Advanced SQL Concepts Assignment DPP

Question 1: What is the syntax of a window function?

Question 2: What is the purpose of the FIRST_VALUE() and LAST_VALUE() functions?

Question 3: Write an SQL query to assign a unique rank to each employee based on salary (highest first) using ROW_NUMBER().

Question 4: Write a query to find each employee's department and their department-wise rank based on salary.

Question 5: What will happen if we use DENSE_RANK() instead of RANK()?

Question 6: Write a query to calculate a running total of salaries across all employees (ordered by hire_date).

Question 7: Write a query to show each employee’s salary and the difference from the highest salary in their department.

Question 8: Write a query to compute a 3-period moving average of salaries based on hire date.

Question 9: Write a query using CUME_DIST() to find the cumulative distribution of salaries.

Question 10: Write a query to retrieve the last hired employee in each department using LAST_VALUE().

Question 11: What happens when you use RANGE instead of ROWS in a window function? Provide an example query.

Question 12: Write an SQL query to list employees whose salary is above their department’s average salary. Use a subquery with a window function.

Question 13: Write a query to join the employees and departments tables

and calculate each employee’s rank within their department based on salary.

Solutions:

Creating Table employees

```
CREATE TABLE employees(
employee_id INT,
name VARCHAR(50),
department VARCHAR(50),
salary INT,
hire_date DATE
);
```

Inserting Values in Table employees

```
INSERT INTO employees VALUES
(1,'Alice','HR',55000,'2020-01-15'),
(2,'Bob','HR',80000,'2019-05-16'),
(3,'Charlie','HR',70000,'2018-07-30'),
(4,'Dev','Finance',48000,'2021-01-10'),
(5,'Imran','IT',68000,'2017-12-25'),
(6,'Jack','Finance',60000,'2019-11-05'),
(7,'Jason','IT',45000,'2018-03-15'),
(8,'Kiara','IT',70000,'2022-06-18'),
(9,'Michael','IT',42000,'2019-05-20'),
(10,'Nalini','Finance',41500,'2021-08-24'),
(11,'Nandini','Finance',52000,'2017-03-25');
```

Creating Table department

```
CREATE TABLE departments(
department_id INT,
department_name VARCHAR(50),
location VARCHAR(50)
);
```

Insert Values in Table departments

```
INSERT INTO departments VALUES
(1,'HR','New York'),
(2,'IT','San Francisco'),
(3,'Finance','Chicago');
```

Question 1:

What is the syntax of a window function?

Solution:

```
SELECT column, FUNCTION() OVER(PARTITION BY col ORDER BY col) FROM table;
```

Question 2:
Purpose of FIRST_VALUE() and LAST_VALUE()?

Solution:

FIRST_VALUE returns first row value. LAST_VALUE returns last row value in window.

Question 3:
Unique rank using ROW_NUMBER()

Solution:

```
SELECT *, ROW_NUMBER() OVER(ORDER BY salary DESC) FROM employees;
```

Question 4:
Department wise rank

Solution:

```
SELECT *, RANK() OVER(PARTITION BY department ORDER BY salary DESC) FROM
employees;
```

Question 5:
DENSE_RANK vs RANK

Solution:

DENSE_RANK removes gaps between ranks.

Question 6:
Running total

Solution:

```
SELECT *, SUM(salary) OVER(ORDER BY hire_date ROWS BETWEEN UNBOUNDED
PRECEDING AND CURRENT ROW) FROM employees;
```

Question 7:
Salary diff from max

Solution:

```
SELECT *, MAX(salary) OVER(PARTITION BY department)-salary FROM employees;
```

Question 8:
3 period moving avg

Solution:

```
SELECT *, AVG(salary) OVER(ORDER BY hire_date ROWS 2 PRECEDING) FROM
employees;
```

Question 9:
CUME_DIST

Solution:

```
SELECT *, CUME_DIST() OVER(ORDER BY salary) FROM employees;
```

Question 10:
Last hired per dept

Solution:

```
SELECT DISTINCT department, LAST_VALUE(name) OVER(PARTITION BY
department ORDER BY hire_date RANGE BETWEEN UNBOUNDED PRECEDING AND
UNBOUNDED FOLLOWING) FROM employees;
```

Question 11:
RANGE vs ROWS

Solution:

RANGE groups equal values while ROWS uses physical rows.

Question 12:
Salary above dept avg

Solution:

```
SELECT * FROM (SELECT *, AVG(salary) OVER(PARTITION BY department) avg_sal
FROM employees)t WHERE salary>avg_sal;
```

Question 13:
Join + rank

Solution:

```
SELECT e.*, d.location, RANK() OVER(PARTITION BY e.department ORDER BY e.salary
DESC) FROM employees e JOIN departments d ON e.department=d.department_name;
```