SQL Worksheet – Employee Table and Aggregate Queries

Aim of the Session

The aim of this practical session is to create an EMPLOYEE table in SQL, insert records into it, and perform various SQL queries using GROUP BY, HAVING, and ORDER BY clauses to analyze employee data.

Software Requirements

MySQL Workbench, SQL Server.

Objective of the Session

• To understand how to create and delete database tables.
• To learn how to insert records into a table.
• To apply aggregate functions such as AVG().
• To use GROUP BY and HAVING clauses effectively.
• To retrieve and analyze data based on conditions.

Practical / Experiment Steps

1. Create the EMPLOYEE table with appropriate columns.
2. Insert multiple employee records into the table.
3. Display all records from the EMPLOYEE table.
4. Calculate average salary department-wise.
5. Retrieve employees having salary greater than 20000.
6. Display departments having average salary greater than 30000.
7. Sort departments based on average salary in descending order.

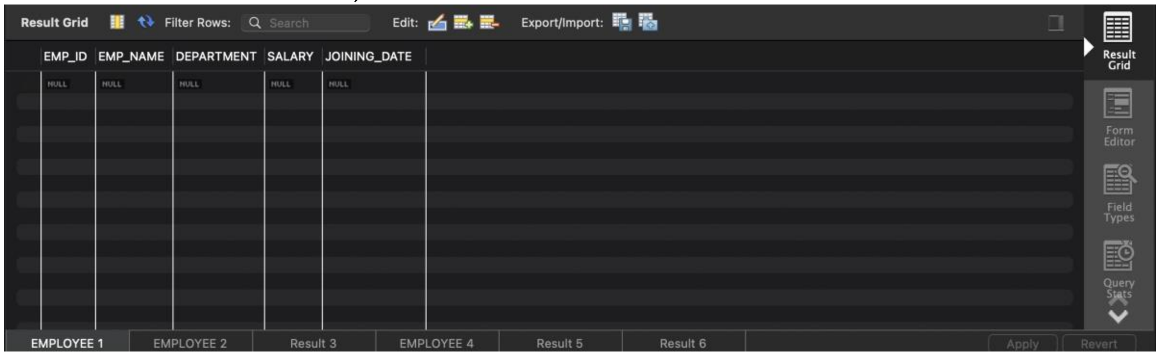
Procedure of the Practical

(i) Start the system and log in to the computer.
(ii) Open the SQL database software (PostgreSQL / MySQL / Oracle).
(iii) Create or select the required database.
(iv) Write the SQL command to drop the EMPLOYEE table if it exists.
(v) Create the EMPLOYEE table with required fields.
(vi) Insert the employee records into the table.
(vii) Execute SELECT queries to retrieve and analyze data.
(viii) Verify the output after each execution.
(ix) Save the work and record the results.

SQL Code Used

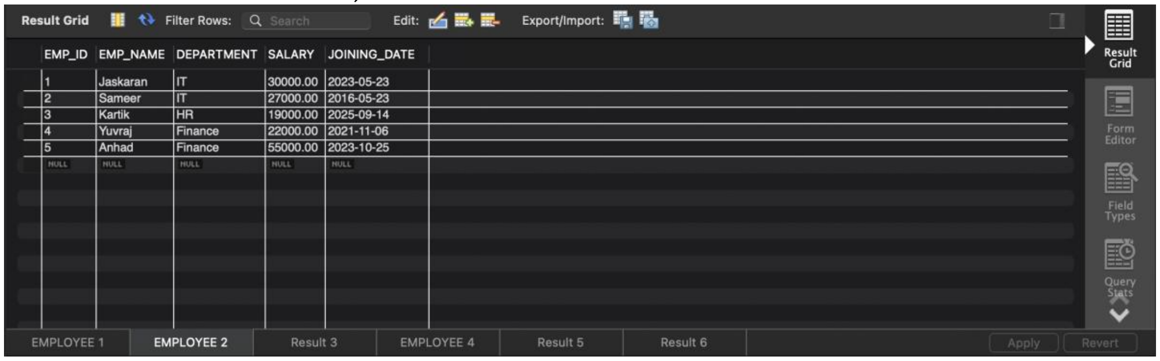
```
DROP TABLE IF EXISTS EMPLOYEE;
CREATE TABLE EMPLOYEE (
EMP_ID INT PRIMARY KEY,
EMP_NAME VARCHAR(20),
DEPARTMENT VARCHAR(20),
 SALARY DECIMAL(10,2),
 JOINING_DATE DATE
);

SELECT * FROM EMPLOYEE;
```



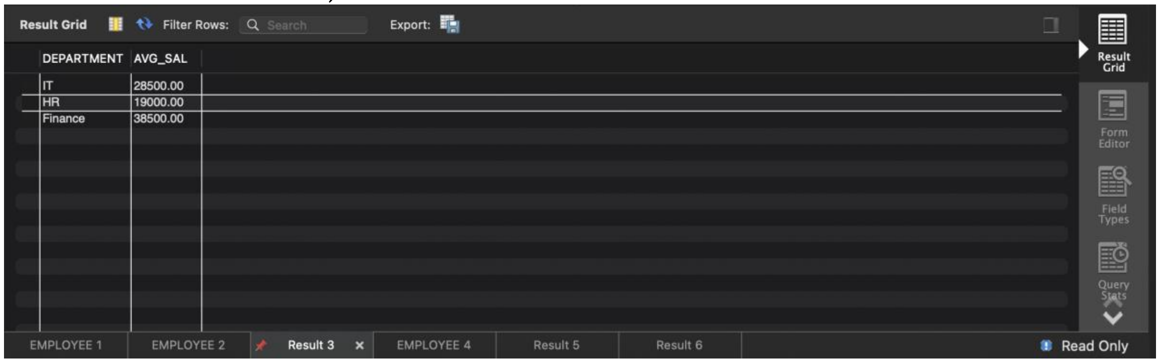
EMP_ID	EMP_NAME	DEPARTMENT	SALARY	JOINING_DATE
NULL	NULL	NULL	NULL	NULL

```
INSERT INTO EMPLOYEE VALUES (1, 'Jaskaran', 'IT', 30000, '2023-05-23');
INSERT INTO EMPLOYEE VALUES (2, 'Sameer', 'IT', 27000, '2016-05-23');
INSERT INTO EMPLOYEE VALUES (3, 'Kartik', 'HR', 19000, '2025-09-14');
INSERT INTO EMPLOYEE VALUES (4, 'Yuvraj', 'Finance', 22000, '2021-11-06');
INSERT INTO EMPLOYEE VALUES (5, 'Anhad', 'Finance', 55000, '2023-10-25');
SELECT * FROM EMPLOYEE;
```



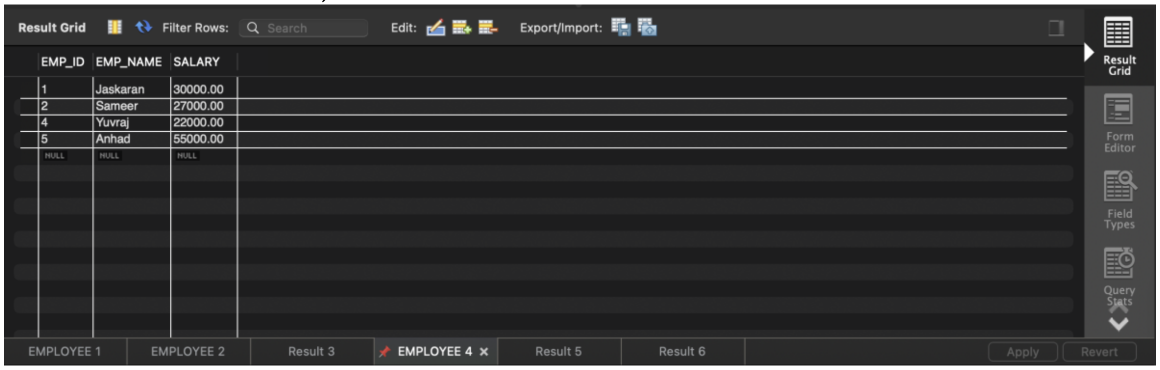
EMP_ID	EMP_NAME	DEPARTMENT	SALARY	JOINING_DATE
1	Jaskaran	IT	30000.00	2023-05-23
2	Sameer	IT	27000.00	2016-05-23
3	Kartik	HR	19000.00	2025-09-14
4	Yuvraj	Finance	22000.00	2021-11-06
5	Anhad	Finance	55000.00	2023-10-25
NULL	NULL	NULL	NULL	NULL

```
SELECT
  DEPARTMENT,
  ROUND(AVG(SALARY), 2) AS AVG_SAL
FROM EMPLOYEE
GROUP BY DEPARTMENT;
```



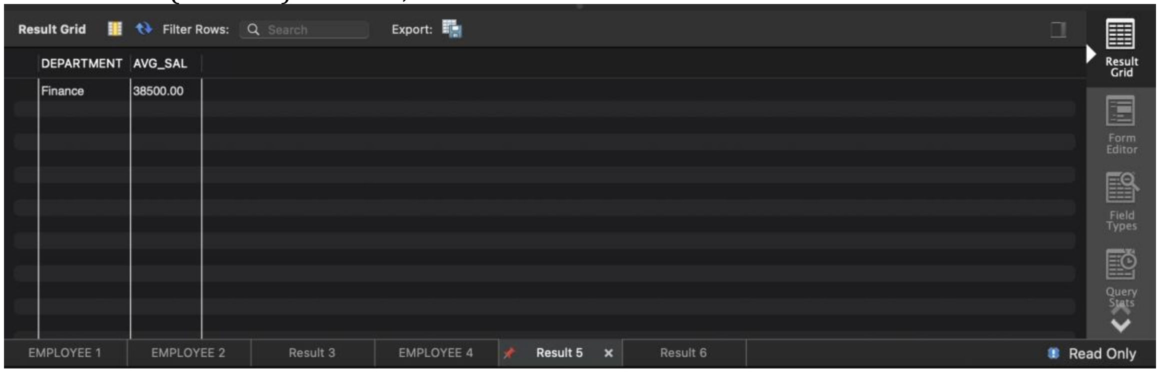
DEPARTMENT	AVG_SAL
IT	28500.00
HR	19000.00
Finance	38500.00

```
SELECT
  EMP_ID,
  EMP_NAME,
SALARY FROM EMPLOYEE
WHERE SALARY > 20000;
```



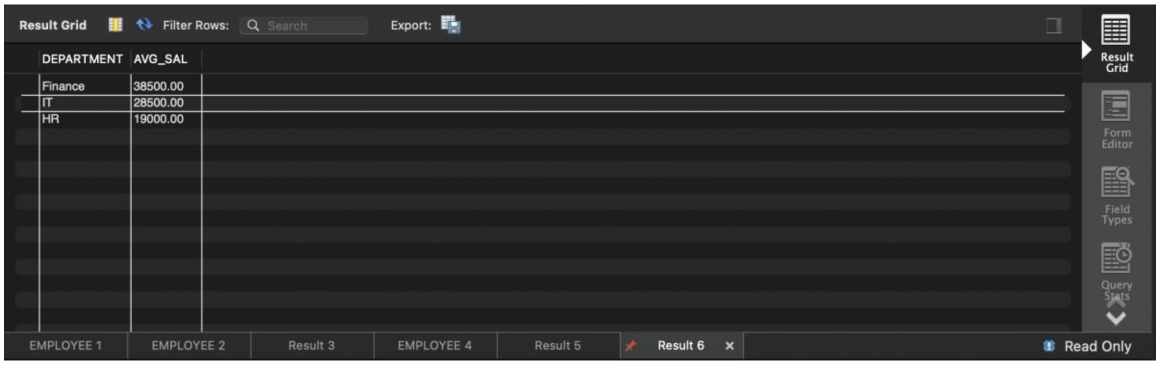
EMP_ID	EMP_NAME	SALARY
1	Jaskaran	30000.00
2	Sameer	27000.00
4	Yuvraj	22000.00
5	Anhad	55000.00
NULL	NULL	NULL

```
SELECT
  DEPARTMENT,
  ROUND(AVG(SALARY), 2) AS AVG_SAL
FROM EMPLOYEE
GROUP BY DEPARTMENT
HAVING AVG(SALARY) > 30000;
```



DEPARTMENT	AVG_SAL
Finance	38500.00

```
SELECT
  DEPARTMENT,
  ROUND(AVG(SALARY), 2) AS AVG_SAL
FROM EMPLOYEE
GROUP BY DEPARTMENT
ORDER BY AVG(SALARY) DESC;
```



```
DEPARTMENT  AVG_SAL
Finance     38500.00
IT          28500.00
HR          19000.00
```

Input / Output Analysis

Input:

• SQL commands for table creation, insertion, and data retrieval.

Output:

• EMPLOYEE table created successfully.
• Records inserted into the table.
• Department-wise average salary displayed.
• Employees with salary greater than 20000 retrieved.
• Departments filtered and sorted based on average salary.

Learning Outcome

After completing this practical, the learner is able to:

• Understand table creation and data insertion in SQL.
• Use aggregate functions for data analysis.
• Apply GROUP BY and HAVING clauses correctly.
• Analyze and sort database records effectively.
• Gain practical exposure to SQL query execution.