

Name: Ritu Choudhary

Branch: Mca(Ai&MI)-Sem2

Section: B

Batch: B3

Roll no : 51

1. Find maximum, minimum, total deposit.

```
SELECT
    MAX(Amount) AS max_deposit,
    MIN(Amount) AS min_deposit,
    SUM(Amount) AS total_deposit
FROM DEPOSIT;
```

MAX_DEPOSIT	MIN_DEPOSIT	TOTAL_DEPOSIT
7000	1000	28700

1 rows returned in 0.02 seconds [Download](#)

2. Find total number of customers.

```
SELECT COUNT(DISTINCT Cname) AS total_customers
FROM CUSTOMERS;
```

Results	Explain	Describe	Saved SQL	History
TOTAL_CUSTOMERS				
6				

1 rows returned in 0.03 seconds [Download](#)

3. List total loan from STATE BANK OF INDIA

```
SELECT SUM(Amount) AS total_loan
FROM BORROW
WHERE Bname = 'STATE BANK OF INDIA';
```

Results	Explain	Describe	Saved SQL	History
TOTAL_LOAN				
50000				
1 rows returned in 0.02 seconds Download				

4. Give maximum loan from branch UNION.
- ```
SELECT MAX(Amount) AS max_loan
FROM BORROW
WHERE Bname = 'UNION';
```

| Results                                                  | Explain | Describe | Saved SQL | History |
|----------------------------------------------------------|---------|----------|-----------|---------|
| MAX_LOAN                                                 |         |          |           |         |
| 18000                                                    |         |          |           |         |
| 1 rows returned in 0.02 seconds <a href="#">Download</a> |         |          |           |         |

5. Count total number of customer's cities.
- ```
SELECT COUNT(DISTINCT City) AS total_customer_cities
FROM CUSTOMERS;
```

Results	Explain	Describe	Saved SQL	History
TOTAL_CUSTOMER_CITIES				
4				
1 rows returned in 0.01 seconds Download				

6. List total deposit of customer having account date after 1-jan-96.
- ```
SELECT SUM(Amount) AS total_deposit
FROM DEPOSIT
WHERE Adate > TO_DATE('1996-01-01', 'YYYY-MM-DD');
```

| Results                                                  | Explain | Describe | Saved SQL | History |
|----------------------------------------------------------|---------|----------|-----------|---------|
| TOTAL_DEPOSIT                                            |         |          |           |         |
| 20000                                                    |         |          |           |         |
| 1 rows returned in 0.02 seconds <a href="#">Download</a> |         |          |           |         |

7. List total deposit of customers living in city Nagpur.
- ```

SELECT SUM(D.Amount) AS total_deposit
FROM DEPOSIT D
JOIN CUSTOMERS C ON D.Cname = C.Cname
WHERE C.City = 'Nagpur';

```

Results	Explain	Describe	Saved SQL	History
TOTAL_DEPOSIT				
5000				
1 rows returned in 0.03 seconds Download				

8. List maximum deposit of customers living in bombay.
- ```

SELECT MAX(D.Amount) AS max_deposit
FROM DEPOSIT D
JOIN CUSTOMERS C ON D.Cname = C.Cname
WHERE C.City = 'Bombay';

```

| Results                                                  | Explain | Describe | Saved SQL | History |
|----------------------------------------------------------|---------|----------|-----------|---------|
| MAX_DEPOSIT                                              |         |          |           |         |
| 7000                                                     |         |          |           |         |
| 1 rows returned in 0.03 seconds <a href="#">Download</a> |         |          |           |         |

Order by, group by and having clause in a database.

1. Display number, name, and commission of employees in their alphabetical order of Name.

```
SELECT ENO, ENAME, COMMISSION
FROM EMPLOYEE
ORDER BY ENAME ASC;
```

| Results Explain Describe Saved SQL History |        |            |  |
|--------------------------------------------|--------|------------|--|
| ENO                                        | ENAME  | COMMISSION |  |
| 101                                        | Amit   | 500        |  |
| 104                                        | Geeta  | 800        |  |
| 105                                        | Raj    | -          |  |
| 102                                        | Ramesh | 700        |  |
| 103                                        | Suresh | 600        |  |
| 5 rows returned in 0.01 seconds Download   |        |            |  |

2. Display details of employees from highest to lowest salary.

```
SELECT *
FROM EMPLOYEE
ORDER BY SALARY DESC;
```

| Results Explain Describe Saved SQL History |        |     |            |        |     |
|--------------------------------------------|--------|-----|------------|--------|-----|
| ENO                                        | ENAME  | AGE | COMMISSION | SALARY | DNO |
| 103                                        | Suresh | 32  | 600        | 4500   | 1   |
| 101                                        | Amit   | 30  | 500        | 4000   | 1   |
| 102                                        | Ramesh | 28  | 700        | 3500   | 2   |
| 104                                        | Geeta  | 26  | 800        | 2800   | 3   |
| 105                                        | Raj    | 29  | -          | 2200   | 3   |
| 5 rows returned in 0.01 seconds Download   |        |     |            |        |     |

3. Display name of employees in ascending order of commission.

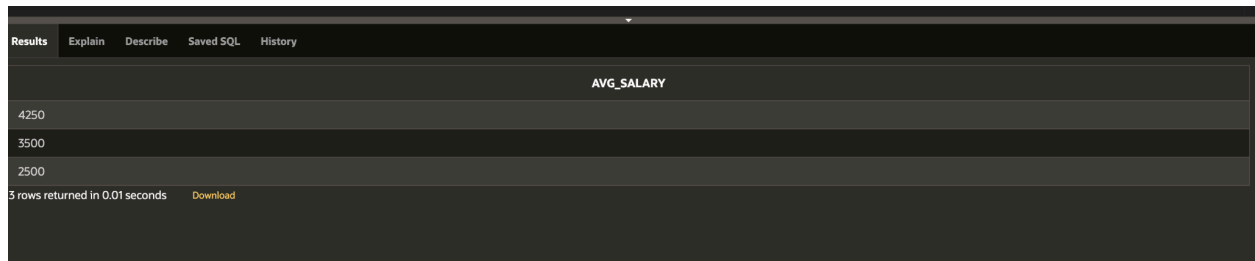
```
SELECT ENAME
FROM EMPLOYEE
ORDER BY COMMISSION ASC NULLS LAST;
```

| Results Explain Describe Saved SQL History |  |  |  |
|--------------------------------------------|--|--|--|
| ENAME                                      |  |  |  |
| Amit                                       |  |  |  |
| Suresh                                     |  |  |  |
| Ramesh                                     |  |  |  |
| Geeta                                      |  |  |  |
| Raj                                        |  |  |  |
| 5 rows returned in 0.00 seconds Download   |  |  |  |

4. Find the average salaries > 2000 for each department without displaying the respective department numbers.

```
SELECT AVG(SALARY) AS Avg_Salary
FROM EMPLOYEE
GROUP BY DNO
```

HAVING AVG(SALARY) > 2000;

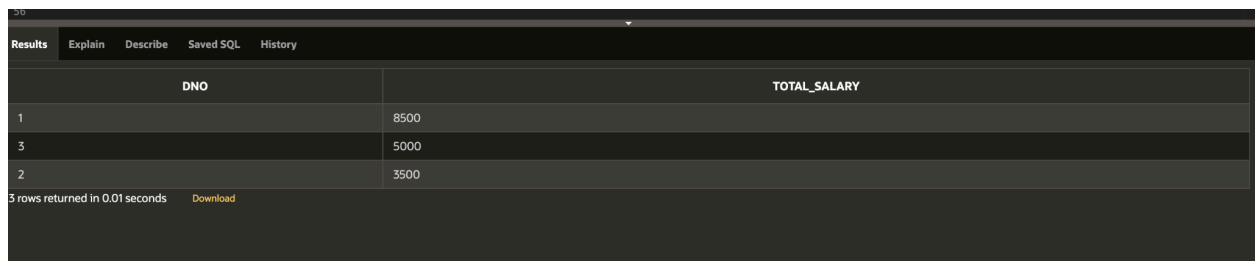


A screenshot of a SQL query results window. The window has a dark theme with a top bar containing tabs: 'Results' (selected), 'Explain', 'Describe', 'Saved SQL', and 'History'. Below the tabs is a table with a single column header 'AVG\_SALARY'. The table contains three rows with values 4250, 3500, and 2500. At the bottom, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

| AVG_SALARY |
|------------|
| 4250       |
| 3500       |
| 2500       |

5. Display total salary for each department with a total salary amount exceeding 3000 and sorts the list by the total salary.

```
SELECT DNO, SUM(SALARY) AS Total_Salary
FROM EMPLOYEE
GROUP BY DNO
HAVING SUM(SALARY) > 3000
ORDER BY Total_Salary DESC;
```

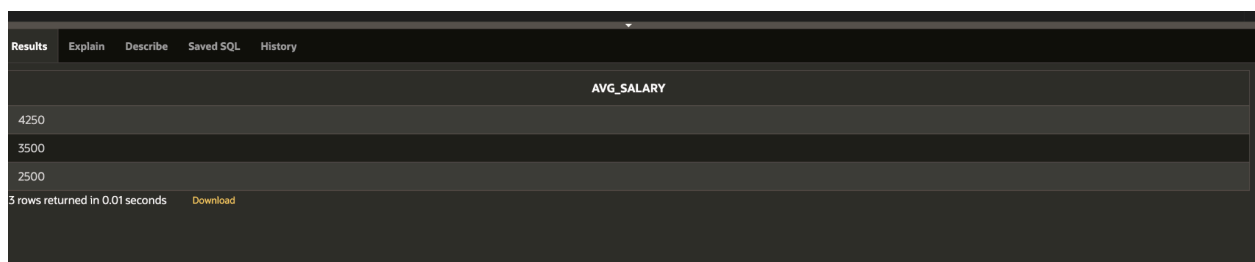


A screenshot of a SQL query results window. The window has a dark theme with a top bar containing tabs: 'Results' (selected), 'Explain', 'Describe', 'Saved SQL', and 'History'. Below the tabs is a table with two columns: 'DNO' and 'TOTAL\_SALARY'. The table contains three rows with values (1, 8500), (3, 5000), and (2, 3500). At the bottom, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

| DNO | TOTAL_SALARY |
|-----|--------------|
| 1   | 8500         |
| 3   | 5000         |
| 2   | 3500         |

6. Find average salaries for each department without displaying the respective department Numbers.

```
SELECT AVG(SALARY) AS Avg_Salary
FROM EMPLOYEE
GROUP BY DNO;
```



A screenshot of a SQL query results window. The window has a dark theme with a top bar containing tabs: 'Results' (selected), 'Explain', 'Describe', 'Saved SQL', and 'History'. Below the tabs is a table with a single column header 'AVG\_SALARY'. The table contains three rows with values 4250, 3500, and 2500. At the bottom, it says '3 rows returned in 0.01 seconds' and has a 'Download' link.

| AVG_SALARY |
|------------|
| 4250       |
| 3500       |
| 2500       |

7. Write a query to display the total salary being paid to each department.

```
SELECT DNO, SUM(SALARY) AS Total_Salary
FROM EMPLOYEE
GROUP BY DNO;
```

ResultsExplainDescribeSaved SQLHistory

| DNO | TOTAL_SALARY |
|-----|--------------|
| 1   | 8500         |
| 2   | 3500         |
| 3   | 5000         |

3 rows returned in 0.00 secondsDownload

CONSIDER THE FOLLOWING SCHEMA

1. Display average loan of each branch.

```
SELECT Bname, AVG(Amount) AS Avg_Loan
FROM BORROW
GROUP BY Bname;
```

ResultsExplainDescribeSaved SQLHistory

| BNAME               | AVG_LOAN |
|---------------------|----------|
| STATE BANK OF INDIA | 25000    |
| UNION               | 16500    |
| HDFC                | 26000    |

3 rows returned in 0.01 secondsDownload

2. Display average loan of each branch and sort them in ascending order of loan amount.

```
SELECT Bname, AVG(Amount) AS Avg_Loan
FROM BORROW
GROUP BY Bname
ORDER BY Avg_Loan ASC;
```

Results

Explain

Describe

Saved SQL

History

| BNAME               | AVG_LOAN |
|---------------------|----------|
| UNION               | 16500    |
| STATE BANK OF INDIA | 25000    |
| HDFC                | 26000    |

3 rows returned in 0.00 seconds

Download

3. List account numbers and their amount of all depositors in ascending order of the deposit Amount.

```
SELECT Actno, Amount
FROM DEPOSIT
ORDER BY Amount ASC;
```

| ACTNO | AMOUNT |
|-------|--------|
| 7     | 1000   |
| 1     | 1000   |
| 4     | 1200   |
| 6     | 2000   |
| 5     | 3000   |
| 3     | 3500   |
| 2     | 5000   |
| 8     | 5000   |
| 101   | 5000   |
| 101   | 7000   |

More than 10 rows available. Increase rows selector to view more rows.

4. List the customers and their loan amount in the descending order of customer names.

```
SELECT Cname, Amount
FROM BORROW
ORDER BY Cname DESC;
```

| CNAME  | AMOUNT |
|--------|--------|
| Suresh | 25000  |
| Ramesh | 20000  |
| Raj    | 27000  |
| Geeta  | 18000  |
| Anita  | 30000  |
| Amit   | 15000  |

6 rows returned in 0.01 seconds [Download](#)

5. List the branches having a sum of deposit more than 4000.

```
SELECT Bname, SUM(Amount) AS Total_Deposit
FROM DEPOSIT
GROUP BY Bname
HAVING SUM(Amount) > 4000;
```

| BNAME               | TOTAL_DEPOSIT |
|---------------------|---------------|
| KOTAK MAHINDRA BANK | 5000          |
| STATE BANK OF INDIA | 13000         |
| UNION               | 5000          |
| NAGPUR              | 7000          |

4 rows returned in 0.01 seconds [Download](#)

6. Give the name of branch where number of depositors is less than 2.

```
SELECT Bname
FROM DEPOSIT
GROUP BY Bname
HAVING COUNT(DISTINCT Cname) < 2;
```

| Results                                                  | Explain | Describe | Saved SQL | History |
|----------------------------------------------------------|---------|----------|-----------|---------|
| BNAME                                                    |         |          |           |         |
| KOTAK MAHINDRA BANK                                      |         |          |           |         |
| UNION                                                    |         |          |           |         |
| UCO                                                      |         |          |           |         |
| CITY                                                     |         |          |           |         |
| ANDHRA BANK                                              |         |          |           |         |
| NAGPUR                                                   |         |          |           |         |
| YES BANK                                                 |         |          |           |         |
| DENA BANK                                                |         |          |           |         |
| 8 rows returned in 0.03 seconds <a href="#">Download</a> |         |          |           |         |