Government College of Engineering, Jalgaon		
(An Autonomous Institute of Government of Maharashtra)		
Name :	**Semester :** V	**PRN :**
Class : T. Y. B.Tech Computer	**Academic Year :** 2024-25	**Subject :** CO307U
Course Teacher : Mr. Vinit Kakde	**Batch :**	
Date of Performance : / / 2024	**Date of Completion :** / / 2024	

Practical - 04

Aim:

To perform various SQL queries on an Employee table, including retrieving data, calculating aggregates, and filtering based on conditions.

Employee Table Structure:

EMPNO	EMP_NAME	DEPT	SALARY	DOJ	BRANCH
E101	Amit	Production	45000	12-Mar-00	Bangalore
E102	Amit	HR	70000	03-Jul-02	Bangalore
E103	Sunita	Management	120000	11-Jan-01	Mysore
E105	Sunita	IT	67000	01-Aug-01	Mysore
E106	Mahesh	Civil	145000	20-Sep-03	Mumbai

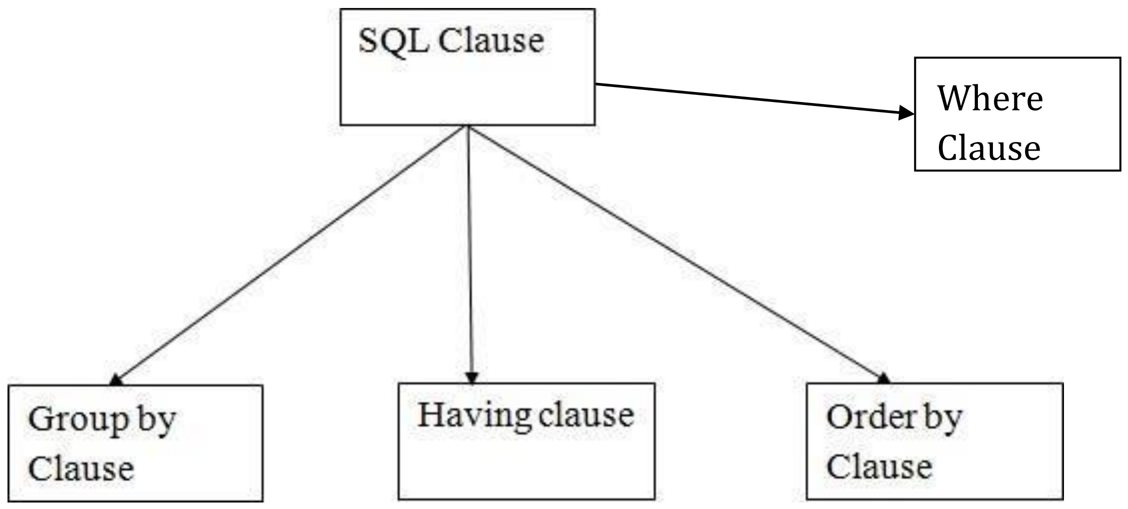
Perform the following

1. Display all the fields of employee table
2. Retrieve employee number and their salary
3. Retrieve average salary of all employee.
4. Retrieve number of employee
5. Retrieve distinct number of employee
6. Retrieve total salary of employee group by employee name and count similar names
7. Retrieve total salary of employee which is greater than >120000
8. Display name of employee in descending order
9. Display details of employee whose name is AMIT and salary greater than 50000;

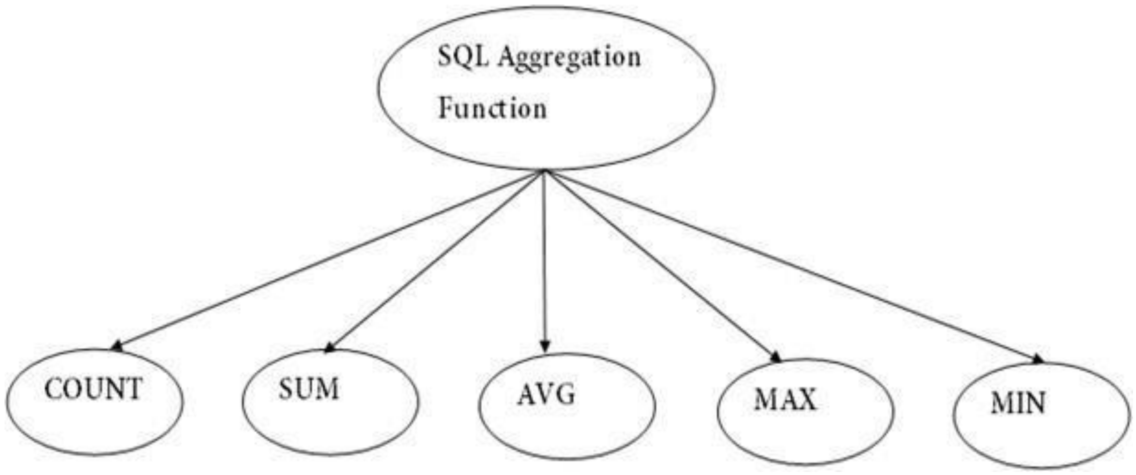
Theory:

In this experiment, we will use various SQL SELECT queries to retrieve data from an Employee table. We will use functions such as aggregate functions (AVG, COUNT, SUM), filtering using WHERE clause, and sorting the results using ORDER BY clause.

SQL Concepts Involved:



1. **SELECT** statement: Used to retrieve data from the database.
2. **WHERE clause**: Used to filter records based on specific conditions.
3. **GROUP BY clause**: Used to group rows that have the same values into summary rows.
4. **HAVING clause**: Used to filter groups based on aggregate functions.
5. **ORDER BY clause**: Used to sort the result-set in ascending or descending order.
6. **Aggregate functions**: COUNT, AVG, SUM, MAX, and MIN are used to perform calculations on multiple values from the table. These functions are explained below:



1. Count Function

COUNT function is used to Count the number of rows in a database table. It can work on both numeric and non-numeric data types.

COUNT function uses the COUNT(*) that returns the count of all the rows in a specified table. COUNT(*) considers duplicate and Null.

Syntax

COUNT(*)

or

COUNT([ALL|DISTINCT] expression)

Sample table:

PRODUCT_MAST

PRODUCT	COMPANY	QTY	RATE	COST
Item1	Com1	2	10	20
Item2	Com2	3	25	75
Item3	Com1	2	30	60
Item4	Com3	5	10	50
Item5	Com2	2	20	40
Item6	Cpm1	3	25	75
Item7	Com1	5	30	150
Item8	Com1	3	10	30
Item9	Com2	2	25	50
Item10	Com3	4	30	120

Example: COUNT()

SELECT COUNT(*) FROM

PRODUCT_MAST;

Output:

10

2. SUM Function

Sum function is used to calculate the sum of all selected columns. It works on numeric fields only.

Syntax

SUM()

or

SUM([ALL|DISTINCT] expression)

Example: SUM()

SELECT SUM(COST) FROM

PRODUCT_MAST;

Output:

670

3. AVG function

The AVG function is used to calculate the average value of the numeric type. AVG function returns the average of all non-Null values.

Syntax

AVG()

or

AVG([ALL|DISTINCT] expression)

Example:

SELECT AVG(COST) FROM

PRODUCT_MAST;

Output:

67.00

4. MAX Function

MAX function is used to find the maximum value of a certain column. This function determines the largest value of all selected values of a column.

Syntax

MAX()

or
MAX([ALL|DISTINCT] expression)

Example:

SELECT MAX(RATE)

FROM PRODUCT_MAST;

30

5. MIN Function

MIN function is used to find the minimum value of a certain column. This function determines the smallest value of all selected values of a column.

Syntax

MIN()

or

MIN([ALL|DISTINCT] expression)

Example:

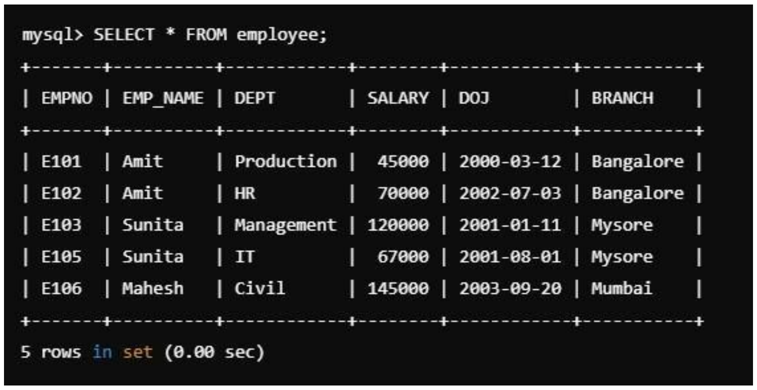
SELECT MIN(RATE) FROM

PRODUCT_MAST;

Output:

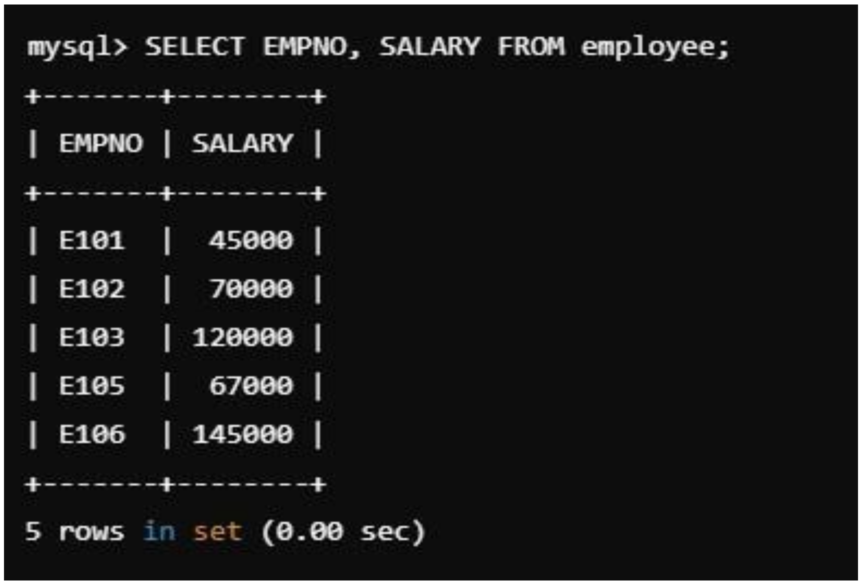
10

Program and Output:
1. Display all the fields of the employee table:
SELECT * FROM employee;



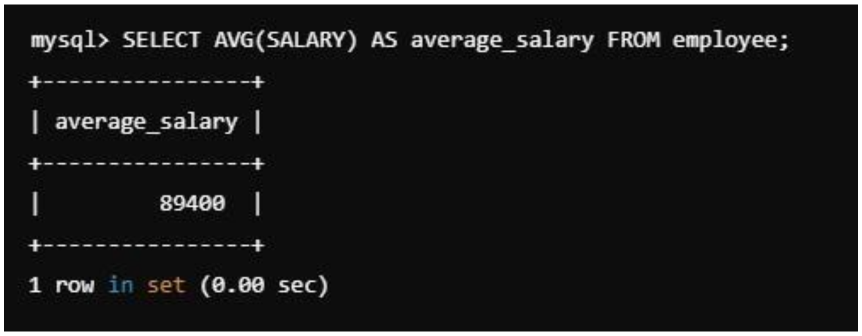
```
mysql> SELECT * FROM employee;
+-------+----------+------------+--------+------------+-----------+
| EMPNO | EMP_NAME | DEPT       | SALARY | DOJ        | BRANCH    |
+-------+----------+------------+--------+------------+-----------+
| E101  | Amit     | Production |  45000 | 2000-03-12 | Bangalore |
| E102  | Amit     | HR         |  70000 | 2002-07-03 | Bangalore |
| E103  | Sunita   | Management | 120000 | 2001-01-11 | Mysore    |
| E105  | Sunita   | IT         |  67000 | 2001-08-01 | Mysore    |
| E106  | Mahesh   | Civil      | 145000 | 2003-09-20 | Mumbai    |
+-------+----------+------------+--------+------------+-----------+
5 rows in set (0.00 sec)
```

2. Retrieve employee number and their salary:
SELECT EMPNO, SALARY FROM employee;



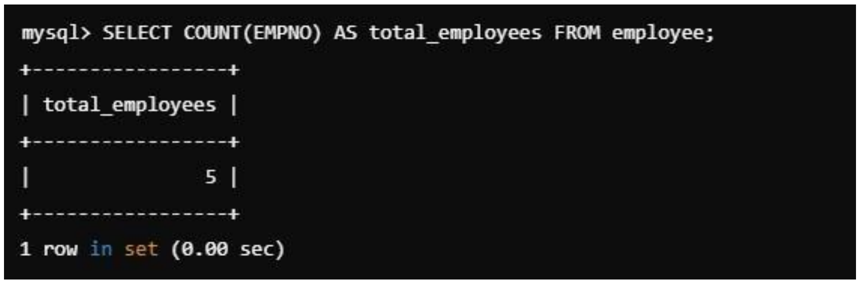
```
mysql> SELECT EMPNO, SALARY FROM employee;
+-------+--------+
| EMPNO | SALARY |
+-------+--------+
| E101  |  45000 |
| E102  |  70000 |
| E103  | 120000 |
| E105  |  67000 |
| E106  | 145000 |
+-------+--------+
5 rows in set (0.00 sec)
```

3. Retrieve the average salary of all employees:
SELECT AVG(SALARY) FROM employee;



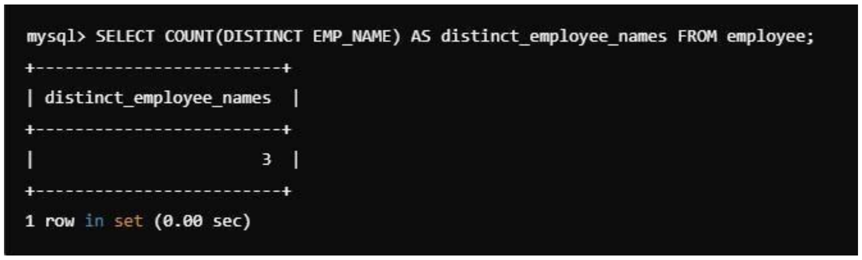
```
mysql> SELECT AVG(SALARY) AS average_salary FROM employee;
+----------------+
| average_salary |
+----------------+
|        89400   |
+----------------+
1 row in set (0.00 sec)
```

4. Retrieve the number of employees:
SELECT COUNT(*) FROM employee;



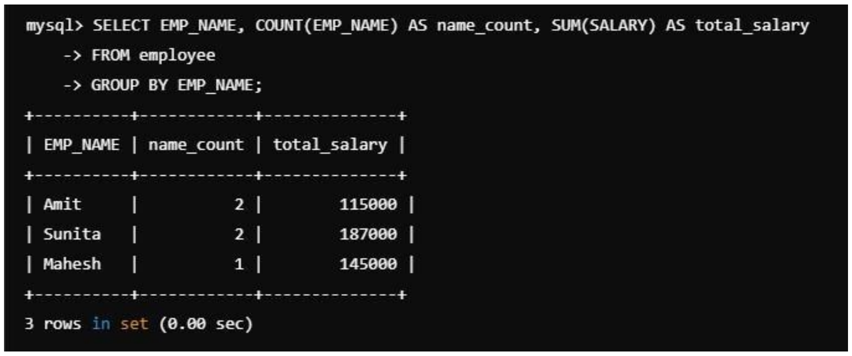
```
mysql> SELECT COUNT(EMPNO) AS total_employees FROM employee;
+-----------------+
| total_employees |
+-----------------+
|               5 |
+-----------------+
1 row in set (0.00 sec)
```

5. Retrieve distinct number of employees:
SELECT COUNT(DISTINCT EMP_NAME) FROM employee;



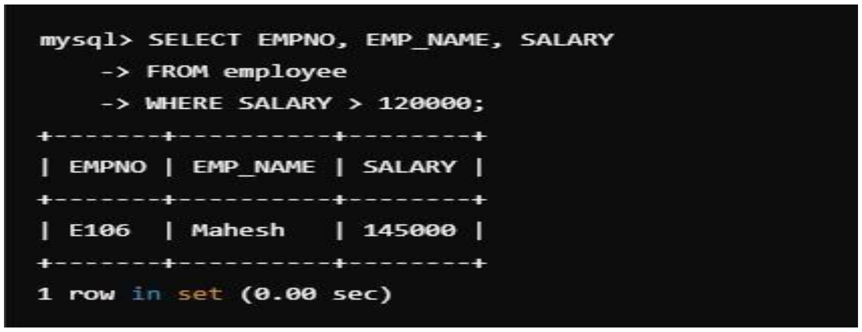
```
mysql> SELECT COUNT(DISTINCT EMP_NAME) AS distinct_employee_names FROM employee;
+------------------------+
| distinct_employee_names  |
+------------------------+
|                      3  |
+------------------------+
1 row in set (0.00 sec)
```

6. Retrieve total salary of employees grouped by name and count similar names:
SELECT EMP_NAME, SUM(SALARY), COUNT(EMP_NAME) FROM employee GROUP BY EMP_NAME;



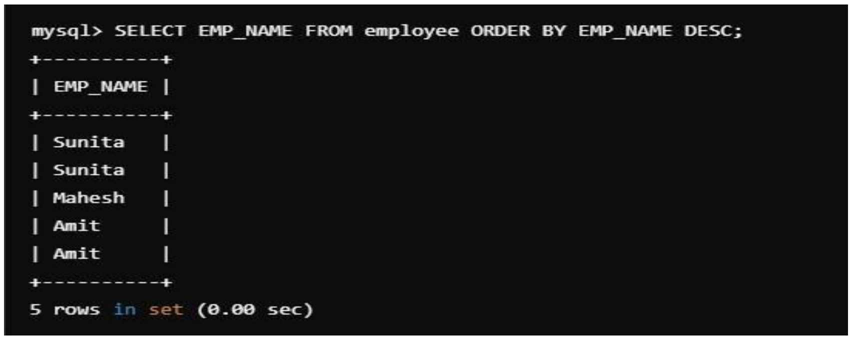
```
mysql> SELECT EMP_NAME, COUNT(EMP_NAME) AS name_count, SUM(SALARY) AS total_salary
    -> FROM employee
    -> GROUP BY EMP_NAME;
+----------+------------+--------------+
| EMP_NAME | name_count | total_salary |
+----------+------------+--------------+
| Amit     |          2 |       115000 |
| Sunita   |          2 |       187000 |
| Mahesh   |          1 |       145000 |
+----------+------------+--------------+
3 rows in set (0.00 sec)
```

7. Retrieve total salary of employees with salary greater than 120000:
SELECT SUM(SALARY) FROM employee WHERE SALARY > 120000;



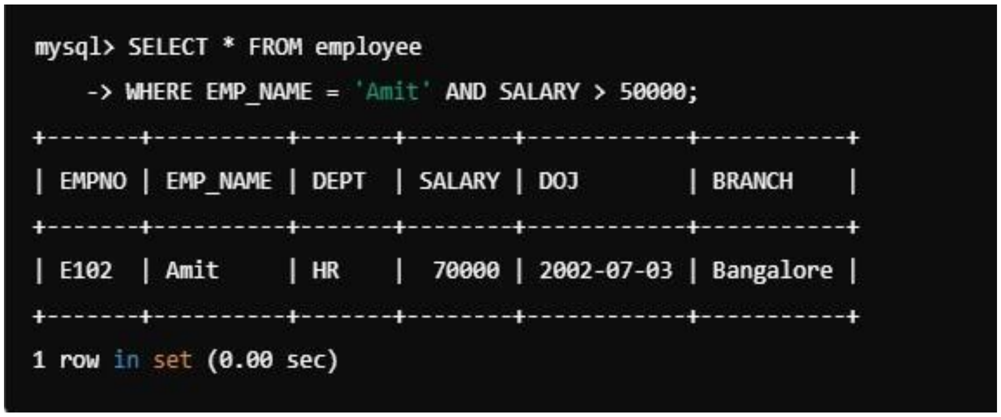
```
mysql> SELECT EMPNO, EMP_NAME, SALARY
    -> FROM employee
    -> WHERE SALARY > 120000;
+-------+----------+--------+
| EMPNO | EMP_NAME | SALARY |
+-------+----------+--------+
| E106  | Mahesh   | 145000 |
+-------+----------+--------+
1 row in set (0.00 sec)
```

8. Display names of employees in descending order:
SELECT EMP_NAME FROM employee ORDER BY EMP_NAME DESC;



```
mysql> SELECT EMP_NAME FROM employee ORDER BY EMP_NAME DESC;
+----------+
| EMP_NAME |
+----------+
| Sunita   |
| Sunita   |
| Mahesh   |
| Amit     |
| Amit     |
+----------+
5 rows in set (0.00 sec)
```

9. Display details of employees whose name is AMIT and salary greater than 50000:
SELECT * FROM employee WHERE EMP_NAME = 'Amit' AND SALARY > 50000;



```
mysql> SELECT * FROM employee
    -> WHERE EMP_NAME = 'Amit' AND SALARY > 50000;
+-------+----------+-------+--------+------------+-----------+
| EMPNO | EMP_NAME | DEPT  | SALARY | DOJ        | BRANCH    |
+-------+----------+-------+--------+------------+-----------+
| E102  | Amit     | HR    |  70000 | 2002-07-03 | Bangalore |
+-------+----------+-------+--------+------------+-----------+
1 row in set (0.00 sec)
```

Conclusion:
In this practical, we have successfully performed SQL queries to retrieve data, calculate aggregates, and filter records from the Employee table. We have used various SQL clauses and functions to achieve the desired results.

Questions and Answers

Question 1 :What is the WHERE clause in SQL, and how is it used?
Answer:
The WHERE clause is used to filter records in a SQL query based on a specified condition. It is typically used with SELECT, UPDATE, and DELETE statements to restrict the rows affected by the query. It can include conditions based on column values, comparisons, logical operators (AND, OR), and arithmetic expressions.

Question 2 : What is the difference between the WHERE clause and the HAVING clause in SQL?
Answer:

- The WHERE clause filters records before any grouping or aggregation takes place. It cannot be used with aggregate functions.
- The HAVING clause filters records after the GROUP BY operation and is specifically used to filter the results of aggregate functions, such as SUM(), COUNT(), or AVG().

Question 3 : What is the GROUP BY clause in SQL, and what is its purpose? Answer:
The GROUP BY clause is used to group rows that have the same values in specified columns into aggregated data sets. It is often used in conjunction with aggregate functions like SUM(), COUNT(), AVG(), MAX(), or MIN(). It allows you to perform calculations on groups of rows rather than on individual rows.

Question 4: Explain the use of aggregate functions in SQL. Provide examples for SUM(), AVG(), and COUNT().
Answer:
Aggregate functions in SQL perform a calculation on a set of values and return a single value. They are used to summarize data over a group of records.

- **SUM()**: Adds up all the values in a column. Example:
  SELECT SUM(SALARY) FROM employee; This
  calculates the total sum of all salaries.

- **AVG()**: Calculates the average of values in a column. Example:
  SELECT AVG(SALARY) FROM employee;
  This calculates the average salary of all employees.

- **COUNT()**: Returns the number of rows that match a specified condition or total rows if no condition is specified. Example:
  SELECT COUNT(*) FROM employee;
  This counts the total number of employee records.

Question 5: What is the purpose of the ORDER BY clause in SQL? How can it be used to sort data?
Answer:
The ORDER BY clause is used to sort the result set of a query in either ascending (ASC) or descending (DESC) order based on one or more columns. By default, it sorts data in ascending order.

Course Teacher's Signature
Mr.Vinit Kakde