Ex.No: 10 AGGREGATING DATA USING GROUP FUNCTION 231901064

DATE:26.10.2024 THENMOZHI

1. **Group functions work across many rows to produce one result per group. True**

*Explanation:* Group functions, like SUM, AVG, COUNT, MAX, and MIN, aggregate data across multiple rows to produce a single result per group defined by the GROUP BY clause.

# Group functions include nulls in calculations. False

*Explanation:* Most group functions ignore NULL values in their calculations (e.g., SUM and AVG skip NULLs). However, COUNT(\*) will count all rows, including those with NULL values.

# The WHERE clause restricts rows prior to inclusion in a group calculation. True

*Explanation:* The WHERE clause filters rows before grouping occurs. Only the rows that meet the WHERE condition are included in the group calculation.

The HR department needs the following reports:

1. Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number

## SELECT

ROUND(MAX(salary)) AS Maximum, ROUND(MIN(salary)) AS Minimum, ROUND(SUM(salary)) AS Sum, ROUND(AVG(salary)) AS Average

FROM employees;



1. Modify the above query to display the minimum, maximum, sum, and average salary for each job type.

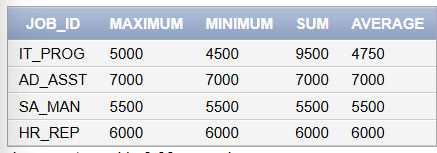
## SELECT

job\_id,

ROUND(MAX(salary)) AS Maximum,

ROUND(MIN(salary)) AS Minimum, ROUND(SUM(salary)) AS Sum, ROUND(AVG(salary)) AS Average

FROM employees GROUP BY job\_id;



1. Write a query to display the number of people with the same job. Generalize the query so that the user in the HR department is prompted for a job title.

## SELECT

COUNT(\*) AS Number\_of\_People FROM employees

WHERE job\_id = :job\_title;





1. Determine the number of managers without listing them. Label the column Number of Managers. Hint: Use the MANAGER\_ID column to determine the number of Managers.

## SELECT

COUNT(DISTINCT manager\_id) AS "Number of Managers" FROM employees

WHERE manager\_id IS NOT NULL;



1. Find the difference between the highest and lowest salaries. Label the column DIFFERENCE.

## SELECT

MAX(salary) - MIN(salary) AS DIFFERENCE FROM employees;



1. Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.

## SELECT

manager\_id,

MIN(salary) AS Lowest\_Salary FROM employees

WHERE manager\_id IS NOT NULL GROUP BY manager\_id

HAVING MIN(salary) > 6000

ORDER BY Lowest\_Salary DESC;



1. Create a query to display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings.

## SELECT

COUNT(\*) AS Total\_Employees,

SUM(CASE WHEN EXTRACT(YEAR FROM hire\_date) = 1995 THEN 1 ELSE 0 END) AS

Hired\_in\_1995,

SUM(CASE WHEN EXTRACT(YEAR FROM hire\_date) = 1996 THEN 1 ELSE 0 END) AS

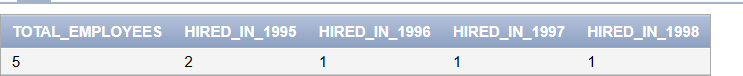
Hired\_in\_1996,

SUM(CASE WHEN EXTRACT(YEAR FROM hire\_date) = 1997 THEN 1 ELSE 0 END) AS

Hired\_in\_1997,

SUM(CASE WHEN EXTRACT(YEAR FROM hire\_date) = 1998 THEN 1 ELSE 0 END) AS

Hired\_in\_1998 FROM employees;



1. Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.

## SELECT

job\_id,

SUM(CASE WHEN department\_id = 20 THEN salary ELSE 0 END) AS Dept\_20\_Salary, SUM(CASE WHEN department\_id = 50 THEN salary ELSE 0 END) AS Dept\_50\_Salary, SUM(CASE WHEN department\_id = 80 THEN salary ELSE 0 END) AS Dept\_80\_Salary, SUM(CASE WHEN department\_id = 90 THEN salary ELSE 0 END) AS Dept\_90\_Salary, SUM(salary) AS Total\_Salary

FROM employees

WHERE department\_id IN (20, 50, 80, 90) GROUP BY job\_id;



1. Write a query to display each department‘s name, location, number of employees, and the average salary for all the employees in that department. Label the column name-Location, Number of people, and salary respectively. Round the average salary to two decimal places.

## SELECT

d.department\_name || '-' || l.location\_id AS "name-Location", COUNT(e.employee\_id) AS "Number of people", ROUND(AVG(e.salary), 2) AS "salary"

FROM departments d

JOIN employees e ON d.department\_id = e.department\_id JOIN locations l ON d.location\_id = l.location\_id

GROUP BY d.department\_name, l.location\_id;

