PRACTICAL : 2

AIM : To understand & apply the following SQL concepts:

- **Logical Operators: AND, OR, NOT**
- **BETWEEN...AND, NOT BETWEEN...AND**
- **LIKE Predicate**
- **IN, NOT IN Predicates**
- **GROUP BY**
- **Scalar / Aggregate Functions**

1. Logical Operators: AND, OR, NOT

▶ **AND – All conditions must be true**

```
SELECT * FROM Employees
WHERE Department = 'HR' AND Salary > 50000;
```

Explanation: Fetch employees who are in HR **and** earn more than 50,000.

▶ **OR – At least one condition must be true**

```
SELECT * FROM Employees
WHERE Department = 'HR' OR Salary > 50000;
```

Explanation: Fetch employees who are in HR **or** earn more than 50,000.

▶ **NOT – Negates a condition**

```
SELECT *
FROM Employees WHERE NOT
Department = 'HR';
```

Explanation: Fetch employees **not** in the HR department.

2. BETWEEN … AND | NOT BETWEEN … AND

▶ **BETWEEN**

```
SELECT * FROM Products
WHERE Price BETWEEN 100 AND 500;
```

Explanation: Fetch products with price **from 100 to 500** (inclusive).

▶ **NOT BETWEEN**

```
SELECT * FROM Products
WHERE Price NOT BETWEEN 100 AND 500;
```

Explanation: Fetch products with price **less than 100 or more than 500**.

3. **LIKE Predicate Wildcards:**

% → zero or more characters

_ → exactly one character

▶ **Examples:**

```
-- Starts with 'A'
SELECT * FROM Customers
WHERE Name LIKE 'A%';
-- Ends with 'son'
SELECT * FROM Customers
WHERE Name LIKE '%son';
-- Contains 'mit'
SELECT * FROM Customers
WHERE Name LIKE '%mit%';
-- Exactly 5-letter names starting with 'A'
SELECT * FROM Customers WHERE
Name LIKE 'A_____';
```

4. **IN and NOT IN Predicates**

▶ **IN**

```
SELECT * FROM Students
WHERE Grade IN ('A', 'B', 'C');
```

Explanation: Fetch students with grade A, B, or C.

▶ **NOT IN**

```
SELECT * FROM Students
WHERE Grade NOT IN ('F', 'D');
```

Explanation: Fetch students whose grades are **not** F or D.

PREREQUISITE STEPS

Step 1 : Use tee PATH\TO\FOLDER\FILENAME.txt in MySQL CLI to start saving all commands and outputs of the current session into a file until stopped with notee or else session is closed.

```
24012011142_Vatsal [(none)] Sat Aug 23 11:51:41 2025
> tee C:\Users\Vatsal\Desktop\Sem3\DBMS\Practical-1\Tasks.txt
Logging to file 'C:\Users\Vatsal\Desktop\Sem3\DBMS\Practical-1\Tasks.txt'
```

- **SYNTAX : tee PATH\TO\FOLDER\FILENAME.txt**
- tee in MySQL Command-Line Client enables logging of all commands and their results into a file during the same session. From this point onward, all entered commands and their outputs will be written to the specified file.
- This is useful for: Record keeping, Creating reports of SQL execution, Debugging query results later. The log continues until you turn it off with "notee" Command

Step 2 : Changing the MySQL Prompt

```
mysql> PROMPT 24012011142_Vatsal [\d] \D \n>
PROMPT set to '24012011142_Vatsal [\d] \D \n> '
24012011142_Vatsal [(none)] Sat Aug 23 11:51:41 2025
>
```

PROMPT changes how the MySQL CLI prompt looks.

Components:

- FULLENROLLMENT_FULLNAME → Custom label or project name
- [\d] → Shows the current database in square brackets
- \D → Shows the current date
- \n → Inserts a newline before
- > → The actual command prompt symbol

Step 3 : Either Create a new Database or Use Existing Database

```
24012011142_Vatsal [(none)] Sat Aug 23 12:02:12 2025
> CREATE DATABASE Practical_1_24012011142;
Query OK, 1 row affected (0.01 sec)
```

Step 4 : Select a Existing Database

```
24012011142_Vatsal [(none)] Sat Aug 23 12:02:24 2025
> USE Practical_1_24012011142;
Database changed
```

REMOVE Note : (also remove these types of notes after understanding) note : option a : you can use existing database created for practical 1 as mentioned above (only if it is performed on your personal system & if you have successfully completed,

REMOVE Note : option b : otherwise you need to create new database for practical 2 then either use earlier typed queries or create new tables with data

REMOVE Note : for the first question (drop address column from account table (if not created, create it first)), you will require address column to drop it, if you have performed following option a, there will be no issue, but if you have done it using option b you need to firstly add new column address : "ALTER TABLE ACCOUNT_FULLENROLLMENT ADD address VARCHAR(20);", then you will be able to drop column for q-1.

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:56:04 2025
>SET SQL_SAFE_UPDATES = 0;
Query OK, 0 rows affected (0.00 sec)
```

- **Purpose:** Disables safe update mode in MySQL for the current session.
- **Safe Mode Function:** Normally blocks UPDATE or DELETE without WHERE, indexed column, or LIMIT.
- **Effect of Disabling:** Allows unrestricted updates or deletions, even on all rows.
- **Common Use:** Bulk updates/deletions without strict conditions.
- **Scope:** Applies only to the current session; resets after disconnecting.
- **Caution:** Increases risk of accidental data loss—use carefully.
- **Re-enable:** Use SET SQL_SAFE_UPDATES = 1; when finished.

TASK

Answer following Queries based on above 3 tables.

1. **Drop address column from ACCOUNT_24012011142 table (if not created, create it first).**

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:49:32 2025
>DESC ACCOUNT_24012011142;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| acc_no     | varchar(5)    | NO   | PRI | NULL    |       |
| Name       | varchar(30)   | YES  |     | NULL    |       |
| City       | varchar(20)   | YES  |     | NULL    |       |
| Balance    | decimal(10,2) | YES  |     | NULL    |       |
| loan_taken | varchar(5)    | YES  |     | NULL    |       |
| address    | varchar(20)   | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:49:50 2025
>ALTER TABLE ACCOUNT_24012011142 DROP COLUMN address;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0


24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:50:00 2025
>DESC ACCOUNT_24012011142;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| acc_no     | varchar(5)    | NO   | PRI | NULL    |       |
| Name       | varchar(30)   | YES  |     | NULL    |       |
| City       | varchar(20)   | YES  |     | NULL    |       |
| Balance    | decimal(10,2) | YES  |     | NULL    |       |
| loan_taken | varchar(5)    | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

2. Rename Name to New_name in Account table.

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:53:22 2025
>DESC ACCOUNT_24012011142;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| acc_no     | varchar(5)    | NO   | PRI | NULL    |       |
| Name       | varchar(30)   | YES  |     | NULL    |       |
| City       | varchar(20)   | YES  |     | NULL    |       |
| Balance    | decimal(10,2) | YES  |     | NULL    |       |
| loan_taken | varchar(5)    | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:53:36 2025
>ALTER TABLE ACCOUNT_24012011142 RENAME COLUMN Name TO New_name;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0

24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:53:42 2025
>DESC ACCOUNT_24012011142;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| acc_no     | varchar(5)    | NO   | PRI | NULL    |       |
| New_name   | varchar(30)   | YES  |     | NULL    |       |
| City       | varchar(20)   | YES  |     | NULL    |       |
| Balance    | decimal(10,2) | YES  |     | NULL    |       |
| loan_taken | varchar(5)    | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

3. Retrieve specific information for the account holder who are not in ‘Ahmedabad’ or ‘Vadodara’.

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:54:20 2025
>SELECT * FROM ACCOUNT_24012011142 WHERE City NOT IN ('Ahmedabad', 'Vadodara');
+--------+--------------+---------+-----------+------------+
| acc_no | New_name     | City    | Balance   | loan_taken |
+--------+--------------+---------+-----------+------------+
| A001   | Patel Hiren  | Mehsana |  50000.00 | YES        |
| A002   | Patel Ramesh | Mehsana |  50000.00 | YES        |
| A005   | Kothari Nehal | Kherva | 100000.00 | YES        |
+--------+--------------+---------+-----------+------------+
3 rows in set (0.00 sec)
```

4. **Display only those data whose account number is 'A001' and city is 'Mehsana'.**

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:54:28 2025
>SELECT * FROM ACCOUNT_24012011142 WHERE acc_no = 'A001' AND City = 'Mehsana';
+--------+-------------+---------+----------+------------+
| acc_no | New_name    | City    | Balance  | loan_taken |
+--------+-------------+---------+----------+------------+
| A001   | Patel Hiren | Mehsana | 50000.00 | YES        |
+--------+-------------+---------+----------+------------+
1 row in set (0.00 sec)
```

5. **Display only those data whose account number is 'A001' or city is 'Mehsana'.**

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:54:36 2025
>SELECT * FROM ACCOUNT_24012011142 WHERE acc_no = 'A001' OR City = 'Mehsana';
+--------+--------------+---------+----------+------------+
| acc_no | New_name     | City    | Balance  | loan_taken |
+--------+--------------+---------+----------+------------+
| A001   | Patel Hiren  | Mehsana | 50000.00 | YES        |
| A002   | Patel Ramesh | Mehsana | 50000.00 | YES        |
+--------+--------------+---------+----------+------------+
2 rows in set (0.00 sec)
```

6. **Retrieve those records of Account holders whose balance between 75000 and 100000.**

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:54:45 2025
>SELECT * FROM ACCOUNT_24012011142 WHERE Balance BETWEEN 75000 AND 100000;
+--------+---------------+-----------+-----------+------------+
| acc_no | New_name      | City      | Balance   | loan_taken |
+--------+---------------+-----------+-----------+------------+
| A003   | Dave Hardik   | Ahmedabad |  75000.00 | NO         |
| A005   | Kothari Nehal | Kherva    | 100000.00 | YES        |
+--------+---------------+-----------+-----------+------------+
2 rows in set (0.00 sec)
```

7. **Retrieve those records of Account holders whose balance not between 50000 and 75000.**

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:54:57 2025
>SELECT * FROM ACCOUNT_24012011142 WHERE Balance NOT BETWEEN 50000 AND 75000;
+--------+---------------+--------+-----------+------------+
| acc_no | New_name      | City   | Balance   | loan_taken |
+--------+---------------+--------+-----------+------------+
| A005   | Kothari Nehal | Kherva | 100000.00 | YES        |
+--------+---------------+--------+-----------+------------+
1 row in set (0.00 sec)
```

8. Display only those records whose amount is 5000, 15000, 30000 from installment table.

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:55:08 2025
>SELECT * FROM INSTALLMENT_24012011142 WHERE Amount IN (5000, 15000, 30000);
+---------+---------------+------------------+----------+
| loan_no | installment_no | installment_date | Amount   |
+---------+---------------+------------------+----------+
| L001    | I001           | 2004-02-02       | 15000.00 |
+---------+---------------+------------------+----------+
1 row in set (0.00 sec)
```

9. Display only those records whose amount is not in 5000, 15000, 30000 from installment table.

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:55:24 2025
>SELECT * FROM INSTALLMENT_24012011142 WHERE Amount NOT IN (5000, 15000, 30000);
+---------+---------------+------------------+----------+
| loan_no | installment_no | installment_date | Amount   |
+---------+---------------+------------------+----------+
| L002    | I002           | 2004-06-18       | 20000.00 |
| L003    | I003           | 2004-07-15       | 20000.00 |
+---------+---------------+------------------+----------+
2 rows in set (0.00 sec)
```

10.Display those records of account holders whose name starts with ‘D’.

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:55:36 2025
>SELECT * FROM ACCOUNT_24012011142 WHERE New_name LIKE 'D%';
+--------+-------------+-----------+----------+------------+
| acc_no | New_name    | City      | Balance  | loan_taken |
+--------+-------------+-----------+----------+------------+
| A003   | Dave Hardik | Ahmedabad | 75000.00 | NO         |
+--------+-------------+-----------+----------+------------+
1 row in set (0.00 sec)
```

11.Display System date.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:03:38 2025
>SELECT
    ->     SYSDATE() AS SystemDate,
    ->     CURDATE() AS CurrentDate,
    ->     NOW() AS NowDateTime
    -> FROM DUAL;
+---------------------+-------------+---------------------+
| SystemDate          | CurrentDate | NowDateTime         |
+---------------------+-------------+---------------------+
| 2025-09-09 00:18:41 | 2025-09-09  | 2025-09-09 00:18:41 |
+---------------------+-------------+---------------------+
1 row in set (0.00 sec)
```

12.Find the date,15 days after today's date.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:18:41 2025
>SELECT SYSDATE() + INTERVAL 15 DAY AS DateAfter15Days;
+---------------------+
| DateAfter15Days     |
+---------------------+
| 2025-09-24 00:19:43 |
+---------------------+
1 row in set (0.00 sec)

24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:19:43 2025
>SELECT DATE_ADD(SYSDATE(), INTERVAL 15 DAY) AS DateAfter15Days_Alt;
+---------------------+
| DateAfter15Days_Alt |
+---------------------+
| 2025-09-24 00:19:52 |
+---------------------+
1 row in set (0.00 sec)
```

13.Find the date,20 days before today's date.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:19:52 2025
>SELECT SYSDATE() - INTERVAL 20 DAY AS DateBefore20Days;
+---------------------+
| DateBefore20Days    |
+---------------------+
| 2025-08-20 00:19:58 |
+---------------------+
1 row in set (0.00 sec)

24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:19:58 2025
>SELECT DATE_SUB(SYSDATE(), INTERVAL 20 DAY) AS DateBefore20Days_Alt;
+---------------------+
| DateBefore20Days_Alt |
+---------------------+
| 2025-08-20 00:20:02  |
+---------------------+
1 row in set (0.00 sec)
```

14.Perform the following operation using DUAL table.

5*5, 34+34, 1000/300, length of 'uvpce', display only month of system date

```
24012011142_Vatsal [practical_1_24012011142] Mon Sep  8 23:55:58 2025
>SELECT
    ->     5 * 5 AS Multiplication,
    ->     34 + 34 AS Addition,
    ->     LENGTH('uvpce') AS StringLength,
    ->     MONTHNAME(CURDATE()) AS MonthName
    -> FROM DUAL;
+----------------+----------+--------------+-----------+
| Multiplication | Addition | StringLength | MonthName |
+----------------+----------+--------------+-----------+
|             25 |       68 |            5 | September |
+----------------+----------+--------------+-----------+
1 row in set (0.00 sec)
```

Function Based Queries :

1. GROUP BY

The GROUP BY clause is used to group rows that have the same values in specified columns. It is commonly used with aggregate functions such as SUM(), AVG(), COUNT(), MAX(), and MIN().

Syntax:

SELECT column_name, AGGREGATE_FUNCTION(column_name)

FROM table_name

GROUP BY column_name;

Example:

-- Total salary by department

SELECT Department, SUM(Salary) AS Total_Salary

FROM Employees

GROUP BY Department;

-- Average marks by subject

SELECT Subject, AVG(Marks) AS Average_Marks

FROM Results

GROUP BY Subject;

2. SCALAR FUNCTIONS

Scalar functions return a single value based on the input value. They operate on each row individually.

Example:

-- Convert name to uppercase

SELECT UPPER(Name) AS Upper_Name FROM Customers;

-- Round salary to nearest thousand

SELECT Name, ROUND(Salary, -3) AS Rounded_Salary FROM Employees;

-- Length of name

SELECT Name, LENGTH(Name) AS Name_Length FROM Customers;

-- Current date and time

SELECT NOW() AS Current_Timestamp;

- Common Scalar Functions:

Function	Description
UPPER()	Converts text to uppercase
LOWER()	Converts text to lowercase
LEN() or LENGTH()	Returns length of a string
ROUND()	Rounds numeric value
NOW()	Returns current date & time
GETDATE()	Same as NOW() in some DBs
ABS()	Returns absolute value

Summary Table

Concept	Used For	Example
GROUP BY	Grouping data & aggregation	GROUP BY Department
COUNT()	Count rows	COUNT(*)
SUM()	Sum of values	SUM(Salary)
AVG()	Average of values	AVG(Marks)
UPPER()	Convert to uppercase	UPPER(Name)
ROUND()	Round numbers	ROUND(Salary, 2)
LENGTH()	Get string length	LENGTH(Name)
NOW()	Current date and time	NOW()

Create TABLE "TRANSACTION_yorEnNo" as given below.

Acc_no	Tr_date	Amt	Type_of_tr	Mode_of_pay
A001	3-may-21	10000	D	Cash
A002	5-july-21	5000	W	Cheque
A003	12-Aug-21	25000	D	Cheque
A004	15-may-21	30000	D	Cheque
A005	22-oct-21	15000	W	Cash

Perform given queries:

1. **Find the total transaction amount of account holder from transaction table.**

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:28:02 2025
>SELECT SUM(Amt) AS Total_Amount FROM TRANSACTION_24012011142;
+--------------+
| Total_Amount |
+--------------+
|     85000.00 |
+--------------+
1 row in set (0.00 sec)
```

2. **Find minimum amount of transaction table.**

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:28:24 2025
>SELECT MIN(Amt) AS Minimum_Amount FROM TRANSACTION_24012011142;
+----------------+
| Minimum_Amount |
+----------------+
|        5000.00 |
+----------------+
1 row in set (0.00 sec)
```

3. **Find maximum amount of transaction table.**

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:28:30 2025
>SELECT MAX(Amt) AS Maximum_Amount FROM TRANSACTION_24012011142;
+----------------+
| Maximum_Amount |
+----------------+
|       30000.00 |
+----------------+
1 row in set (0.00 sec)
```

4. **Count the total account holders from transaction table.**

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:28:35 2025
>SELECT COUNT(*) AS Total_Transactions FROM TRANSACTION_24012011142;
+--------------------+
| Total_Transactions |
+--------------------+
|                  5 |
+--------------------+
1 row in set (0.01 sec)
```

5. Count only those records whose made of payment is ‘Cheque’.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:28:55 2025
>SELECT COUNT(*) AS Cheque_Transactions FROM TRANSACTION_24012011142 WHERE Mode_of_pay = 'Cheque';
+---------------------+
| Cheque_Transactions |
+---------------------+
|                   3 |
+---------------------+
1 row in set (0.00 sec)
```

6. Count only those records whose Type_of_tr is not ‘D’.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:29:04 2025
>SELECT COUNT(*) AS Not_Deposit_Transactions FROM TRANSACTION_24012011142 WHERE NOT Type_of_tr = 'D';
+----------------------+
| Deposit_Transactions |
+----------------------+
|                    2 |
+----------------------+
1 row in set (0.00 sec)
```

7. Count only those records whose transaction made in the month of ‘may’.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:29:19 2025
>SELECT COUNT(*) AS May_Transactions FROM TRANSACTION_24012011142 WHERE DATE_FORMAT(Tr_date, '%m') = '05';
+------------------+
| May_Transactions |
+------------------+
|                2 |
+------------------+
1 row in set (0.00 sec)
```

8. Find the average value of transaction.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:29:31 2025
>SELECT AVG(Amt) AS Average_Amount FROM TRANSACTION_24012011142;
+-----------------+
| Average_Amount |
+-----------------+
|   17000.000000 |
+-----------------+
1 row in set (0.00 sec)
```

9. Display the result of 4 rest to 4 (use power function).

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:29:44 2025
>SELECT POWER(4, 4) AS Power_Value FROM DUAL;
+--------------+
| Power_Value |
+--------------+
|          256 |
+--------------+
1 row in set (0.00 sec)
```

10.Find the square root of 25 (use sqrt function).

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:29:53 2025
>SELECT SQRT(25) AS Square_Root FROM DUAL;
+-------------+
| Square_Root |
+-------------+
|           5 |
+-------------+
1 row in set (0.00 sec)
```

11.Write the query for the following inbuilt Function.

LOWER, INITCAP, UPPER, SUBSTR, LENGTH, LTRIM, RTRIM, LPAD, RPAD.

```
24012011142_Vatsal [practical_1_24012011142] Tue Sep  9 00:30:04 2025
>SELECT
    ->     LOWER('Hello World') AS lower_case,
    ->     UPPER('Hello World') AS upper_case,
    ->     SUBSTR('Hello World', 7, 5) AS sub_string,
    ->     LENGTH('Hello World') AS text_length,
    ->     LTRIM('  Hello World  ') AS left_trim,
    ->     RTRIM('  Hello World  ') AS right_trim,
    ->     LPAD('SQL', 8, '*') AS left_pad,
    ->     RPAD('SQL', 8, '-') AS right_pad
    -> FROM DUAL;
+-------------+-------------+------------+-------------+---------------+---------------+----------+-----------+
| lower_case  | upper_case  | sub_string | text_length | left_trim     | right_trim    | left_pad | right_pad |
+-------------+-------------+------------+-------------+---------------+---------------+----------+-----------+
| hello world | HELLO WORLD | World      |          11 | Hello World   |   Hello World | *****SQL | SQL-----  |
+-------------+-------------+------------+-------------+---------------+---------------+----------+-----------+
1 row in set (0.00 sec)
```