

Ex.No.: 10		AGGREGATING DATA USING GROUP FUNCTIONS
Date:	20/9/24	

Objectives

After the completion of this exercise, the students be will be able to do the following:

- Identify the available group functions
- Describe the use of group functions
- Group data by using the GROUP BY clause
- Include or exclude grouped rows by using the HAVING clause

What Are Group Functions?

Group functions operate on sets of rows to give one result per group

Types of Group Functions

- AVG
- COUNT
- MAX
- MIN
- STDDEV
- SUM
- VARIANCE

Each of the functions accepts an argument. The following table identifies the options that you can use in the syntax:

Function	Description
AVG ([DISTINCT ALL] n)	Average value of n, ignoring null values
COUNT ((* [DISTINCT ALL] expr))	Number of rows, where expr evaluates to something other than null (count all selected rows using *, including duplicates and rows with nulls)
MAX ([DISTINCT ALL] expr)	Maximum value of expr, ignoring null values
MIN ([DISTINCT ALL] expr)	Minimum value of expr, ignoring null values
STDDEV ([DISTINCT ALL] x)	Standard deviation of n, ignoring null values
SUM ([DISTINCT ALL] n)	Sum values of n, ignoring null values
VARIANCE ([DISTINCT ALL] x)	Variance of n, ignoring null values

Group Functions: Syntax

```
SELECT [column,] group_function(column), ...
FROM table
[WHERE condition]
```


Group functions can be nested to a depth of two. The slide example displays the maximum average salary.

```
SELECT MAX(AVG(salary)) FROM employees GROUP BY department_id;
```

Summary

In this exercise, students should have learned how to:

- Use the group functions COUNT, MAX, MIN, and AVG
- Write queries that use the GROUP BY clause
- Write queries that use the HAVING clause

```
SELECT column, group_function  
FROM table  
[WHERE condition]  
[GROUP BY group_by_expression]  
[HAVING group_condition]  
[ORDER BY column];
```

Find the Solution for the following:

Determine the validity of the following three statements. Circle either True or False.

1. Group functions work across many rows to produce one result per group.
True/False
2. Group functions include nulls in calculations.
True/False
3. The WHERE clause restricts rows prior to inclusion in a group calculation.
True/False

The HR department needs the following reports:

4. 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),0) as Maximum, round(min(salary),0) as Minimum
sum(salary) as sum, round(avg(salary),0) as salary from employee;*

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

*select round(max(salary),0) as Maximum, round(min(salary),0) as Minimum
sum(salary) as sum, round(avg(salary),0) as average from employee
group by jobtype;*

6. 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 job_id, count(*) as total_employee from employee
group by job_id order by total_employee desc;
```

7. 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 manager_id, count(*) as number_of_managers from employee
group by manager_id;
```

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

```
select max(salary) - min(salary) as difference
from employees;
```

9. 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) from employee where manager_id
is not null group by manager_id having min(salary) > 6000
order by min(salary) desc;
```

10. 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_employee,
count(if(entry_year = 1995, 1, NULL)) as hired_in_1995,
```


11. select jobtype as job

sum (case when department_id = 20 then salary else 0 end) as salary_dept_20,
sum (case when department_id = 50 then salary else 0 end) as salary_dept_50,
sum (case when department_id = 80 then salary else 0 end) as salary_dept_80,
sum (case when department_id = 90 then salary else 0 end) as salary_dept_90,
sum (salary) as total_salary

from employees

where department_id in (20, 50, 80, 90) group by jobtype;

count (if (entry-year = 1996, 1, NULL)) as hired_in_1996
 count (if (entry-year = 1997, 1, NULL)) as hired_in_1997
 count (if (entry-year = 1998, 1, NULL)) as hired_in_1998
 from employees.

11. 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.

12. 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 department_name as Department, location as location,
 count (employee-id) as 'Number of people',
 round (Avg (salary), 2) as 'salary';
 from employees
 group by department_name, location;

Evaluation Procedure	Marks awarded
Query(5)	5
Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	