**Windows function Class work**

CREATE TABLE employees (

emp\_no INT PRIMARY KEY AUTO\_INCREMENT,

department VARCHAR(20),

salary INT

);

INSERT INTO employees (department, salary) VALUES

('engineering', 80000),

('engineering', 69000),

('engineering', 70000),

('engineering', 103000),

('engineering', 67000),

('engineering', 89000),

('engineering', 91000),

('sales', 59000),

('sales', 70000),

('sales', 159000),

('sales', 72000),

('sales', 60000),

('sales', 61000),

('sales', 61000),

('customer service', 38000),

('customer service', 45000),

('customer service', 61000),

('customer service', 40000),

('customer service', 31000),

('customer service', 56000),

('customer service', 55000);

SELECT emp\_no, department, salary, AVG(salary) OVER() FROM employees;

SELECT

emp\_no,

department,

salary,

MIN(salary) OVER(),

MAX(salary) OVER()

FROM employees;

SELECT

emp\_no, department, salary, MIN(salary), MAX(salary)

FROM

employees;

**PARTITION BY**

SELECT

emp\_no,

department,

salary,

AVG(salary) OVER(PARTITION BY department) AS dept\_avg,

AVG(salary) OVER() AS company\_avg

FROM employees;

SELECT

emp\_no,

department,

salary,

COUNT(\*) OVER(PARTITION BY department) as dept\_count

FROM employees;

SELECT

emp\_no,

department,

salary,

SUM(salary) OVER(PARTITION BY department) AS dept\_payroll,

SUM(salary) OVER() AS total\_payroll

FROM employees;

**ORDER BY with Windows**

1. SELECT
2. emp\_no,
3. department,
4. salary,
5. SUM(salary) OVER(PARTITION BY department ORDER BY salary) AS rolling\_dept\_salary,
6. SUM(salary) OVER(PARTITION BY department) AS total\_dept\_salary
7. FROM employees;
9. SELECT
10. emp\_no,
11. department,
12. salary,
13. MIN(salary) OVER(PARTITION BY department ORDER BY salary DESC) as rolling\_min
14. FROM employees;

**Rank**

1. SELECT
2. emp\_no,
3. department,
4. salary,
5. ROW\_NUMBER() OVER(PARTITION BY department ORDER BY SALARY DESC) as dept\_row\_number,
6. RANK() OVER(PARTITION BY department ORDER BY SALARY DESC) as dept\_salary\_rank,
7. RANK() OVER(ORDER BY salary DESC) as overall\_rank,
8. DENSE\_RANK() OVER(ORDER BY salary DESC) as overall\_dense\_rank,
9. ROW\_NUMBER() OVER(ORDER BY salary DESC) as overall\_num
10. FROM employees ORDER BY overall\_rank;

**First Value CODE**

1. SELECT
2. emp\_no,
3. department,
4. salary,
5. FIRST\_VALUE(emp\_no) OVER(PARTITION BY department ORDER BY salary DESC) as highest\_paid\_dept,
6. FIRST\_VALUE(emp\_no) OVER(ORDER BY salary DESC) as highest\_paid\_overall
7. FROM employees;

**LEAD Value CODE**

1. SELECT
2. emp\_no,
3. department,
4. salary,
5. salary - LAG(salary) OVER(ORDER BY salary DESC) as salary\_diff
6. FROM employees;
8. SELECT
9. emp\_no,
10. department,
11. salary,
12. salary - LAG(salary) OVER(PARTITION BY department ORDER BY salary DESC) as dept\_salary\_diff
13. FROM employees;

**Windows function Assignments**

Query to create table

-- Create the sales table CREATE TABLE sales (

id INT PRIMARY KEY,

salesperson VARCHAR(50), region VARCHAR(50), sales\_amount DECIMAL(10, 2)

);

-- Insert sample data into the sales table

INSERT INTO sales (id, salesperson, region, sales\_amount) VALUES (1, 'Alice', 'North', 5000),

(2, 'Bob', 'South', 7000),

(3, 'Charlie', 'East', 7000),

(4, 'Dave', 'West', 6000),

(5, 'Eve', 'North', 8000),

(6, 'Frank', 'South', 5000),

(7, 'Grace', 'East', 7000),

(8, 'Heidi', 'West', 6000),

(9, 'Ivan', 'North', 8000),

(10, 'Judy', 'South', 9000);

Questions :

## 20 Questions Based on `RANK()`, `DENSE\_RANK()`, `ROW\_NUMBER()`, and

`PARTITION BY`

1. What is the difference between `RANK()`, `DENSE\_RANK()`, and `ROW\_NUMBER()` functions?
2. Write a query to assign row numbers to each row in the `sales` table.
3. Write a query to rank salespeople based on their sales amounts in descending order.
4. Write a query to dense rank salespeople based on their sales amounts in descending order.
5. Write a query to find the salesperson(s) with the highest sales amount in each region.
6. Write a query to rank salespeople based on their sales amounts within each region.
7. Write a query to dense rank salespeople based on their sales amounts within each region.
8. Write a query to assign row numbers to salespeople within each region.
9. What happens if two salespeople have the same sales amount when using the `RANK()` function?
10. What happens if two salespeople have the same sales amount when using the

`DENSE\_RANK()` function?

1. Write a query to find the top 3 salespeople with the highest sales amounts across all regions.
2. Write a query to find the top 3 salespeople with the highest sales amounts within each region.
3. Explain how the `PARTITION BY` clause works in the context of window functions.
4. Write a query to calculate the cumulative sales amount for each salesperson.
5. Write a query to calculate the cumulative sales amount for each salesperson within each region.
6. Write a query to rank salespeople based on their sales amounts, partitioned by region and ordered by sales amount.
7. Write a query to find the salesperson with the lowest sales amount in each region.
8. Write a query to rank salespeople based on their sales amounts, with ties being given the same rank.
9. Write a query to dense rank salespeople based on their sales amounts, with ties being given the same rank.
10. Write a query to assign row numbers to salespeople within each region and order by sales amount in descending order.