercentageAboveAverage

FROM Employee;

41) **Write a query to find the total salary of employees in each department and sort it in descending order.**

SELECT DepartmentID, SUM(Salary) AS TotalSalary

FROM Employee

GROUP BY DepartmentID

ORDER BY TotalSalary DESC;

42) . **Write a query to calculate the total salary of employees who joined after a specific date and belong to a specific department.**

SELECT SUM(Salary) AS TotalSalary

FROM Employee

WHERE JoinDate > '2022-01-01' AND DepartmentID = 2;

**43) Write a query to list employees who have worked in more than two departments.**

SELECT EmployeeID, COUNT(DISTINCT DepartmentID) AS DepartmentCount

FROM Employee

GROUP BY EmployeeID

HAVING COUNT(DISTINCT DepartmentID) > 2;

**44) Write a query to find employees who have a salary between the 50th and 90th percentiles.**

WITH SalaryPercentile AS (

SELECT Salary, PERCENT\_RANK() OVER (ORDER BY Salary) AS Percentile

FROM Employee

)

SELECT \*

FROM SalaryPercentile

WHERE Percentile BETWEEN 0.5 AND 0.9;

45) **Write a query to find the departments that have an average salary greater than the company’s overall average salary**.

SELECT DepartmentID, AVG(Salary) AS AverageSalary

FROM Employee

GROUP BY DepartmentID

HAVING AVG(Salary) > (SELECT AVG(Salary) FROM Employee);

**46) Write a query to find the second highest salary in the company.**

SELECT MAX(Salary) AS SecondHighestSalary

FROM Employee

WHERE Salary < (SELECT MAX(Salary) FROM Employee);

**47) Write a query to find employees who earn more than the highest salary in the 'IT' department.**

SELECT \*

FROM Employee

WHERE Salary > (SELECT MAX(Salary) FROM Employee WHERE DepartmentID = (SELECT DepartmentID FROM Department WHERE DepartmentName = 'IT'));

48) . **Write a query to find employees who have been with the company for more than 3 years and belong to the 'Sales' department.**

SELECT \*

FROM Employee

WHERE DATEDIFF(YEAR, JoinDate, GETDATE()) > 3

AND DepartmentID = (SELECT DepartmentID FROM Department WHERE DepartmentName = 'Sales');

**49) Write a query to list employees who do not have a manager assigned.**

SELECT \*

FROM Employee

WHERE ManagerID IS NULL;

50) **Write a query to find the employees who are in the same department and have the same salary.**

SELECT e1.EmployeeID, e1.EmployeeName, e1.DepartmentID, e1.Salary

FROM Employees e1

JOIN Employees e2 ON e1.DepartmentID = e2.DepartmentID

AND e1.Salary = e2.Salary

AND e1.EmployeeID <> e2.EmployeeID;

**51) Write a query to list employees who have worked in more than one department.**

SELECT EmployeeID

FROM Employee

GROUP BY EmployeeID

HAVING COUNT(DISTINCT DepartmentID) > 1;

53) **Write a query to return employees whose salary is above the average but below the highest salary in the company**

SELECT \*

FROM Employee

WHERE Salary > (SELECT AVG(Salary) FROM Employee)

AND Salary < (SELECT MAX(Salary) FROM Employee);

57)