Practical File

OF

Database Management System

(CS – 502)

SUBMITTED TO-

Dr.Archana Tomar

Asst. Professor

Department of CSE

ITM GOI, Gwalior

SUBMITTED BY-

Ashikaa Saxena

0905CS231050

CS - A (B2)

5th Semester

Institute of Technology & Management, Gwalior

Department of Computer Science & Engineering

Continuous Assessment Sheet (CAS)

Academic Year: July – Dec 2025 **Course Name:** DBMS Lab (CS-502)

Name of Student: Ashikaa Saxena **Class :** B.Tech (CSE) V Sem (B) **Batch :** B2 **Roll No.**: 0905CS231050

Exp. No.	Date	Title of Experiment	Concept Understanding (10)	Program Execution and (10)	Oral and Report Writing (10)	Total (30)	Date of Submission	Signature of Staff
1		Learn and apply the concept of Data Definition Language (DDL) commands in RDBMS, Data Manipulation Language (DML) and Data Control Language (DCL).						
2		Simple queries: selection, projection, sorting on a simple table, Small-large number of attributes, Distinct output values, Renaming attributes, Computed attributes Simple-complex conditions (AND, OR, NOT).						
3		Partial Matching operators (LIKE, %, _, *, ?), ASC-DESC ordering combinations Checking for Nulls.						
4		Multi-table queries (JOIN OPERATIONS), Simple joins (no INNER JOIN) Aliasing tables – Full/Partial name qualification, Inner-joins (two and more (different) tables), Inner-recursive- joins (joining to itself), Outer-joins (restrictions as part of the WHERE and ON clauses), Using where & having clauses.						
5		Nested queries: In, Not in Exists, Not Exists Dynamic relations (as part of SELECT, FROM, and WHERE clauses).						
6		Set Oriented Operations: Union, Difference, Intersection, Division.						
7		PL/SQL Programming I: Programs using named and unnamed blocks, Programs using Cursors, Cursor loops and records.						
8		PL/SQL Programming II: Creating stored procedures, functions, Trigger						
9		create your optimize database in a team for any given problem						
					Total Marks			

Signature
Head of Department

Sign
Course Teacher

Signature
Student

Institute of Technology & Management, Gwalior

Rubrics for Lab Evaluation

Course: DBMS Lab (CS 502)

Criteria	Level 1 Poor	Level 2 Average	Level 3 Good	Level 4 Very Good
Range(Marks)	0-2	3-5	6-8	9-10
Concept Understandings (10M)	Student is not able to understand given problem statement.	Student is able to understand given problem statement.	Student is able to understand given problem statement and interpret it as well.	Student is able to demonstrate knowledge about given problem statement convincingly.
Range(Marks)	0-2	3-5	6-8	9-10
Program Execution (10M)	Not able to write, apply and implement concepts.	Able to write but not apply and implement concepts.	Able to write, apply but not implement concepts.	Able to write, apply and implement concepts.
Range(Marks)	0-2	3-5	6-8	9-10
File submission and Oral (10M)	Not able to submit file in given deadline and poor performance in oral as well.	Able to submit file in given deadline but incomplete and average performance in oral as well.	Able to submit file in given deadline as well as good performance in oral.	Able to submit file in given deadline with exceptional performance in oral.

Signature
Course I/C

Signature
Course coordinator

Signature
Head of Department

EXPERIMENT-1

DDL (Data Definition Language)-

Used to define **or** change **the** structure of a database (schema). It does not modify the actual data, only the structure.

Commands:

- **CREATE** - create a new database/table
- **ALTER** - modify the structure (add/remove/change columns)
- **DROP** - delete a database/table
- **TRUNCATE** - remove all rows from a table but keep the structure
- **RENAME** - rename a table

DML(Data Manipulation Language)-

Used to insert**,** update**,** delete**, or** retrieve data from a table.

Commands:

- **INSERT** - add data
- **UPDATE** - change data
- **DELETE** - remove data
- **SELECT** - fetch/read data

DCL (Data Control Language)-

Used to control access and permissions in a database.

Commands:

- **GRANT** - give permissions to a user
- **REVOKE** - take back permissions

Create Database and Use Database

```
mysql> create database hotelmg;
Query OK, 1 row affected (0.04 sec)

mysql> USE hotelmg;
Database changed
```

Create tables

1.GUESTS

```
mysql> CREATE TABLE guests (
    ->     guest_id        INT AUTO_INCREMENT PRIMARY KEY,
    ->     first_name      VARCHAR(50) NOT NULL,
    ->     last_name       VARCHAR(50) NOT NULL,
    ->     phone           VARCHAR(20) UNIQUE,
    ->     email           VARCHAR(100) UNIQUE,
    ->     id_doc_type     ENUM('Passport','ID','DriverLicense') DEFAULT NULL,
    ->     id_doc_number   VARCHAR(50) UNIQUE,
    ->     created_at      TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.01 sec)
```

2.ROOM_RATES

```
mysql> CREATE TABLE room_rates (
    ->     rate_id         INT AUTO_INCREMENT PRIMARY KEY,
    ->     room_type       ENUM('Single','Double','Deluxe','Suite') NOT NULL,
    ->     season          ENUM('Low','Mid','High','Peak') NOT NULL,
    ->     base_rate       DECIMAL(10,2) NOT NULL CHECK (base_rate >= 0),
    ->     UNIQUE (room_type, season)
    -> );
Query OK, 0 rows affected (0.00 sec)
```

3.PAYMENTS

```
mysql>
mysql> CREATE TABLE payments (
    ->     payment_id      INT AUTO_INCREMENT PRIMARY KEY,
    ->     booking_id      INT NOT NULL,
    ->     amount          DECIMAL(10,2) NOT NULL CHECK (amount > 0),
    ->     method          ENUM('Cash','Card','UPI','BankTransfer') NOT NULL,
    ->     paid_at         TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     note            VARCHAR(255),
    ->     CONSTRAINT fk_payment_booking FOREIGN KEY (booking_id) REFERENCES bookings(booking_id) ON DELETE CASCADE
    -> );
Query OK, 0 rows affected (0.04 sec)
```

4.STAFF

```
mysql> CREATE TABLE staff (
    ->   staff_id       INT AUTO_INCREMENT PRIMARY KEY,
    ->   first_name     VARCHAR(50) NOT NULL,
    ->   last_name      VARCHAR(50) NOT NULL,
    ->   role           ENUM('Manager','Reception','Housekeeping','Security','Kitchen') NOT NULL,
    ->   phone          VARCHAR(20) UNIQUE,
    ->   email          VARCHAR(100) UNIQUE,
    ->   hired_on       DATE NOT NULL
    -> );
Query OK, 0 rows affected (0.05 sec)
```

5. BOOKINGS

```
mysql> CREATE TABLE bookings (
    ->   booking_id     INT AUTO_INCREMENT PRIMARY KEY,
    ->   guest_id       INT NOT NULL,
    ->   room_id        INT NOT NULL,
    ->   check_in       DATE NOT NULL,
    ->   check_out      DATE NOT NULL,
    ->   adults         INT NOT NULL CHECK (adults >= 1),
    ->   children       INT NOT NULL DEFAULT 0 CHECK (children >= 0),
    ->   status         ENUM('Reserved','CheckedIn','CheckedOut','Cancelled') NOT NULL DEFAULT 'Reserved
    ->   created_at     TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->   CONSTRAINT fk_booking_guest FOREIGN KEY (guest_id) REFERENCES guests(guest_id),
    ->   CONSTRAINT fk_booking_room  FOREIGN KEY (room_id)  REFERENCES rooms(room_id),
    ->   CONSTRAINT chk_dates CHECK (check_out > check_in)
    -> );
Query OK, 0 rows affected (0.02 sec)
```

6. ROOM

```
mysql>  CREATE TABLE rooms (
    ->   room_id        INT AUTO_INCREMENT PRIMARY KEY,
    ->   room_number    VARCHAR(10) NOT NULL UNIQUE,
    ->   floor          INT NOT NULL,
    ->   type           ENUM('Single','Double','Deluxe','Suite') NOT NULL,
    ->   status         ENUM('Available','Occupied','Maintenance') NOT NULL DEFAULT 'Available'
    -> );
Query OK, 0 rows affected (0.04 sec)
```

7. AUDIT_BOOKINGS

```
mysql> CREATE TABLE audit_bookings (
    ->   audit_id       INT AUTO_INCREMENT PRIMARY KEY,
    ->   booking_id     INT NOT NULL,
    ->   old_status     VARCHAR(20),
    ->   new_status     VARCHAR(20),
    ->   changed_at     TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.04 sec)
```

8.SERVICES

```
mysql> CREATE TABLE services (
    ->   service_id     INT AUTO_INCREMENT PRIMARY KEY,
    ->   name           VARCHAR(100) NOT NULL UNIQUE,
    ->   unit_price     DECIMAL(10,2) NOT NULL CHECK (unit_price >= 0),
    ->   active         BOOLEAN NOT NULL DEFAULT TRUE
    -> );
Query OK, 0 rows affected (0.05 sec)
```

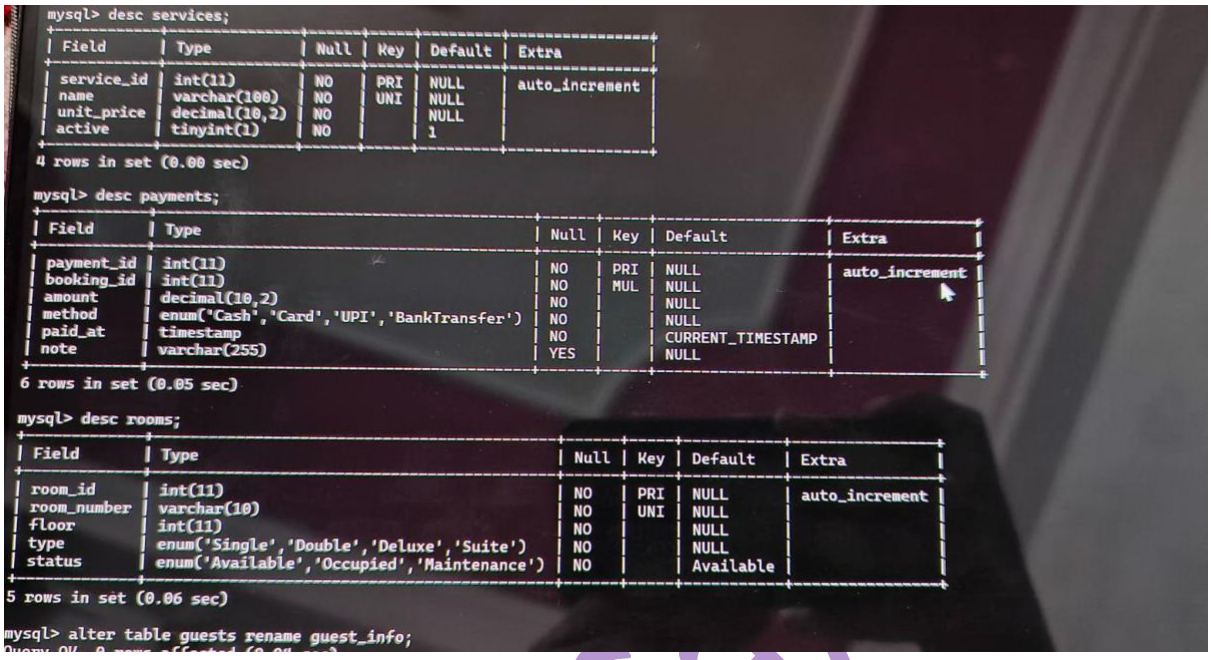
Show tables

```
mysql> show tables;
+--------------------+
| Tables_in_hotelmg  |
+--------------------+
| audit_bookings     |
| booking_services   |
| bookings           |
| guest_info         |
| payments           |
| room_rates         |
| rooms              |
| services           |
| staff              |
+--------------------+
9 rows in set (0.00 sec)
```

Description of all tables

```
mysql> desc bookings;
+--------------+-----------------------------------------------------+------+-----+-------------------+----------------+
| Field        | Type                                                | Null | Key | Default           | Extra          |
+--------------+-----------------------------------------------------+------+-----+-------------------+----------------+
| booking_id   | int(11)                                             | NO   | PRI | NULL              | auto_increment |
| guest_id     | int(11)                                             | NO   | MUL | NULL              |                |
| room_id      | int(11)                                             | NO   | MUL | NULL              |                |
| check_in     | date                                                | NO   |     | NULL              |                |
| check_out    | date                                                | NO   |     | NULL              |                |
| adults       | int(11)                                             | NO   |     | NULL              |                |
| children     | int(11)                                             | NO   |     | 0                 |                |
| status       | enum('Reserved','CheckedIn','CheckedOut','Cancelled') | NO   |     | Reserved          |                |
| created_at   | timestamp                                           | NO   |     | CURRENT_TIMESTAMP |                |
+--------------+-----------------------------------------------------+------+-----+-------------------+----------------+
9 rows in set (0.04 sec)

mysql> desc staff;
+------------+-----------------------------------------------------+------+-----+---------+----------------+
| Field      | Type                                                | Null | Key | Default | Extra          |
+------------+-----------------------------------------------------+------+-----+---------+----------------+
| staff_id   | int(11)                                             | NO   | PRI | NULL    | auto_increment |
| first_name | varchar(50)                                         | NO   |     | NULL    |                |
| last_name  | varchar(50)                                         | NO   |     | NULL    |                |
| role       | enum('Manager','Reception','Housekeeping','Security','Kitchen') | NO   |     | NULL    |                |
| phone      | varchar(20)                                         | YES  | UNI | NULL    |                |
| email      | varchar(100)                                        | YES  | UNI | NULL    |                |
| hired_on   | date                                                | NO   |     | NULL    |                |
+------------+-----------------------------------------------------+------+-----+---------+----------------+
```



```
mysql> desc services;
+--------------+---------------+------+-----+---------+----------------+
| Field        | Type          | Null | Key | Default | Extra          |
+--------------+---------------+------+-----+---------+----------------+
| service_id   | int(11)       | NO   | PRI | NULL    | auto_increment |
| name         | varchar(100)  | NO   | UNI | NULL    |                |
| unit_price   | decimal(10,2) | NO   |     | NULL    |                |
| active       | tinyint(1)    | NO   |     | 1       |                |
+--------------+---------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)

mysql> desc payments;
+--------------+-----------------------------------------------+------+-----+-------------------+----------------+
| Field        | Type                                          | Null | Key | Default           | Extra          |
+--------------+-----------------------------------------------+------+-----+-------------------+----------------+
| payment_id   | int(11)                                       | NO   | PRI | NULL              | auto_increment |
| booking_id   | int(11)                                       | NO   | MUL | NULL              |                |
| amount       | decimal(10,2)                                 | NO   |     | NULL              |                |
| method       | enum('Cash','Card','UPI','BankTransfer')      | NO   |     | NULL              |                |
| paid_at      | timestamp                                     | NO   |     | CURRENT_TIMESTAMP |                |
| note         | varchar(255)                                  | YES  |     | NULL              |                |
+--------------+-----------------------------------------------+------+-----+-------------------+----------------+
6 rows in set (0.05 sec)

mysql> desc rooms;
+-------------+-----------------------------------------------+------+-----+-----------+----------------+
| Field       | Type                                          | Null | Key | Default   | Extra          |
+-------------+-----------------------------------------------+------+-----+-----------+----------------+
| room_id     | int(11)                                       | NO   | PRI | NULL      | auto_increment |
| room_number | varchar(10)                                   | NO   | UNI | NULL      |                |
| floor       | int(11)                                       | NO   |     | NULL      |                |
| type        | enum('Single','Double','Deluxe','Suite')      | NO   |     | NULL      |                |
| status      | enum('Available','Occupied','Maintenance')    | NO   |     | Available |                |
+-------------+-----------------------------------------------+------+-----+-----------+----------------+
5 rows in set (0.06 sec)

mysql> alter table guests rename guest_info;
```

Rename table guest_info to guests_details

```
mysql> Alter table customers rename customer_info;
Query OK, 0 rows affected (0.16 sec)

mysql> show tables;
+--------------------+
| Tables_in_cinemadb |
+--------------------+
| bookings           |
| customer_info      |
| movies             |
| showtimes          |
+--------------------+
4 rows in set (0.00 sec)
```

Add column in

```
mysql> alter table staff add age int(18);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc staff;
+------------+-----------------------------------------------------------------+------+-----+---------+----------------+
| Field      | Type                                                            | Null | Key | Default | Extra          |
+------------+-----------------------------------------------------------------+------+-----+---------+----------------+
| staff_id   | int(11)                                                         | NO   | PRI | NULL    | auto_increment |
| first_name | varchar(50)                                                     | NO   |     | NULL    |                |
| last_name  | varchar(50)                                                     | NO   |     | NULL    |                |
| role       | enum('Manager','Reception','Housekeeping','Security','Kitchen') | NO   |     | NULL    |                |
| phone      | varchar(20)                                                     | YES  | UNI | NULL    |                |
| email      | varchar(100)                                                    | YES  | UNI | NULL    |                |
| hired_on   | date                                                            | NO   |     | NULL    |                |
| age        | int(18)                                                         | YES  |     | NULL    |                |
+------------+-----------------------------------------------------------------+------+-----+---------+----------------+
8 rows in set (0.01 sec)
```

staff

Insert records in tables and display records

guests_details

```
ysql> INSERT INTO guests_details
   -> (first_name, last_name, phone, email, id_doc_type, id_doc_number, nationality)
   -> VALUES
   -> ('Priya', 'Mehta', '9123456780', 'priya.mehta@example.com', 'ID', 'ID98765', 'Indian'),
   -> ('Amit', 'Verma', '9988776655', 'amit.verma@example.com', 'DriverLicense', 'DL445566', 'Indian'),
   -> ('Sneha', 'Kapoor', '9001122334', 'sneha.kapoor@example.com', 'Passport', 'P7654321', 'Indian');
uery OK, 3 rows affected (0.04 sec)
ecords: 3  Duplicates: 0  Warnings: 0

ysql> select*from guests_details;
+----------+------------+-----------+------------+-------------------------+--------------+---------------+---------------------+-------------+
| guest_id | first_name | last_name | phone      | email                   | id_doc_type  | id_doc_number | created_at          | nationality |
+----------+------------+-----------+------------+-------------------------+--------------+---------------+---------------------+-------------+
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID           | ID98765       | 2025-12-11 21:48:01 | Indian      |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com  | DriverLicense | DL445566      | 2025-12-11 21:48:01 | Indian      |
|        3 | Sneha      | Kapoor    | 9001122334 | sneha.kapoor@example.com | Passport     | P7654321      | 2025-12-11 21:48:01 | Indian      |
+----------+------------+-----------+------------+-------------------------+--------------+---------------+---------------------+-------------+
rows in set (0.04 sec)
```

staff

```
mysql> INSERT INTO staff
    -> (first_name, last_name, role, phone, email, hired_on, age)
    -> VALUES
    -> ('Priya', 'Sharma', 'Reception', '9123456780', 'priya.sharma@hotelmg.com', '2024-01-15', 28),
    -> ('Amit', 'Verma', 'Housekeeping', '9988776655', 'amit.verma@hotelmg.com', '2022-09-10', 32),
    -> ('Sneha', 'Kapoor', 'Kitchen', '9001122334', 'sneha.kapoor@hotelmg.com', '2025-03-20', 26),
    -> ('Rajesh', 'Singh', 'Security', '9112233445', 'rajesh.singh@hotelmg.com', '2021-11-05', 40);
Query OK, 4 rows affected (0.04 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select*from staff;
+----------+------------+-----------+--------------+------------+--------------------------+------------+------+
| staff_id | first_name | last_name | role         | phone      | email                    | hired_on   | age  |
+----------+------------+-----------+--------------+------------+--------------------------+------------+------+
|        1 | Priya      | Sharma    | Reception    | 9123456780 | priya.sharma@hotelmg.com | 2024-01-15 |   28 |
|        2 | Amit       | Verma     | Housekeeping | 9988776655 | amit.verma@hotelmg.com   | 2022-09-10 |   32 |
|        3 | Sneha      | Kapoor    | Kitchen      | 9001122334 | sneha.kapoor@hotelmg.com | 2025-03-20 |   26 |
|        4 | Rajesh     | Singh     | Security     | 9112233445 | rajesh.singh@hotelmg.com | 2021-11-05 |   40 |
+----------+------------+-----------+--------------+------------+--------------------------+------------+------+
4 rows in set (0.00 sec)
```

Services

```
mysql> INSERT INTO services (name, unit_price, active)
    -> VALUES
    -> ('Laundry', 300.00, 1),
    -> ('Spa Treatment', 1500.00, 1),
    -> ('Airport Pickup', 1200.00, 1),
    -> ('Breakfast Buffet', 800.00, 1),
    -> ('Gym Access', 400.00, 1),
    -> ('Mini Bar', 600.00, 0);  -- inactive service
Query OK, 6 rows affected (0.05 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql> select*from services;
+------------+------------------+------------+--------+
| service_id | name             | unit_price | active |
+------------+------------------+------------+--------+
|          1 | Laundry          |     300.00 |      1 |
|          2 | Spa Treatment    |    1500.00 |      1 |
|          3 | Airport Pickup   |    1200.00 |      1 |
|          4 | Breakfast Buffet |     800.00 |      1 |
|          5 | Gym Access       |     400.00 |      1 |
|          6 | Mini Bar         |     600.00 |      0 |
+------------+------------------+------------+--------+
6 rows in set (0.00 sec)
```

rooms

```
mysql> INSERT INTO rooms (room_number, floor, type, status)
    -> VALUES ('101', 1, 'Single', 'Available');
Query OK, 1 row affected (0.04 sec)
```

```
mysql>  INSERT INTO rooms (room_number, floor, type, status)VALUES ('102', 2, 'Double', 'Available');
Query OK, 1 row affected (0.05 sec)
```

```
mysql>   INSERT INTO rooms (room_number, floor, type, status)VALUES('104', 3, 'Suite', 'Available');
Query OK, 1 row affected (0.05 sec)

mysql>  INSERT INTO rooms (room_number, floor, type, status)VALUES('105', 4, 'Single', 'Maintenance'),('106', 4, 'Double', 'Occupied');
Query OK, 2 rows affected (0.04 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select*from rooms;
+---------+-------------+-------+--------+-------------+
| room_id | room_number | floor | type   | status      |
+---------+-------------+-------+--------+-------------+
|       1 | 101         |     1 | Single | Available   |
|       2 | 102         |     2 | Double | Available   |
|       3 | 104         |     3 | Suite  | Available   |
|       4 | 105         |     4 | Single | Maintenance |
|       5 | 106         |     4 | Double | Occupied    |
+---------+-------------+-------+--------+-------------+
5 rows in set (0.01 sec)
```

```
mysql> INSERT INTO room_rates (room_type, season, base_rate)
    -> VALUES
    -> ('Single', 'Low', 1500.00),
    -> ('Single', 'High', 2500.00),
    -> ('Double', 'Mid', 3000.00),
    -> ('Deluxe', 'Peak', 5500.00),
    -> ('Suite', 'Peak', 8000.00);
Query OK, 5 rows affected (0.04 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> select*from room_rates;
+---------+-----------+--------+-----------+
| rate_id | room_type | season | base_rate |
+---------+-----------+--------+-----------+
|       1 | Single    | Low    |   1500.00 |
|       2 | Single    | High   |   2500.00 |
|       3 | Double    | Mid    |   3000.00 |
|       4 | Deluxe    | Peak   |   5500.00 |
|       5 | Suite     | Peak   |   8000.00 |
+---------+-----------+--------+-----------+
5 rows in set (0.00 sec)
```

room_rates

audit_bookings

```
mysql> INSERT INTO audit_bookings (booking_id, old_status, new_status)
    -> VALUES
    -> (1, 'Reserved', 'CheckedIn'),
    -> (2, 'CheckedIn', 'CheckedOut'),
    -> (3, 'Reserved', 'Cancelled');
Query OK, 3 rows affected (0.04 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select*from audit_bookings;
+----------+------------+------------+------------+---------------------+
| audit_id | booking_id | old_status | new_status | changed_at          |
+----------+------------+------------+------------+---------------------+
|        1 |          1 | Reserved   | CheckedIn  | 2025-12-11 22:35:12 |
|        2 |          2 | CheckedIn  | CheckedOut | 2025-12-11 22:35:12 |
|        3 |          3 | Reserved   | Cancelled  | 2025-12-11 22:35:12 |
+----------+------------+------------+------------+---------------------+
3 rows in set (0.00 sec)
```

bookings

```
mysql> INSERT INTO bookings
    -> (guest_id, room_id, check_in, check_out, adults, children, status)
    -> VALUES
    -> (1, 1, '2025-12-12', '2025-12-15', 2, 1, 'Reserved'),
    -> (2, 2, '2025-12-13', '2025-12-16', 1, 0, 'CheckedIn'),
    -> (3, 3, '2025-12-14', '2025-12-18', 2, 2, 'Reserved');
Query OK, 3 rows affected (0.02 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select*from bookings;
+------------+----------+---------+------------+------------+--------+----------+-----------+---------------------+
| booking_id | guest_id | room_id | check_in   | check_out  | adults | children | status    | created_at          |
+------------+----------+---------+------------+------------+--------+----------+-----------+---------------------+
|         14 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved  | 2025-12-11 23:07:30 |
|         15 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:07:30 |
|         16 |        3 |       3 | 2025-12-14 | 2025-12-18 |      2 |        2 | Reserved  | 2025-12-11 23:07:30 |
+------------+----------+---------+------------+------------+--------+----------+-----------+---------------------+
3 rows in set (0.00 sec)
```

Update in table room

```
mysql> UPDATE rooms
    -> SET floor = 2
    -> WHERE room_number = '202';
Query OK, 0 rows affected (0.01 sec)
Rows matched: 0  Changed: 0  Warnings: 0

mysql> select*from rooms;
+---------+-------------+-------+--------+-------------+
| room_id | room_number | floor | type   | status      |
+---------+-------------+-------+--------+-------------+
|       1 | 101         |     1 | Single | Available   |
|       2 | 102         |     2 | Double | Available   |
|       3 | 104         |     3 | Suite  | Available   |
|       4 | 105         |     4 | Single | Maintenance |
|       5 | 106         |     4 | Double | Occupied    |
+---------+-------------+-------+--------+-------------+
5 rows in set (0.00 sec)

mysql> UPDATE rooms
    -> SET status = 'Occupied'
    -> WHERE room_id = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
mysql> SELECT * FROM rooms WHERE room_id = 1;
+---------+-------------+-------+--------+----------+
| room_id | room_number | floor | type   | status   |
+---------+-------------+-------+--------+----------+
|       1 | 101         |     1 | Single | Occupied |
+---------+-------------+-------+--------+----------+
1 row in set (0.00 sec)

mysql> UPDATE rooms
    -> SET type = 'Deluxe',
    ->     status = 'Reserved'
    -> WHERE room_id = 3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select*from rooms;
+---------+-------------+-------+--------+-------------+
| room_id | room_number | floor | type   | status      |
+---------+-------------+-------+--------+-------------+
|       1 | 101         |     1 | Single | Occupied    |
|       2 | 102         |     2 | Double | Available   |
|       3 | 104         |     3 | Deluxe | Reserved    |
|       4 | 105         |     4 | Single | Maintenance |
|       5 | 106         |     4 | Double | Occupied    |
+---------+-------------+-------+--------+-------------+
5 rows in set (0.00 sec)
```

Delete from guests_details where guests_id=3T

```
mysql>
mysql> DELETE FROM guests_details
    -> WHERE guest_id = 3;
Query OK, 1 row affected (0.00 sec)

mysql> DELETE FROM guests_details
    -> WHERE guest_id = 3;
Query OK, 0 rows affected (0.01 sec)

mysql> select*from guests_details;
+----------+------------+-----------+------------+-----------------------+--------------+--------------+---------------------+-------------+
| guest_id | first_name | last_name | phone      | email                 | id_doc_type  | id_doc_number | created_at          | nationality |
+----------+------------+-----------+------------+-----------------------+--------------+--------------+---------------------+-------------+
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID           | ID98765      | 2025-12-11 21:48:01 | Indian      |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com  | DriverLicense | DL445566     | 2025-12-11 21:48:01 | Indian      |
+----------+------------+-----------+------------+-----------------------+--------------+--------------+---------------------+-------------+
2 rows in set (0.00 sec)
```

runcate audit bookings table

```
mysql> select*from audit_bookings;
+----------+------------+------------+------------+---------------------+
| audit_id | booking_id | old_status | new_status | changed_at          |
+----------+------------+------------+------------+---------------------+
|        1 |          1 | Reserved   | CheckedIn  | 2025-12-11 22:35:12 |
|        2 |          2 | CheckedIn  | CheckedOut | 2025-12-11 22:35:12 |
|        3 |          3 | Reserved   | Cancelled  | 2025-12-11 22:35:12 |
+----------+------------+------------+------------+---------------------+
3 rows in set (0.00 sec)

mysql> truncate table audit_bookings;
Query OK, 0 rows affected (0.01 sec)

mysql> select*from audit_bookinngs;
ERROR 1146 (42S02): Table 'hotelmg.audit_bookinngs' doesn't exist
mysql> select*from audit_bookings;
Empty set (0.00 sec)
```

Drop theater table

```
ysql> drop table audit_bookings;
uery OK, 0 rows affected (0.02 sec)

ysql> select*from audit_bookings;
RROR 1146 (42S02): Table 'hotelmg.audit_bookings' doesn't exist
```

EXPERIMENT-2

MySQL Functions

In MySQL, functions are built-in operations that perform calculations, data conversions, or manipulations on values stored in the database.

Functions help to:

- Simplify queries
- Perform calculations
- Work with strings, numbers, and dates easily

Types of MySQL Functions

1. **String Functions**- Used to manipulate and work with text (string) data.

Commands:

- **CONCAT** (str 1, str 2, ….) – Join two or more strings.
- **LENGTH(**str) – Returns the length of a string.
- **UPPER(**str)/Lower(str) – convert text to uppercase/lowercase.
- **SUBSTRING(**str, start, length)- Extracts part of a string.
- **TRIM**(str)- Remove spaces from both ends.
- **REPLACE**(str, from_str, to_str) – Replaces part of string.

2. **Numeric Functions**- Used for mathematical calculations.

Commands:

- **ROUND** (x,d)- Rounds to d decimal places.
- **MOD** (x,y)- Returns Remainder
- **RAND**()-Generates Random number(0-1)

3. **Date and Time functions**- Used to handle and format date/time values.

Commands:

- **NOW()-** Returns Current Date and time.
- **CURDATE()-**Returns Current Date
- **CURTIME()-Returns Current time**
- **DATEDIFF(**Date1 , Date2)- Return difference in days

4. **Aggregate functions-**

Aggregate functions perform **a** calculation **on a set (**group**)** of values and return a single **value** as a result.

Commands:

- **COUNT()-** Count the number of rows
- **SUM()-**Adds up all the numeric values
- **AVG()-**Returns the average of numeric values
- **MIN()-**Find the smallest value in the column
- **MAX()-**Find the largest value in the column

Show Database and use it

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| hotel              |
| hoteldb            |
| hotelmanagement    |
| hotelmg            |
| mysql              |
| performance_schema |
| test               |
+--------------------+
8 rows in set (0.01 sec)

mysql> use hotelmg;
Database changed
mysql> show tables;
+------------------+
| Tables_in_hotelmg |
+------------------+
| booking_services  |
| bookings          |
| guests_details    |
| payments          |
| room_rates        |
| rooms             |
| services          |
| staff             |
+------------------+
8 rows in set (0.00 sec)
```

Tables in database hotelmg

```
mysql> select*from rooms;
+---------+-------------+-------+--------+-------------+
| room_id | room_number | floor | type   | status      |
+---------+-------------+-------+--------+-------------+
|       1 | 101         |     1 | Single | Occupied    |
|       2 | 102         |     2 | Double | Available   |
|       3 | 104         |     3 | Deluxe | Reserved    |
|       4 | 105         |     4 | Single | Maintenance |
|       5 | 106         |     4 | Double | Occupied    |
+---------+-------------+-------+--------+-------------+
5 rows in set (0.00 sec)
```

```
mysql> select*from room_rates;
+---------+-----------+--------+-----------+
| rate_id | room_type | season | base_rate |
+---------+-----------+--------+-----------+
|       1 | Single    | Low    |   1500.00 |
|       2 | Single    | High   |   2500.00 |
|       3 | Double    | Mid    |   3000.00 |
|       4 | Deluxe    | Peak   |   5500.00 |
|       5 | Suite     | Peak   |   8000.00 |
+---------+-----------+--------+-----------+
5 rows in set (0.00 sec)

mysql> select*from guests_details;
+----------+------------+-----------+------------+-----------------------+-------------+---------------+---------------------+-------------+
| guest_id | first_name | last_name | phone      | email                 | id_doc_type | id_doc_number | created_at          | nationality |
+----------+------------+-----------+------------+-----------------------+-------------+---------------+---------------------+-------------+
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID          | ID98765       | 2025-12-11 21:48:01 | Indian      |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com | DriverLicense | DL445566    | 2025-12-11 21:48:01 | Indian      |
+----------+------------+-----------+------------+-----------------------+-------------+---------------+---------------------+-------------+
2 rows in set (0.00 sec)
```

Select Avg (total_amount) from bookings table

```
mysql> SELECT AVG(base_rate) AS avg_room_rate
    -> FROM room_rates;
+---------------+
| avg_room_rate |
+---------------+
|   4100.000000 |
+---------------+
1 row in set (0.05 sec)
```

Select concat(title,'from',genre) from movies table as moviesform

```
ysql> SELECT CONCAT(room_type, ' - ', season) AS rate_plan
   -> FROM room_rates;
+--------------+
| rate_plan    |
+--------------+
| Single - Low |
| Single - High |
| Double - Mid |
| Deluxe - Peak |
| Suite - Peak |
+--------------+
5 rows in set (0.02 sec)
```

Select count(*) as total rates from room rates

```
mysql> SELECT COUNT(*) AS total_rates
    -> FROM room_rates;
+-------------+
| total_rates |
+-------------+
|           5 |
+-------------+
1 row in set (0.00 sec)
```

SELECT COUNT(*) AS total guests FROM guests details;

```
mysql> SELECT COUNT(*) AS total_guests
    -> FROM guests_details;
+--------------+
| total_guests |
+--------------+
|            2 |
+--------------+
1 row in set (0.00 sec)
```

SELECT first_name, LENGTH(first_name) AS name_length FROM guests_details

```
mysql> SELECT first_name, LENGTH(first_name) AS name_length
    -> FROM guests_details;
+------------+-------------+
| first_name | name_length |
+------------+-------------+
| Priya      |           5 |
| Amit       |           4 |
+------------+-------------+
2 rows in set (0.04 sec)
```

SELECT MAX(base_rate) AS highest_rate FROM room_rates;

```
mysql> SELECT MAX(base_rate) AS highest_rate
    -> FROM room_rates;
+--------------+
| highest_rate |
+--------------+
|      8000.00 |
+--------------+
1 row in set (0.00 sec)
```

SELECT MIN(base_rate) AS lowest_rate FROM room_rates;

```
mysql> SELECT MIN(base_rate) AS lowest_rate
    -> FROM room_rates;
+-------------+
| lowest_rate |
+-------------+
|     1500.00 |
+-------------+
1 row in set (0.00 sec)
```

mysql> SELECT b.guest_id, MOD(p.amount, 100) AS remainder FROM payments p JOIN bookings b ON p.booking_id = b.booking_id;

```
mysql> SELECT b.guest_id, MOD(p.amount, 100) AS remainder
    -> FROM payments p
    -> JOIN bookings b ON p.booking_id = b.booking_id;
+----------+-----------+
| guest_id | remainder |
+----------+-----------+
|        1 |      0.00 |
|        2 |      0.00 |
|        1 |      0.00 |
|        1 |      0.00 |
|        2 |      0.00 |
+----------+-----------+
5 rows in set (0.00 sec)
```

SELECT CONCAT(first_name, ' ', last_name) AS name, RAND() AS randomvalue FROM guests_details;

```
mysql> SELECT CONCAT(first_name, ' ', last_name) AS name,
    ->        RAND() AS randomvalue
    -> FROM guests_details;
+-------------+--------------------+
| name        | randomvalue        |
+-------------+--------------------+
| Priya Mehta | 0.5549710658890857 |
| Amit Verma  | 0.7224879727908298 |
+-------------+--------------------+
2 rows in set (0.04 sec)
```

SELECT CONCAT

```
mysql> SELECT CONCAT(first_name, ' ', last_name) AS name,
    ->        REPLACE(CONCAT(first_name, ' ', last_name), 'a
', '#') AS hashedname
    -> FROM guests_details;
+-------------+-------------+
| name        | hashedname  |
+-------------+-------------+
| Priya Mehta | Priy# Meht# |
| Amit Verma  | Amit Verm#  |
+-------------+-------------+
2 rows in set (0.04 sec)
```

SELECT booking_id, ROUND(amount, 2) AS roundamount FROM payments

```
mysql> SELECT booking_id, ROUND(amount, 2) AS roundamount
    -> FROM payments;
+------------+-------------+
| booking_id | roundamount |
+------------+-------------+
|         14 |     4500.00 |
|         15 |     7200.00 |
|         17 |     3500.00 |
|         18 |     9800.00 |
|         19 |     2500.00 |
+------------+-------------+
5 rows in set (0.00 sec)
```

SELECT CONCAT(first_name, ' ', last_name) AS name, SUBSTRING(CONCAT(first_name, ' ', last_name), 1, 5) AS nameshort FROM guests_details;

```
mysql> SELECT CONCAT(first_name, ' ', last_name) AS name,
    ->        SUBSTRING(CONCAT(first_name, ' ', last_name),
1, 5) AS nameshort
    -> FROM guests_details;
+-------------+-----------+
| name        | nameshort |
+-------------+-----------+
| Priya Mehta | Priya     |
| Amit Verma  | Amit      |
+-------------+-----------+
2 rows in set (0.00 sec)
```

SELECT SUM(total_amount) AS total_revenue FROM bookings;

```
mysql> SELECT SUM(total_amount) AS total_revenue
    -> FROM bookings;
+---------------+
| total_revenue |
+---------------+
|      27500.00 |
+---------------+
1 row in set (0.00 sec)
```

SELECT UPPER(CONCAT(first_name, ' ', last_name)) AS name_upperFROM guests_details;

```
mysql> SELECT UPPER(CONCAT(first_name, ' ', last_name)) AS n
ame_upper
    -> FROM guests_details;
+-------------+
| name_upper  |
+-------------+
| PRIYA MEHTA |
| AMIT VERMA  |
+-------------+
2 rows in set (0.00 sec)
```

SELECT LOWER(CONCAT(first_name, ' ', last_name)) AS name_lower FROM guests_details;

```
mysql> SELECT LOWER(CONCAT(first_name, ' ', last_name)) AS n
ame_lower
    -> FROM guests_details;
+-------------+
| name_lower  |
+-------------+
| priya mehta |
| amit verma  |
+-------------+
2 rows in set (0.00 sec)
```

SELECT CURDATE() AS today_date;

```
mysql> SELECT CURDATE() AS today_date;
+------------+
| today_date |
+------------+
| 2025-12-12 |
+------------+
1 row in set (0.04 sec)
```

Select costumer_info,booking_date,dayname(booking_date) as bookingday, and (curdate(),booking_date) as daysinroder from bookings table

```
mysql> SELECT b.booking_id,
    ->        b.check_in AS booking_date,
    ->        DAYNAME(b.check_in) AS bookingday,
    ->        DATEDIFF(CURDATE(), b.check_in) AS daysinorder

    -> FROM bookings b;
+------------+--------------+------------+-------------+
| booking_id | booking_date | bookingday | daysinorder |
+------------+--------------+------------+-------------+
|         14 | 2025-12-12   | Friday     |           0 |
|         15 | 2025-12-13   | Saturday   |          -1 |
|         17 | 2025-12-12   | Friday     |           0 |
|         18 | 2025-12-12   | Friday     |           0 |
|         19 | 2025-12-13   | Saturday   |          -1 |
|         21 | 2025-12-12   | Friday     |           0 |
|         22 | 2025-12-13   | Saturday   |          -1 |
|         24 | 2025-12-12   | Friday     |           0 |
|         25 | 2025-12-13   | Saturday   |          -1 |
+------------+--------------+------------+-------------+
9 rows in set (0.05 sec)
```

Select current date and time

```
mysql> select now()as currentdatetime;
+---------------------+
| currentdatetime     |
+---------------------+
| 2025-12-12 13:17:55 |
+---------------------+
1 row in set (0.04 sec)
```

EXPERIMENT 3

Clauses in MySQL-

A **clause** in SQL is a **part of a query** that performs a specific task like selecting data, filtering data, sorting data, grouping data, etc.

1. SELECT Clause

- It tells MySQL what data (columns) you want to see.

2. FROM Clause

- Used to specify the table from which the data should be selected or retrieved.

3. WHERE Clause

- Used to apply conditions and filter rows based on specific criteria.

4. ORDER BY Clause

- Used to arrange or sort the result set in ascending or descending order.

5. GROUP BY Clause

- Used to group rows that have similar values, often used with aggregate functions such as COUNT, SUM, AVG, MAX, and MIN.

6. HAVING Clause

- Used to apply conditions on groups formed by the GROUP BY clause. It filters grouped data.

7. LIMIT Clause

- Used to restrict the number of rows returned in the query result.

8. DISTINCT Clause

- Used to remove duplicate values and display only unique entries from a column.

9. JOIN Clause

- Used to combine and retrieve related data from two or more tables based on a common field.

Show databases

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| hotel              |
| hoteldb            |
| hotelmanagement    |
| hotelmg            |
| mysql              |
| performance_schema |
| test               |
+--------------------+
8 rows in set (0.04 sec)
```

Use database and show tables

```
mysql> use hotelmg;
Database changed
mysql> show tables;
+--------------------+
| Tables_in_hotelmg  |
+--------------------+
| booking_services   |
| bookings           |
| guests_details     |
| payments           |
| room_rates         |
| rooms              |
| services           |
| staff              |
+--------------------+
8 rows in set (0.00 sec)
```

Tables in Database hotelmg

```
ysql> select*from bookings;
+------------+----------+---------+------------+-----------+--------+----------+----------+---------------------+--------------+
| booking_id | guest_id | room_id | check_in   | check_out | adults | children | status   | created_at          | total_amount |
+------------+----------+---------+------------+-----------+--------+----------+----------+---------------------+--------------+
|         14 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved | 2025-12-11 23:07:30 |      4500.00 |
|         15 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:07:30 |      7200.00 |
|         17 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved | 2025-12-11 23:09:50 |      3500.00 |
|         18 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved | 2025-12-11 23:11:01 |      9800.00 |
|         19 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:11:01 |      2500.00 |
|         21 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved | 2025-12-11 23:14:05 |         NULL |
|         22 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:14:05 |         NULL |
|         24 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved | 2025-12-11 23:14:48 |         NULL |
|         25 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:14:48 |         NULL |
+------------+----------+---------+------------+-----------+--------+----------+----------+---------------------+--------------+
 rows in set (0.00 sec)

ysql> select*from guests_details;
+----------+------------+-----------+------------+-----------------------+--------------+----------------+---------------------+-------------+
| guest_id | first_name | last_name | phone      | email                 | id_doc_type  | id_doc_number  | created_at          | nationality |
+----------+------------+-----------+------------+-----------------------+--------------+----------------+---------------------+-------------+
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID           | ID98765        | 2025-12-11 21:48:01 | Indian      |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com | DriverLicense | DL445566       | 2025-12-11 21:48:01 | Indian      |
+----------+------------+-----------+------------+-----------------------+--------------+----------------+---------------------+-------------+
 rows in set (0.00 sec)

ysql> select*from payments;
+------------+------------+---------+---------------+---------------------+--------------------+
| payment_id | booking_id | amount  | method        | paid_at             | note               |
+------------+------------+---------+---------------+---------------------+--------------------+
|         27 |         14 | 4500.00 | Cash          | 2025-12-12 12:56:10 | Advance payment    |
|         28 |         15 | 7200.00 | Card          | 2025-12-12 12:56:10 | Full payment       |
|         29 |         17 | 3500.00 | UPI           | 2025-12-12 12:56:10 | Partial payment    |
|         30 |         18 | 9800.00 | BankTransfer  | 2025-12-12 12:56:10 | Corporate booking  |
|         31 |         19 | 2500.00 | Cash          | 2025-12-12 12:56:10 | Balance cleared    |
+------------+------------+---------+---------------+---------------------+--------------------+
 rows in set (0.00 sec)

ysql> select*from room_rates;
+---------+-----------+--------+-----------+
| rate_id | room_type | season | base_rate |
+---------+-----------+--------+-----------+
|       1 | Single    | Low    |   1500.00 |
|       2 | Single    | High   |   2500.00 |
|       3 | Double    | Mid    |   3000.00 |
|       4 | Deluxe    | Peak   |   5500.00 |
|       5 | Suite     | Peak   |   8000.00 |
+---------+-----------+--------+-----------+
 rows in set (0.00 sec)

ysql> select*from rooms;
+---------+-------------+-------+--------+-------------+
| room_id | room_number | floor | type   | status      |
+---------+-------------+-------+--------+-------------+
|       1 | 101         |     1 | Single | Occupied    |
|       2 | 102         |     2 | Double | Available   |
|       3 | 104         |     3 | Deluxe | Reserved    |
|       4 | 105         |     4 | Single | Maintenance |
|       5 | 106         |     4 | Double | Occupied    |
+---------+-------------+-------+--------+-------------+
 rows in set (0.00 sec)

ysql> select*from services;
+------------+-----------------+------------+--------+
| service_id | name            | unit_price | active |
+------------+-----------------+------------+--------+
|          1 | Laundry         |     300.00 |      1 |
|          2 | Spa Treatment   |    1500.00 |      1 |
|          3 | Airport Pickup  |    1200.00 |      1 |
|          4 | Breakfast Buffet|     800.00 |      1 |
|          5 | Gym Access      |     400.00 |      1 |
|          6 | Mini Bar        |     600.00 |      0 |
+------------+-----------------+------------+--------+
 rows in set (0.00 sec)

ysql> select*from staff;
+----------+------------+-----------+-------------+------------+-----------------------+------------+-----+
| staff_id | first_name | last_name | role        | phone      | email                 | hired_on   | age |
+----------+------------+-----------+-------------+------------+-----------------------+------------+-----+
|        1 | Priya      | Sharma    | Reception   | 9123456780 | priya.sharma@hotelmg.com | 2024-01-15 |  28 |
|        2 | Amit       | Verma     | Housekeeping | 9988776655 | amit.verma@hotelmg.com | 2022-09-10 |  32 |
|        3 | Sneha      | Kapoor    | Kitchen     | 9001122334 | sneha.kapoor@hotelmg.com | 2025-03-20 |  26 |
|        4 | Rajesh     | Singh     | Security    | 9112233445 | rajesh.singh@hotelmg.com | 2021-11-05 |  40 |
+----------+------------+-----------+-------------+------------+-----------------------+------------+-----+
 rows in set (0.00 sec)
```

Select distinct age from staff;

```
mysql> select distinct age from staff;
+------+
| age  |
+------+
|   28 |
|   32 |
|   26 |
|   40 |
+------+
4 rows in set (0.00 sec)
```

Select customer_id , avg(total_amount) group by customer_info;

```
mysql> SELECT CONCAT(g.first_name, ' ', g.last_name) AS guest_name,
    ->        AVG(b.total_amount) AS avg_amount
    -> FROM bookings b
    -> JOIN guests_details g ON b.guest_id = g.guest_id
    -> GROUP BY g.first_name, g.last_name;
+-------------+-------------+
| guest_name  | avg_amount  |
+-------------+-------------+
| Amit Verma  | 4850.000000 |
| Priya Mehta | 5933.333333 |
+-------------+-------------+
2 rows in set (0.00 sec)
```

Select customer_id, count from customer_info group by customer_id;

```
mysql> SELECT b.guest_id,
    ->        AVG(b.total_amount) AS avg_amount
    -> FROM bookings b
    -> GROUP BY b.guest_id;
+----------+-------------+
| guest_id | avg_amount  |
+----------+-------------+
|        1 | 5933.333333 |
|        2 | 4850.000000 |
+----------+-------------+
2 rows in set (0.00 sec)
```

Select all details from customer_info order by age in ascending order;

```
mysql> SELECT *
    -> FROM guests_details
    -> ORDER BY age ASC;
+----------+------------+-----------+------------+-------------------------+--------------+----------------+---------------------+-------------+-----+
| guest_id | first_name | last_name | phone      | email                   | id_doc_type  | id_doc_number  | created_at          | nationality | age |
+----------+------------+-----------+------------+-------------------------+--------------+----------------+---------------------+-------------+-----+
|       14 | Ananya     | Gupta     | 9876501234 | ananya.gupta@example.com | Passport     | P778899        | 2025-12-12 13:45:14 | Indian      |  26 |
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com  | ID           | ID98765        | 2025-12-11 21:48:01 | Indian      |  28 |
|       16 | Neha       | Joshi     | 9988771122 | neha.joshi@example.com   | DriverLicense | DL223344       | 2025-12-12 13:45:14 | Indian      |  29 |
|       18 | Pooja      | Desai     | 9099112233 | pooja.desai@example.com  | Aadhar       | A667788        | 2025-12-12 13:45:14 | Indian      |  31 |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com   | DriverLicense | DL445566       | 2025-12-11 21:48:01 | Indian      |  32 |
|       15 | Vikram     | Rao       | 9123456701 | vikram.rao@example.com   | Aadhar       | A445566        | 2025-12-12 13:45:14 | Indian      |  34 |
|       17 | Suresh     | Iyer      | 9011225566 | suresh.iyer@example.com  | Passport     | P998877        | 2025-12-12 13:45:14 | Indian      |  40 |
+----------+------------+-----------+------------+-------------------------+--------------+----------------+---------------------+-------------+-----+
7 rows in set (0.00 sec)
```

Select all details from customer info order by age in descending order;

```
mysql> SELECT *
    -> FROM guests_details
    -> ORDER BY age DESC;
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
| guest_id | first_name | last_name | phone      | email                   | id_doc_type   | id_doc_number  | created_at          | nationality | age  |
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
|       17 | Suresh     | Iyer      | 9011225566 | suresh.iyer@example.com | Passport      | P998877        | 2025-12-12 13:45:14 | Indian      |   40 |
|       15 | Vikram     | Rao       | 9123456701 | vikram.rao@example.com  | Aadhar        | A445566        | 2025-12-12 13:45:14 | Indian      |   34 |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com  | DriverLicense | DL445566       | 2025-12-11 21:48:01 | Indian      |   32 |
|       18 | Pooja      | Desai     | 9099112233 | pooja.desai@example.com | Aadhar        | A667788        | 2025-12-12 13:45:14 | Indian      |   31 |
|       16 | Neha       | Joshi     | 9988771122 | neha.joshi@example.com  | DriverLicense | DL223344       | 2025-12-12 13:45:14 | Indian      |   29 |
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID            | ID98765        | 2025-12-11 21:48:01 | Indian      |   28 |
|       14 | Ananya     | Gupta     | 9876501234 | ananya.gupta@example.com| Passport      | P778899        | 2025-12-12 13:45:14 | Indian      |   26 |
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
```

Select all details from customer info where age between 23 and 28;

```
mysql> SELECT *
    -> FROM guests_details
    -> WHERE age BETWEEN 23 AND 28;
+----------+------------+-----------+------------+-------------------------+-------------+----------------+---------------------+-------------+------+
| guest_id | first_name | last_name | phone      | email                   | id_doc_type | id_doc_number  | created_at          | nationality | age  |
+----------+------------+-----------+------------+-------------------------+-------------+----------------+---------------------+-------------+------+
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID          | ID98765        | 2025-12-11 21:48:01 | Indian      |   28 |
|       14 | Ananya     | Gupta     | 9876501234 | ananya.gupta@example.com| Passport    | P778899        | 2025-12-12 13:45:14 | Indian      |   26 |
+----------+------------+-----------+------------+-------------------------+-------------+----------------+---------------------+-------------+------+
2 rows in set (0.04 sec)
```

- Select all details from customer info where name like ‘A%’;

```
mysql> SELECT *
    -> FROM guests_details
    -> WHERE first_name LIKE 'A%';
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
| guest_id | first_name | last_name | phone      | email                   | id_doc_type   | id_doc_number  | created_at          | nationality | age  |
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com  | DriverLicense | DL445566       | 2025-12-11 21:48:01 | Indian      |   32 |
|       14 | Ananya     | Gupta     | 9876501234 | ananya.gupta@example.com| Passport      | P778899        | 2025-12-12 13:45:14 | Indian      |   26 |
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
2 rows in set (0.04 sec)
```

-Select all details from customer info where name like ‘%i%;

```
mysql> SELECT *
    -> FROM guests_details
    -> WHERE first_name LIKE '%i%';
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
| guest_id | first_name | last_name | phone      | email                   | id_doc_type   | id_doc_number  | created_at          | nationality | age  |
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
|        1 | Priya      | Mehta     | 9123456780 | priya.mehta@example.com | ID            | ID98765        | 2025-12-11 21:48:01 | Indian      |   28 |
|        2 | Amit       | Verma     | 9988776655 | amit.verma@example.com  | DriverLicense | DL445566       | 2025-12-11 21:48:01 | Indian      |   32 |
|       15 | Vikram     | Rao       | 9123456701 | vikram.rao@example.com  | Aadhar        | A445566        | 2025-12-12 13:45:14 | Indian      |   34 |
+----------+------------+-----------+------------+-------------------------+---------------+----------------+---------------------+-------------+------+
3 rows in set (0.00 sec)
```

Select all movies where genre is not deluxe;

```
mysql> SELECT *
    -> FROM rooms
    -> WHERE type != 'Deluxe';
+---------+-------------+-------+--------+-------------+
| room_id | room_number | floor | type   | status      |
+---------+-------------+-------+--------+-------------+
|       1 | 101         |     1 | Single | Occupied    |
|       2 | 102         |     2 | Double | Available   |
|       4 | 105         |     4 | Single | Maintenance |
|       5 | 106         |     4 | Double | Occupied    |
+---------+-------------+-------+--------+-------------+
4 rows in set (0.01 sec)
```

EXPERIMENT 4

Joins in MySQL-

A **JOIN** in MySQL is used to combine rows from two or more tables based on a related column between them.
It helps retrieve meaningful data spread across multiple tables.

Types of Joins in MySQL

1. INNER JOIN

Returns only the matching rows from both tables.
Non-matching rows are excluded.

2. LEFT JOIN

Returns all rows from the left table and only matching rows from the right table.
Non-matching rows from the right table appear as NULL.

3. RIGHT JOIN

Returns all rows from the right table and only matching rows from the left table.
Non-matching rows from the left table appear as NULL.

4.FULL JOIN

Returns all rows from both tables, matching or not.
Non-matching rows from either table appear as NULL.

MySQL doesn't support full join directly

It simulate a full join using a combination of Left join , right join and union .

5. CROSS JOIN

Returns the **Cartesian product** of two tables.
Every row of the first table is paired with every row of the second table.

6. SELF JOIN

A table is joined with itself.
Used to compare rows within the same table.

7. NATURAL JOIN

Automatically joins tables based on columns with the same name in both tables.
No ON condition is needed.

Join table Showtimes with movies

```
mysql> SELECT g.guest_id, g.first_name, g.last_name, b.booking_id, b.check_in, b.check_out
    -> FROM guests_details g
    -> JOIN bookings b ON g.guest_id = b.guest_id;
+----------+------------+-----------+------------+------------+------------+
| guest_id | first_name | last_name | booking_id | check_in   | check_out  |
+----------+------------+-----------+------------+------------+------------+
|        1 | Priya      | Mehta     |         14 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         15 | 2025-12-13 | 2025-12-16 |
|        1 | Priya      | Mehta     |         17 | 2025-12-12 | 2025-12-15 |
|        1 | Priya      | Mehta     |         18 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         19 | 2025-12-13 | 2025-12-16 |
|        1 | Priya      | Mehta     |         21 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         22 | 2025-12-13 | 2025-12-16 |
|        1 | Priya      | Mehta     |         24 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         25 | 2025-12-13 | 2025-12-16 |
+----------+------------+-----------+------------+------------+------------+
9 rows in set (0.00 sec)
```

Inner join – Customer with their bookings

```
mysql> SELECT g.guest_id, g.first_name, g.last_name, b.booking_id, b.check_in, b.check_out
    -> FROM guests_details g
    -> INNER JOIN bookings b ON g.guest_id = b.guest_id;
+----------+------------+-----------+------------+------------+------------+
| guest_id | first_name | last_name | booking_id | check_in   | check_out  |
+----------+------------+-----------+------------+------------+------------+
|        1 | Priya      | Mehta     |         14 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         15 | 2025-12-13 | 2025-12-16 |
|        1 | Priya      | Mehta     |         17 | 2025-12-12 | 2025-12-15 |
|        1 | Priya      | Mehta     |         18 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         19 | 2025-12-13 | 2025-12-16 |
|        1 | Priya      | Mehta     |         21 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         22 | 2025-12-13 | 2025-12-16 |
|        1 | Priya      | Mehta     |         24 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         25 | 2025-12-13 | 2025-12-16 |
+----------+------------+-----------+------------+------------+------------+
9 rows in set (0.00 sec)
```

Left join – All customer booked or not

```
mysql> SELECT g.guest_id, g.first_name, g.last_name, b.booking_id, b.check_in, b.check_out
    -> FROM guests_details g
    -> LEFT JOIN bookings b ON g.guest_id = b.guest_id;
+----------+------------+-----------+------------+------------+------------+
| guest_id | first_name | last_name | booking_id | check_in   | check_out  |
+----------+------------+-----------+------------+------------+------------+
|        1 | Priya      | Mehta     |         14 | 2025-12-12 | 2025-12-15 |
|        1 | Priya      | Mehta     |         17 | 2025-12-12 | 2025-12-15 |
|        1 | Priya      | Mehta     |         18 | 2025-12-12 | 2025-12-15 |
|        1 | Priya      | Mehta     |         21 | 2025-12-12 | 2025-12-15 |
|        1 | Priya      | Mehta     |         24 | 2025-12-12 | 2025-12-15 |
|        2 | Amit       | Verma     |         15 | 2025-12-13 | 2025-12-16 |
|        2 | Amit       | Verma     |         19 | 2025-12-13 | 2025-12-16 |
|        2 | Amit       | Verma     |         22 | 2025-12-13 | 2025-12-16 |
|        2 | Amit       | Verma     |         25 | 2025-12-13 | 2025-12-16 |
|       14 | Ananya     | Gupta     |       NULL | NULL       | NULL       |
|       15 | Vikram     | Rao       |       NULL | NULL       | NULL       |
|       16 | Neha       | Joshi     |       NULL | NULL       | NULL       |
|       17 | Suresh     | Iyer      |       NULL | NULL       | NULL       |
|       18 | Pooja      | Desai     |       NULL | NULL       | NULL       |
+----------+------------+-----------+------------+------------+------------+
14 rows in set (0.00 sec)
```

Right join – Showtimes with movie titles

```
mysql> SELECT
    ->     b.booking_id,
    ->     b.check_in,
    ->     b.check_out,
    ->     r.room_number,
    ->     r.type,
    ->     r.status
    -> FROM Bookings b
    -> RIGHT JOIN Rooms r
    ->     ON b.room_id = r.room_id;
+------------+------------+------------+-------------+--------+-------------+
| booking_id | check_in   | check_out  | room_number | type   | status      |
+------------+------------+------------+-------------+--------+-------------+
|         14 | 2025-12-12 | 2025-12-15 | 101         | Single | Occupied    |
|         17 | 2025-12-12 | 2025-12-15 | 101         | Single | Occupied    |
|         18 | 2025-12-12 | 2025-12-15 | 101         | Single | Occupied    |
|         21 | 2025-12-12 | 2025-12-15 | 101         | Single | Occupied    |
|         24 | 2025-12-12 | 2025-12-15 | 101         | Single | Occupied    |
|         15 | 2025-12-13 | 2025-12-16 | 102         | Double | Available   |
|         19 | 2025-12-13 | 2025-12-16 | 102         | Double | Available   |
|         22 | 2025-12-13 | 2025-12-16 | 102         | Double | Available   |
|         25 | 2025-12-13 | 2025-12-16 | 102         | Double | Available   |
|       NULL | NULL       | NULL       | 104         | Deluxe | Reserved    |
|       NULL | NULL       | NULL       | 105         | Single | Maintenance |
|       NULL | NULL       | NULL       | 106         | Double | Occupied    |
+------------+------------+------------+-------------+--------+-------------+
12 rows in set (0.00 sec)
```

Right join – Bookings with customer name

```
mysql> SELECT c.name, b.booking_id
    -> FROM customer_info c
    -> RIGHT JOIN bookings b ON c.customer_id = b.customer_id;
+-----------------+------------+
| name            | booking_id |
+-----------------+------------+
| akshita sharma  |          7 |
| bhoomika sharma |          9 |
| riddhi verma    |          6 |
| shaurya sharma  |         10 |
| krish yadav     |          8 |
| vicky yadav     |          4 |
+-----------------+------------+
6 rows in set (0.00 sec)
```

Full join (MySQL alternative using Union)

```
mysql> -- LEFT JOIN: all guests, even if no booking
mysql> SELECT g.guest_id, g.first_name, g.last_name,
    ->        b.booking_id, b.room_number, b.check_in, b.check_out
    -> FROM guests_details g
    -> LEFT JOIN bookings_details b
    -> ON g.guest_id = b.guest_id
    ->
    -> UNION
    ->
    -> -- RIGHT JOIN: all bookings, even if no guest
    -> SELECT g.guest_id, g.first_name, g.last_name,
    ->        b.booking_id, b.room_number, b.check_in, b.check_out
    -> FROM guests_details g
    -> RIGHT JOIN bookings_details b
    -> ON g.guest_id = b.guest_id;
+----------+------------+-----------+------------+-------------+------------+------------+
| guest_id | first_name | last_name | booking_id | room_number | check_in   | check_out  |
+----------+------------+-----------+------------+-------------+------------+------------+
|        1 | Priya      | Mehta     |          1 |         101 | 2025-12-01 | 2025-12-05 |
|        2 | Amit       | Verma     |          2 |         102 | 2025-12-02 | 2025-12-06 |
|       14 | Ananya     | Gupta     |       NULL |        NULL | NULL       | NULL       |
|       15 | Vikram     | Rao       |       NULL |        NULL | NULL       | NULL       |
|       16 | Neha       | Joshi     |       NULL |        NULL | NULL       | NULL       |
|       17 | Suresh     | Iyer      |       NULL |        NULL | NULL       | NULL       |
|       18 | Pooja      | Desai     |       NULL |        NULL | NULL       | NULL       |
|     NULL | NULL       | NULL      |          3 |         103 | 2025-12-03 | 2025-12-07 |
+----------+------------+-----------+------------+-------------+------------+------------+
8 rows in set (0.00 sec)
```

Cross join – Every movie × every showtimes

```
mysql> SELECT g.guest_id, g.first_name, g.last_name,
    ->        r.room_id, r.room_number, r.room_type
    -> FROM guests_details g
    -> CROSS JOIN rooms_details r;
+----------+------------+-----------+---------+-------------+-----------+
| guest_id | first_name | last_name | room_id | room_number | room_type |
+----------+------------+-----------+---------+-------------+-----------+
|        1 | Priya      | Mehta     |       1 |         101 | Deluxe    |
|        1 | Priya      | Mehta     |       2 |         102 | Standard  |
|        1 | Priya      | Mehta     |       3 |         103 | Suite     |
|        2 | Amit       | Verma     |       1 |         101 | Deluxe    |
|        2 | Amit       | Verma     |       2 |         102 | Standard  |
|        2 | Amit       | Verma     |       3 |         103 | Suite     |
|       14 | Ananya     | Gupta     |       1 |         101 | Deluxe    |
|       14 | Ananya     | Gupta     |       2 |         102 | Standard  |
|       14 | Ananya     | Gupta     |       3 |         103 | Suite     |
|       15 | Vikram     | Rao       |       1 |         101 | Deluxe    |
|       15 | Vikram     | Rao       |       2 |         102 | Standard  |
|       15 | Vikram     | Rao       |       3 |         103 | Suite     |
|       16 | Neha       | Joshi     |       1 |         101 | Deluxe    |
|       16 | Neha       | Joshi     |       2 |         102 | Standard  |
|       16 | Neha       | Joshi     |       3 |         103 | Suite     |
|       17 | Suresh     | Iyer      |       1 |         101 | Deluxe    |
|       17 | Suresh     | Iyer      |       2 |         102 | Standard  |
|       17 | Suresh     | Iyer      |       3 |         103 | Suite     |
|       18 | Pooja      | Desai     |       1 |         101 | Deluxe    |
|       18 | Pooja      | Desai     |       2 |         102 | Standard  |
|       18 | Pooja      | Desai     |       3 |         103 | Suite     |
+----------+------------+-----------+---------+-------------+-----------+
21 rows in set (0.00 sec)
```

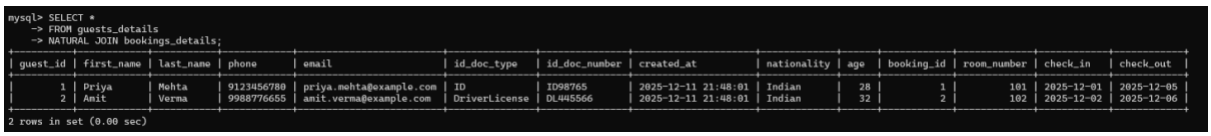
Self join – Using customer_info table

```
mysql> SELECT g1.guest_id, g1.first_name, g1.last_name,
    ->        g2.guest_id AS other_guest_id, g2.first_name AS other_first_name
    -> FROM guests_details g1
    -> JOIN guests_details g2
    -> ON g1.nationality = g2.nationality
    -> AND g1.guest_id <> g2.guest_id;
```

guest_id	first_name	last_name	other_guest_id	other_first_name
2	Amit	Verma	1	Priya
14	Ananya	Gupta	1	Priya
15	Vikram	Rao	1	Priya
16	Neha	Joshi	1	Priya
17	Suresh	Iyer	1	Priya
18	Pooja	Desai	1	Priya
1	Priya	Mehta	2	Amit
14	Ananya	Gupta	2	Amit
15	Vikram	Rao	2	Amit
16	Neha	Joshi	2	Amit
17	Suresh	Iyer	2	Amit
18	Pooja	Desai	2	Amit
1	Priya	Mehta	14	Ananya
2	Amit	Verma	14	Ananya
15	Vikram	Rao	14	Ananya
16	Neha	Joshi	14	Ananya
17	Suresh	Iyer	14	Ananya
18	Pooja	Desai	14	Ananya
1	Priya	Mehta	15	Vikram
2	Amit	Verma	15	Vikram
14	Ananya	Gupta	15	Vikram
16	Neha	Joshi	15	Vikram
17	Suresh	Iyer	15	Vikram
18	Pooja	Desai	15	Vikram
1	Priya	Mehta	16	Neha
2	Amit	Verma	16	Neha
14	Ananya	Gupta	16	Neha
15	Vikram	Rao	16	Neha
17	Suresh	Iyer	16	Neha
18	Pooja	Desai	16	Neha
1	Priya	Mehta	17	Suresh
2	Amit	Verma	17	Suresh
14	Ananya	Gupta	17	Suresh
15	Vikram	Rao	17	Suresh
16	Neha	Joshi	17	Suresh
18	Pooja	Desai	17	Suresh
1	Priya	Mehta	18	Pooja
2	Amit	Verma	18	Pooja
14	Ananya	Gupta	18	Pooja
15	Vikram	Rao	18	Pooja
16	Neha	Joshi	18	Pooja
17	Suresh	Iyer	18	Pooja

42 rows in set (0.02 sec)

Natural joins using showtimes and movies



```
mysql> SELECT *
    -> FROM guests_details
    -> NATURAL JOIN bookings_details;
```

guest_id	first_name	last_name	phone	email	id_doc_type	id_doc_number	created_at	nationality	age	booking_id	room_number	check_in	check_out
1	Priya	Mehta	9123456780	priya.mehta@example.com	ID	ID98765	2025-12-11 21:48:01	Indian	28	1	101	2025-12-01	2025-12-05
2	Amit	Verma	9988776655	amit.verma@example.com	DriverLicense	DL4455566	2025-12-11 21:48:01	Indian	32	2	102	2025-12-02	2025-12-06

2 rows in set (0.00 sec)

EXPERIMENT 5

NESTED QUERIES –

A **nested query** or **subquery** is a query written inside another SQL query.
The inner query runs first, and its result is used by the outer query.
Nested queries help in performing complex operations by breaking them into small steps.

Subqueries can be used in **SELECT**, **WHERE**, and **FROM** clauses.
They are mainly used for filtering data, comparing values, checking conditions, and working with data from multiple tables.

Types of nested queries:

1. **IN** – checks if a value matches any value returned by the subquery.
2. **NOT IN** – checks if a value does not match any returned value
3. **EXISTS** – returns true if the subquery gives at least one row.
4. **NOT EXISTS** – returns true if the subquery gives no rows.
5. **Subquery in SELECT** – used for calculated or aggregated values.

IN:-

```
mysql> SELECT first_name, last_name
    -> FROM guests_details
    -> WHERE guest_id IN (
    ->     SELECT guest_id
    ->     FROM bookings_details
    -> );
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| Priya      | Mehta     |
| Amit       | Verma     |
+------------+-----------+
2 rows in set (0.04 sec)
```

Not in:-

```
mysql> SELECT first_name, last_name
    -> FROM guests_details
    -> WHERE guest_id NOT IN (
    ->     SELECT guest_id
    ->     FROM bookings_details
    ->     WHERE guest_id IS NOT NULL
    -> );
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| Ananya     | Gupta     |
| Vikram     | Rao       |
| Neha       | Joshi     |
| Suresh     | Iyer      |
| Pooja      | Desai     |
+------------+-----------+
5 rows in set (0.00 sec)
```

EXISTS:-

```
mysql> SELECT first_name, last_name
    -> FROM guests_details g
    -> WHERE EXISTS (
    ->     SELECT 1
    ->     FROM bookings_details b
    ->     WHERE b.guest_id = g.guest_id
    -> );
+------------+-----------+
| first_name | last_name |
+------------+-----------+
| Priya      | Mehta     |
| Amit       | Verma     |
+------------+-----------+
2 rows in set (0.00 sec)
```

NOT EXISTS:-

```
mysql> SELECT g.GuestID, g.GuestName
    -> FROM guests g
    -> WHERE NOT EXISTS (
    ->     SELECT 1
    ->     FROM bookings b
    ->     WHERE b.GuestID = g.GuestID
    -> );
+---------+------------+
| GuestID | GuestName  |
+---------+------------+
|       1 | Test Guest |
+---------+------------+
1 row in set (0.01 sec)
```

EXPERIMENT – 6

Set Oriented Operations –

Set-oriented operations in DBMS are used to combine or compare **two relations (tables)**. These operations work on **sets of tuples** and return a new set as output.

1. UNION

- Combines the tuples of two relations.
- Removes duplicate records.
- Both relations must have **same number of columns** and **same data types**.

2. DIFFERENCE (MINUS)

- Returns tuples that appear in **first relation but not in the second**.
- Used to find **unique records** from one table compared to another.
- Tables must be **union-compatible**.

3. INTERSECTION

- Returns only those tuples that are **common to both relations**.
- Helpful to find **matching records** between two sets.
- Both tables must have same structure.

4. DIVISION

- Used when one relation contains **pairs** and another contains **single values**, and we want rows related to **all values** of the second relation.
- Commonly used in “find all” type queries (e.g., find students who took all courses).

Union –

```
mysql> SELECT Email FROM guests
    -> UNION
    -> SELECT Email FROM staff;
+--------------------------+
| Email                    |
+--------------------------+
| test@example.com         |
| amit.verma@hotelmg.com   |
| priya.sharma@hotelmg.com |
| rajesh.singh@hotelmg.com |
| sneha.kapoor@hotelmg.com |
+--------------------------+
5 rows in set (0.01 sec)
```

Intersection –

```
mysql> SELECT DISTINCT g.GuestID, g.GuestName
    -> FROM guests g
    -> INNER JOIN bookings b ON g.GuestID = b.guest_id;
+---------+-----------------+
| GuestID | GuestName       |
+---------+-----------------+
|       1 | Test Guest      |
|       2 | Division Tester |
|       4 | Division Tester |
|       5 | Division Tester |
+---------+-----------------+
4 rows in set (0.00 sec)
```

Difference –

```
mysql> SELECT g.GuestID, g.GuestName
    -> FROM guests g
    -> WHERE g.GuestID NOT IN (
    ->     SELECT b.guest_id
    ->     FROM bookings b
    -> );
+---------+-----------------+
| GuestID | GuestName       |
+---------+-----------------+
|       3 | Division Tester |
|       6 | Division Tester |
|       7 | Division Tester |
+---------+-----------------+
3 rows in set (0.00 sec)
```

Division –

```
mysql> SELECT g.GuestID, g.GuestName
    -> FROM guests g
    -> WHERE NOT EXISTS (
    ->     SELECT r.room_id
    ->     FROM rooms r
    ->     WHERE NOT EXISTS (
    ->         SELECT 1
    ->         FROM bookings b
    ->         WHERE b.guest_id = g.GuestID
    ->           AND b.room_id = r.room_id
    ->     )
    -> );
+---------+-----------------+
| GuestID | GuestName       |
+---------+-----------------+
|       8 | Division Tester |
+---------+-----------------+
1 row in set (0.00 sec)
```

(Union all):-

```
ysql> SELECT GuestID AS id, GuestName AS name
    -> FROM guests
    -> UNION ALL
    -> SELECT guest_id AS id, status AS name
    -> FROM bookings;
----+-----------------+
 id | name            |
----+-----------------+
  1 | Test Guest      |
  2 | Division Tester |
  3 | Division Tester |
  4 | Division Tester |
  5 | Division Tester |
  6 | Division Tester |
  7 | Division Tester |
  8 | Division Tester |
  9 | Division Tester |
  1 | Reserved        |
  2 | CheckedIn       |
  1 | Reserved        |
  1 | Reserved        |
  2 | CheckedIn       |
  1 | Reserved        |
  2 | CheckedIn       |
  1 | Reserved        |
  2 | CheckedIn       |
  2 | Reserved        |
  4 | Reserved        |
  5 | Reserved        |
  5 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
  8 | Reserved        |
----+-----------------+
2 rows in set (0.00 sec)
```

Experiment :- 07

Object:- PL/SQL Programming I :Programs using named and unnamed blocks, Programs using Cursors, Cursor loops and records.

- In PL/SQL, programs are written inside blocks that can be named (stored procedures/functions) or unnamed (anonymous).

- Cursors allow row-by-row processing of query results using DECLARE, OPEN, FETCH, CLOSE operations.

- PL/SQL also supports parameterized cursors, nested cursors, and using cursors inside stored procedures for complex data handling.

PL/SQL Concept Used:-

Concept	Description
Anonymous Block	A temporary PL/SQL block executed once, not stored in DB.
named block	Stored permanently (Procedures/Functions). Can be reused.
cursor	Pointer to result set of a SQL query. Used for row-by-row processing.
cursor operation	DECLARE → OPEN → FETCH → CLOSE (manual control).
cursor for loop	Automatically opens, fetches, and closes cursor.
cursor records	Store full row values using %ROWTYPE.
parametrized cursor	Cursor that accepts parameters (dynamic queries).
nested cursor	Cursor inside another cursor. Used for parent-child relationships.
cursor inside procedure	Used to display or process multiple rows in stored procedures.

SQL Commands and Queries Using hotelmg database:-

Anonymous Block (Unnamed Block):-

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE CountGuests()
    -> BEGIN
    ->     DECLARE v_guest_count INT;
    ->     SELECT COUNT(*) INTO v_guest_count FROM guests;
    ->     SELECT CONCAT('Total Guests: ', v_guest_count) AS Res
ult;
    -> END$$
ERROR 1304 (42000): PROCEDURE CountGuests already exists
mysql>
mysql> DELIMITER ;
mysql> CALL CountGuests();
+-----------------+
| Result          |
+-----------------+
| Total Guests: 9 |
+-----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.01 sec)
```

Simple Cursor :-

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE ShowGuests()
    -> BEGIN
    ->    DECLARE done INT DEFAULT 0;
    ->    DECLARE v_guest_id INT;
    ->    DECLARE v_guest_name VARCHAR(100);
    ->
    ->    DECLARE guest_cursor CURSOR FOR
    ->       SELECT GuestID, GuestName FROM guests;
    ->
    ->    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->    OPEN guest_cursor;
    ->    read_loop: LOOP
    ->       FETCH guest_cursor INTO v_guest_id, v_guest_name;
    ->       IF done THEN
    ->          LEAVE read_loop;
    ->       END IF;
    ->       -- Instead of DBMS_OUTPUT, use SELECT
    ->       SELECT CONCAT('Guest ID: ', v_guest_id, ', Name: ', v_guest_name) AS Output;
    ->    END LOOP;
    ->    CLOSE guest_cursor;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql>
mysql> CALL ShowGuests();
+------------------------------+
| Output                       |
+------------------------------+
| Guest ID: 1, Name: Test Guest |
+------------------------------+
1 row in set (0.00 sec)
```

```
+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 2, Name: Division Tester |
+--------------------------------------+
1 row in set (0.02 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 3, Name: Division Tester |
+--------------------------------------+
1 row in set (0.05 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 4, Name: Division Tester |
+--------------------------------------+
1 row in set (0.06 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 5, Name: Division Tester |
+--------------------------------------+
1 row in set (0.08 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 6, Name: Division Tester |
+--------------------------------------+
1 row in set (0.10 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 7, Name: Division Tester |
+--------------------------------------+
1 row in set (0.12 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 8, Name: Division Tester |
+--------------------------------------+
1 row in set (0.14 sec)

+--------------------------------------+
| Output                               |
+--------------------------------------+
| Guest ID: 9, Name: Division Tester |
+--------------------------------------+
1 row in set (0.15 sec)

Query OK, 0 rows affected, 1 warning (0.18 sec)
```

3.Calling Procedure:-

```
mysql> CALL CountGuests();
+------------------+
| Result           |
+------------------+
| Total Guests: 9 |
+------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.02 sec)
```

4. Stored procedure

```
mysql> DELIMITER ;
mysql> CALL CountGuests();
+-----------------+
| Result          |
+-----------------+
| Total Guests: 9 |
+-----------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.02 sec)

mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE GetBookingsByGuest(IN g_id INT)
    -> BEGIN
    ->     SELECT booking_id, room_id, check_in, check_out, status
    ->     FROM bookings
    ->     WHERE guest_id = g_id;
    -> END$$
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL GetBookingsByGuest(8);
+------------+---------+------------+------------+----------+
| booking_id | room_id | check_in   | check_out  | status   |
+------------+---------+------------+------------+----------+
|         37 |       1 | 2025-12-15 | 2025-12-16 | Reserved |
|         39 |       1 | 2025-12-15 | 2025-12-16 | Reserved |
|         40 |       2 | 2025-12-17 | 2025-12-18 | Reserved |
|         41 |       3 | 2025-12-19 | 2025-12-20 | Reserved |
|         42 |       4 | 2025-12-21 | 2025-12-22 | Reserved |
|         44 |       1 | 2025-12-15 | 2025-12-16 | Reserved |
|         45 |       2 | 2025-12-17 | 2025-12-18 | Reserved |
|         46 |       3 | 2025-12-19 | 2025-12-20 | Reserved |
|         47 |       4 | 2025-12-21 | 2025-12-22 | Reserved |
|         48 |       5 | 2025-12-23 | 2025-12-24 | Reserved |
+------------+---------+------------+------------+----------+
10 rows in set (0.00 sec)
```

4. Parametrized Cursor:-

```
mysql>
mysql> CREATE PROCEDURE ShowBookingsByGuest(IN p_guest_id INT)
    -> BEGIN
    ->     DECLARE done INT DEFAULT 0;
    ->     DECLARE v_booking_id INT;
    ->     DECLARE v_room_id INT;
    ->     DECLARE v_check_in DATE;
    ->     DECLARE v_check_out DATE;
    ->
    ->     -- Define the cursor with a parameterized query
    ->     DECLARE booking_cursor CURSOR FOR
    ->         SELECT booking_id, room_id, check_in, check_out
    ->         FROM bookings
    ->         WHERE guest_id = p_guest_id
    ->         ORDER BY booking_id;
    ->
    ->     -- Handler to detect end of cursor
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN booking_cursor;
    ->
    ->     read_loop: LOOP
    ->         FETCH booking_cursor INTO v_booking_id, v_room_id, v_check_in, v_check_out;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         -- Use SELECT to show output (MySQL has no DBMS_OUTPUT)
    ->         SELECT
    ->             CONCAT('Booking ID: ', v_booking_id,
    ->                    ', Room ID: ', v_room_id,
    ->                    ', Check-in: ', v_check_in,
    ->                    ', Check-out: ', v_check_out) AS Output;
    ->     END LOOP;
    ->
    ->     CLOSE booking_cursor;
    -> END$$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
```

calling procedure

```
mysql> CALL ShowBookingsByGuest(8);
+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 37, Room ID: 1, Check-in: 2025-12-15, Check-out: 2025-12-16 |
+-----------------------------------------------------------------------+
1 row in set (0.00 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 39, Room ID: 1, Check-in: 2025-12-15, Check-out: 2025-12-16 |
+-----------------------------------------------------------------------+
1 row in set (0.04 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 40, Room ID: 2, Check-in: 2025-12-17, Check-out: 2025-12-18 |
+-----------------------------------------------------------------------+
1 row in set (0.07 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 41, Room ID: 3, Check-in: 2025-12-19, Check-out: 2025-12-20 |
+-----------------------------------------------------------------------+
1 row in set (0.10 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 42, Room ID: 4, Check-in: 2025-12-21, Check-out: 2025-12-22 |
+-----------------------------------------------------------------------+
1 row in set (0.12 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 44, Room ID: 1, Check-in: 2025-12-15, Check-out: 2025-12-16 |
+-----------------------------------------------------------------------+
1 row in set (0.15 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 45, Room ID: 2, Check-in: 2025-12-17, Check-out: 2025-12-18 |
+-----------------------------------------------------------------------+
1 row in set (0.18 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 46, Room ID: 3, Check-in: 2025-12-19, Check-out: 2025-12-20 |
+-----------------------------------------------------------------------+
1 row in set (0.21 sec)

+-----------------------------------------------------------------------+
| Output                                                                |
+-----------------------------------------------------------------------+
| Booking ID: 47, Room ID: 4, Check-in: 2025-12-21, Check-out: 2025-12-22 |
+-----------------------------------------------------------------------+
1 row in set (0.23 sec)
```

5.Nested Cursor:-

```
mysql>
mysql> CREATE PROCEDURE ShowGuestsAndBookings()
    -> BEGIN
    ->     DECLARE done_guest INT DEFAULT 0;
    ->     DECLARE v_guest_id INT;
    ->     DECLARE v_guest_name VARCHAR(100);
    ->
    ->     DECLARE guest_cursor CURSOR FOR
    ->         SELECT GuestID, GuestName FROM guests;
    ->
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done_guest = 1;
    ->
    ->     OPEN guest_cursor;
    ->     guest_loop: LOOP
    ->         FETCH guest_cursor INTO v_guest_id, v_guest_name;
    ->         IF done_guest THEN
    ->             LEAVE guest_loop;
    ->         END IF;
    ->
    ->         SELECT CONCAT('Guest: ', v_guest_name) AS Output;
    ->
    ->         -- Instead of a nested cursor, just run a SELECT for bookings
    ->         SELECT booking_id, room_id, check_in, check_out
    ->         FROM bookings
    ->         WHERE guest_id = v_guest_id;
    ->     END LOOP;
    ->     CLOSE guest_cursor;
    -> END$$
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL CountGuests();
+-----------------+
| Result          |
+-----------------+
| Total Guests: 9 |
+-----------------+
1 row in set (0.00 sec)
```

:

Experiment :- 08

Object:- PL/SQL Programming II : Creating stored procedures,functions,Trigger.

In PL/SQL, Stored Procedures, Functions, and Triggers are named program units stored permanently in the database.

They allow automation, code reuse, and custom logic inside the database engine.

- PL/SQL Program Unit:

Program Unit	Description
Procedure	Named PL/SQL block that performs an action but does not return a value.
Function	Similar to a procedure but returns a single value using RETURN.
Trigger	Automatically executes when a specific event (INSERT / UPDATE / DELETE) occurs on a table.

- SQL Commands and Queries Using Cab Service Database:-

Procedures:

creating procedure(SELECT) and calling the procedure:-

```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE procedure_name()
    -> BEGIN
    ->     -- SQL statements go here
    -> END //
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL ShowAllGuests();
+---------+-----------------+-----------+-------------------
| GuestID | GuestName       | ContactNo | Email
+---------+-----------------+-----------+-------------------
|       1 | Test Guest      | 9999999999 | test@example.com
|       2 | Division Tester | 8888888888 | division@test.com
|       3 | Division Tester | 8888888888 | division@test.com
|       4 | Division Tester | 8888888888 | division@test.com
|       5 | Division Tester | 8888888888 | division@test.com
|       6 | Division Tester | 8888888888 | division@test.com
|       7 | Division Tester | 8888888888 | division@test.com
|       8 | Division Tester | 9999999999 | division@test.com
|       9 | Division Tester | 9999999999 | division@test.com
+---------+-----------------+-----------+-------------------
9 rows in set (0.00 sec)

Query OK, 0 rows affected (0.25 sec)
```

creating procedure(insert) and calling it:-

```
mysql> DELIMITER //
mysql>
mysql> DROP PROCEDURE IF EXISTS InsertGuest;
    ->
    -> CREATE PROCEDURE InsertGuest(
    ->     IN p_GuestName VARCHAR(100),
    ->     IN p_ContactNo VARCHAR(15),
    ->     IN p_Email VARCHAR(100)
    -> )
    -> BEGIN
    ->     INSERT INTO Guests (GuestName, ContactNo, Email)
    ->     VALUES (p_GuestName, p_ContactNo, p_Email);
    -> END //
Query OK, 0 rows affected, 1 warning (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL InsertGuest('Rahul Sharma', '9876543210', 'rahul@example.com');
Query OK, 1 row affected (0.04 sec)

mysql> SELECT * FROM Guests;
+---------+----------------+------------+------------------+
| GuestID | GuestName      | ContactNo  | Email            |
+---------+----------------+------------+------------------+
|       1 | Test Guest     | 9999999999 | test@example.com |
|       2 | Division Tester | 8888888888 | division@test.com |
|       3 | Division Tester | 8888888888 | division@test.com |
|       4 | Division Tester | 8888888888 | division@test.com |
|       5 | Division Tester | 8888888888 | division@test.com |
|       6 | Division Tester | 8888888888 | division@test.com |
|       7 | Division Tester | 8888888888 | division@test.com |
|       8 | Division Tester | 9999999999 | division@test.com |
|       9 | Division Tester | 9999999999 | division@test.com |
|      10 | Rahul Sharma    | 9876543210 | rahul@example.com |
+---------+----------------+------------+------------------+
```

Creating procedure (Select + Insert)and calling procedure:-

```
mysql> DELIMITER //
mysql>
mysql> DROP PROCEDURE IF EXISTS InsertAndShowGuests;
    ->
    -> CREATE PROCEDURE InsertAndShowGuests(
    ->     IN p_GuestName VARCHAR(100),
    ->     IN p_ContactNo VARCHAR(15),
    ->     IN p_Email VARCHAR(100)
    -> )
    -> BEGIN
    ->     -- Insert new guest
    ->     INSERT INTO Guests (GuestName, ContactNo, Email)
    ->     VALUES (p_GuestName, p_ContactNo, p_Email);
    ->
    ->     -- Show all guests after insertion
    ->     SELECT GuestID, GuestName, ContactNo, Email
    ->     FROM Guests;
    -> END //
Query OK, 0 rows affected, 1 warning (0.01 sec)

Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL InsertAndShowGuests('Amit Verma', '9123456780', 'amit@example.co
+---------+-----------------+------------+-------------------+
| GuestID | GuestName       | ContactNo  | Email             |
+---------+-----------------+------------+-------------------+
|       1 | Test Guest      | 9999999999 | test@example.com  |
|       2 | Division Tester | 8888888888 | division@test.com |
|       3 | Division Tester | 8888888888 | division@test.com |
|       4 | Division Tester | 8888888888 | division@test.com |
|       5 | Division Tester | 8888888888 | division@test.com |
|       6 | Division Tester | 8888888888 | division@test.com |
|       7 | Division Tester | 8888888888 | division@test.com |
|       8 | Division Tester | 9999999999 | division@test.com |
|       9 | Division Tester | 9999999999 | division@test.com |
|      10 | Rahul Sharma    | 9876543210 | rahul@example.com |
|      11 | Amit Verma      | 9123456780 | amit@example.com  |
+---------+-----------------+------------+-------------------+
11 rows in set (0.04 sec)

Query OK, 0 rows affected (0.27 sec)
```

2. Triggers:-

Create table for logging:-

```
mysql> CREATE TABLE GuestLog (
    ->     LogID INT AUTO_INCREMENT PRIMARY KEY,
    ->     GuestID INT,
    ->     ActionType VARCHAR(20),
    ->     ActionTime TIMESTAMP DEFAULT CURRENT_TIMESTAM
    ->     OldGuestName VARCHAR(100),
    ->     OldContactNo VARCHAR(15),
    ->     OldEmail VARCHAR(100),
    ->     NewGuestName VARCHAR(100),
    ->     NewContactNo VARCHAR(15),
    ->     NewEmail VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.01 sec)
```

Create Trigger:-

```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER after_guest_insert
    -> AFTER INSERT ON Guests
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO GuestLog (GuestID, ActionType, NewGuestName, NewContactNo, NewE
    ->     VALUES (NEW.GuestID, 'INSERT', NEW.GuestName, NEW.ContactNo, NEW.Email);
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

Insert values to test trigger:-

```
mysql> INSERT INTO Guests (GuestName, ContactNo, Email)
    -> VALUES ('Ravi Sharma', '9876543210', 'ravi@example.com
Query OK, 1 row affected (0.01 sec)
```

Trigger Output:-

```
mysql> SELECT * FROM GuestLog;
+-------+---------+------------+---------------------+--------------+-------------+----------+--------------+--------------+-----------
| LogID | GuestID | ActionType | ActionTime          | OldGuestName | OldContactNo | OldEmail | NewGuestName | NewContactNo | NewEmail
+-------+---------+------------+---------------------+--------------+-------------+----------+--------------+--------------+-----------
|     1 |      12 | INSERT     | 2025-12-13 17:39:36 | NULL         | NULL        | NULL     | Ravi Sharma  | 9876543210   | ravi@examp
+-------+---------+------------+---------------------+--------------+-------------+----------+--------------+--------------+-----------
1 row in set (0.00 sec)
```

Creating Trigger (Before Update):-

```
mysql>
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER before_guest_update
    -> BEFORE UPDATE ON Guests
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO GuestLog (GuestID, ActionType,
    ->                           OldGuestName, OldContactNo, OldEmail)
    ->     VALUES (OLD.GuestID, 'BEFORE UPDATE',
    ->             OLD.GuestName, OLD.ContactNo, OLD.Email);
    -> END //
Query OK, 0 rows affected (0.02 sec)
```

Update Trigger:-

```
mysql>
mysql> DELIMITER ;
mysql> INSERT INTO Guests (GuestName, ContactNo, Email)
    -> VALUES ('Amit Verma', '9123456780', 'amit@example.com');
Query OK, 1 row affected (0.02 sec)

mysql> UPDATE Guests
    -> SET ContactNo = '9999999999', Email = 'amit_new@example.com'
    -> WHERE GuestName = 'Amit Verma';
Query OK, 2 rows affected (0.02 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

Trigger Output:-

```
mysql> SELECT * FROM GuestLog;
+-------+---------+---------------+---------------------+--------------+---------------+---------------+--------------+---------------+---------------+
| LogID | GuestID | ActionType    | ActionTime          | OldGuestName | OldContactNo  | OldEmail      | NewGuestName | NewContactNo  | NewEmail      |
+-------+---------+---------------+---------------------+--------------+---------------+---------------+--------------+---------------+---------------+
|     1 |      12 | INSERT        | 2025-12-13 17:39:36 | NULL         | NULL          | NULL          | Ravi Sharma  | 9876543210    | ravi@example. |
|     2 |      13 | INSERT        | 2025-12-13 17:45:07 | NULL         | NULL          | NULL          | Amit Verma   | 9123456780    | amit@example. |
|     3 |      11 | BEFORE UPDATE | 2025-12-13 17:45:18 | Amit Verma   | 9123456780    | amit@example.com | NULL      | NULL          | NULL          |
|     4 |      11 | UPDATE        | 2025-12-13 17:45:18 | Amit Verma   | 9123456780    | amit@example.com | Amit Verma | 9999999999   | amit_new@exam |
|     5 |      13 | BEFORE UPDATE | 2025-12-13 17:45:18 | Amit Verma   | 9123456780    | amit@example.com | NULL      | NULL          | NULL          |
|     6 |      13 | UPDATE        | 2025-12-13 17:45:18 | Amit Verma   | 9123456780    | amit@example.com | Amit Verma | 9999999999   | amit_new@exam |
+-------+---------+---------------+---------------------+--------------+---------------+---------------+--------------+---------------+---------------+
6 rows in set (0.00 sec)
```

3.Function:-

Count bookings:-

```
mysql> CREATE FUNCTION CountBookings(p_guest_id INT)
    -> RETURNS INT
    -> DETERMINISTIC
    -> BEGIN
    ->     DECLARE booking_count INT;
    ->
    ->     SELECT COUNT(*) INTO booking_count
    ->     FROM Bookings
    ->     WHERE guest_id = p_guest_id;
    ->
    ->     RETURN booking_count;
    -> END //
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT CountBookings(101) AS TotalBookings;
+---------------+
| TotalBookings |
+---------------+
|             3 |
+---------------+
1 row in set (0.00 sec)
```

Total amount paid by a guest:-

```
mysql> CREATE FUNCTION TotalAmountPaid(p_guest_id INT)
    -> RETURNS DECIMAL(10,2)
    -> DETERMINISTIC
    -> BEGIN
    ->     DECLARE total_paid DECIMAL(10,2);
    ->
    ->     SELECT IFNULL(SUM(total_amount),0) INTO total
    ->     FROM Bookings
    ->     WHERE guest_id = p_guest_id;
    ->
    ->     RETURN total_paid;
    -> END //
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT TotalAmountPaid(101) AS AmountPaid;
+------------+
| AmountPaid |
+------------+
|   16500.00 |
+------------+
1 row in set (0.00 sec)
```

Most recent booking date-

```
ysql>
ysql> CREATE FUNCTION MostRecentBookingDate(p_guest_id
    -> RETURNS DATE
    -> DETERMINISTIC
    -> BEGIN
    ->     DECLARE recent_date DATE;
    ->
    ->     SELECT MAX(check_in) INTO recent_date
    ->     FROM Bookings
    ->     WHERE guest_id = p_guest_id;
    ->
    ->     RETURN recent_date;
    -> END //
uery OK, 0 rows affected (0.00 sec)

ysql>
ysql> DELIMITER ;
ysql> SELECT MostRecentBookingDate(101) AS LatestBookir
+---------------+
| LatestBooking |
+---------------+
| 2025-12-25    |
+---------------+
 row in set (0.00 sec)
```


:

Experiment:-09

Object:- DBMS Project Description – Hotel management System

The primary object of the Hotel Management System Database Project is to design and implement a structured database that can efficiently manage all core operations of a hotel. The system aims to provide a centralized platform for storing, retrieving, and processing hotel data with accuracy, consistency, and security.

Key Features of the Cab Service Database :-

- **Guest Information Management**
  Maintain complete records of guests including personal details, contact information, and booking history to ensure personalized service and efficient customer handling. **Room Allocation and Tracking**
  Manage room details such as room number, type, floor, and status (Available, Occupied, Reserved, Maintenance). Ensure real-time tracking of room availability to avoid conflicts and maximize occupancy.

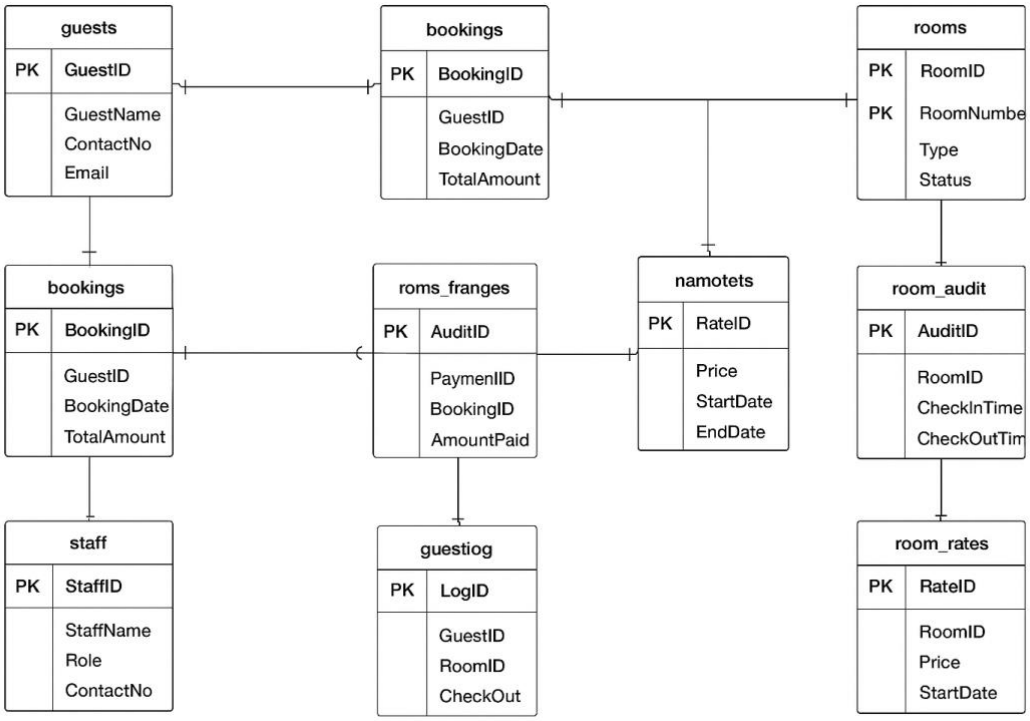
- **Booking & Reservation Handling**
  Record and manage guest bookings with check-in and check-out dates, number of adults/children, and booking status. Support multiple bookings per guest and handle cancellations or modifications seamlessly.

- **Payment & Billing System**
  Track payments linked to bookings, calculate total amount paid by each guest, and generate invoices. Support multiple payment modes and ensure financial accuracy.

- **Service Management**
  Integrate additional services (food, laundry, cab, spa, etc.) into the database. Link these services to guest accounts for consolidated billing and better service tracking.
- **Audit & Reporting**
  Maintain logs of all operations (bookings, payments, cancellations) for accountability. Generate reports such as occupancy rates, revenue summaries, and guest activity to support managerial decision-making.

- **Functions & Procedures for Efficiency**
  Implement SQL functions like:
- `CountBookings(GuestID)` → Total bookings of a guest.
- `TotalAmountPaid(GuestID)` → Total amount spent by a guest.
- `MostRecentBookingDate(GuestID)` → Latest booking date of a guest.
  These functions simplify queries and enhance usability.

ER-diagram of hotel management system



Experiment :- 10

Object:- RGPV Queries

1.Delete Duplicate Row from a table:-

```
mysql> DELETE g1 FROM Guests g1
    -> JOIN Guests g2
    -> ON g1.GuestName = g2.GuestName
    ->    AND g1.ContactNo = g2.ContactNo
    ->    AND g1.Email = g2.Email
    -> WHERE g1.GuestID > g2.GuestID
    -> AND g1.GuestID NOT IN (SELECT GuestID FROM Bookings);
Query OK, 0 rows affected (0.00 sec)
```

2.Display Alternate Row from a table:-

```
mysql> SELECT GuestID, GuestName, ContactNo, Email
    -> FROM Guests
    -> WHERE ( @row_num := @row_num + 1 ) % 2 = 1;
+---------+----------------+-----------+----------------------+
| GuestID | GuestName      | ContactNo | Email                |
+---------+----------------+-----------+----------------------+
|       1 | Test Guest     | 9999999999 | test@example.com     |
|       8 | Division Tester | 9999999999 | division@test.com    |
|      11 | Amit Verma     | 9999999999 | amit_new@example.com |
+---------+----------------+-----------+----------------------+
3 rows in set (0.00 sec)
```

3.Delete Alternate Row from a table:

```
mysql> DELETE FROM Guests
    -> WHERE GuestID % 2 = 0;
Query OK, 4 rows affected (0.01 sec)
```

4.Update multiple rows using single UPDATE statement:-

```
mysql> UPDATE Guests
    -> SET ContactNo = CASE GuestID
    ->     WHEN 1 THEN '9876543210'
    ->     WHEN 2 THEN '9123456789'
    ->     WHEN 3 THEN '9998887777'
    -> END,
    -> Email = CASE GuestID
    ->     WHEN 1 THEN 'amit_new@gmail.com'
    ->     WHEN 2 THEN 'priya_new@gmail.com'
    ->     WHEN 3 THEN 'rahul_new@gmail.com'
    -> END
    -> WHERE GuestID IN (1,2,3);
Query OK, 1 row affected (0.00 sec)
```

5. Find 3rd highest & 3rd lowest fare:-

3rd HIGHEST fare:-

```
mysql> -- 3rd Highest Fare
mysql> SELECT DISTINCT fare
    -> FROM Rooms
    -> ORDER BY fare DESC
    -> LIMIT 1 OFFSET 2;
+---------+
| fare    |
+---------+
| 3500.00 |
+---------+
1 row in set (0.00 sec)
```

3rd LOWEST fare:-

```
mysql> -- 3rd Lowest Fare
mysql> SELECT DISTINCT fare
    -> FROM Rooms
    -> ORDER BY fare ASC
    -> LIMIT 1 OFFSET 2;
+---------+
| fare    |
+---------+
| 5000.00 |
+---------+
1 row in set (0.00 sec)
```

6.Display 3rd, 4th and 9th rows:-

```
mysql> SELECT * FROM Guests LIMIT 1 OFFSET 2;
+---------+------------+-----------+-----------------+
| GuestID | GuestName  | ContactNo | Email           |
+---------+------------+-----------+-----------------+
|     102 | Rahul Mehta | 9998887777 | rahul@gmail.com |
+---------+------------+-----------+-----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM Guests LIMIT 1 OFFSET 3;
+---------+------------+-----------+-----------------+
| GuestID | GuestName  | ContactNo | Email           |
+---------+------------+-----------+-----------------+
|     103 | Sneha Gupta | 8887776666 | sneha@gmail.com |
+---------+------------+-----------+-----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM Guests LIMIT 1 OFFSET 8;
+---------+-------------+-----------+-----------------+
| GuestID | GuestName   | ContactNo | Email           |
+---------+-------------+-----------+-----------------+
|     108 | Rohit Sharma | 3332221111 | rohit@gmail.com |
+---------+-------------+-----------+-----------------+
1 row in set (0.00 sec)
```

7. Display driver names starting with D, S, R or P:-

```
mysql> SELECT GuestName
    -> FROM Guests
    -> WHERE GuestName LIKE 'D%'
    ->     OR GuestName LIKE 'S%'
    ->     OR GuestName LIKE 'R%'
    ->     OR GuestName LIKE 'P%';
+---------------+
| GuestName     |
+---------------+
| Rahul Mehta   |
| Sneha Gupta   |
| Rohit Sharma  |
+---------------+
3 rows in set (0.00 sec)
```

8.Show bookings done in current month:-

```
mysql> SELECT *
    -> FROM Bookings
    -> WHERE MONTH(check_in) = MONTH(CURDATE())
    ->   AND YEAR(check_in) = YEAR(CURDATE())
    ->   AND DAY(check_in) BETWEEN 1 AND 15;
+------------+----------+---------+------------+------------+--------+----------+-----------+---------------------+--------------+---------+
| booking_id | guest_id | room_id | check_in   | check_out  | adults | children | status    | created_at          | total_amount | GuestID |
+------------+----------+---------+------------+------------+--------+----------+-----------+---------------------+--------------+---------+
|         14 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved  | 2025-12-11 23:07:30 |      4500.00 |    NULL |
|         15 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:07:30 |      7200.00 |    NULL |
|         17 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved  | 2025-12-11 23:09:50 |      3500.00 |    NULL |
|         18 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved  | 2025-12-11 23:11:01 |      9800.00 |    NULL |
|         19 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:11:01 |      2500.00 |    NULL |
|         21 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved  | 2025-12-11 23:14:05 |         NULL |    NULL |
|         22 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:14:05 |         NULL |    NULL |
|         24 |        1 |       1 | 2025-12-12 | 2025-12-15 |      2 |        1 | Reserved  | 2025-12-11 23:14:48 |         NULL |    NULL |
|         25 |        2 |       2 | 2025-12-13 | 2025-12-16 |      1 |        0 | CheckedIn | 2025-12-11 23:14:48 |         NULL |    NULL |
|         26 |        2 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:44:05 |      2000.00 |    NULL |
|         32 |        4 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:46:01 |      2000.00 |    NULL |
|         34 |        5 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:47:18 |      2000.00 |    NULL |
|         36 |        5 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:49:27 |      2000.00 |    NULL |
|         37 |        8 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:55:58 |      2000.00 |    NULL |
|         39 |        8 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:56:51 |      2000.00 |    NULL |
|         44 |        8 |       1 | 2025-12-15 | 2025-12-16 |      1 |        0 | Reserved  | 2025-12-13 14:59:09 |      2000.00 |    NULL |
|         55 |      101 |       1 | 2025-12-14 | 2025-12-16 |      2 |        0 | Reserved  | 2025-12-13 17:55:53 |      5000.00 |    NULL |
+------------+----------+---------+------------+------------+--------+----------+-----------+---------------------+--------------+---------+
17 rows in set (0.00 sec)
```

9., ROLLBACK & SAVEPOINT:-

```
mysql>
mysql> INSERT INTO Guests (GuestName, ContactNo, Email)
    -> VALUES ('Rahul Mehta', '9998887777', 'rahul@gmail.com');
Query OK, 1 row affected (0.00 sec)

mysql> SAVEPOINT sp1;
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> INSERT INTO Guests (GuestName, ContactNo, Email)
    -> VALUES ('Sneha Gupta', '8887776666', 'sneha@gmail.com');
Query OK, 1 row affected (0.00 sec)

mysql> ROLLBACK TO sp1;
Query OK, 0 rows affected (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
```

10.Select, Insert, Update and Delete (MySQL Procedure):-

11.Delete a record and return 1 if successful else 0:-

Display name and booking date using cursor:-

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE DeleteGuest(
    ->     IN p_GuestID INT,
    ->     OUT result INT
    -> )
    -> BEGIN
    ->     DELETE FROM Guests WHERE GuestID = p_GuestID;
    ->
    ->     IF ROW_COUNT() > 0 THEN
    ->         SET result = 1;   -- success
    ->     ELSE
    ->         SET result = 0;   -- failure (no record found)
    ->     END IF;
    -> END$$
ERROR 1304 (42000): PROCEDURE DeleteGuest already exists
mysql>
mysql> DELIMITER ;
mysql> CALL DeleteGuest(102, @output);
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT @output;
+---------+
| @output |
+---------+
|       0 |
+---------+
1 row in set (0.00 sec)
```

Trigger to restrict insert/update/delete after 7 PM:-

```
mysql> DELIMITER $$
mysql>
mysql> CREATE TRIGGER trg_no_insert_after_7pm
    -> BEFORE INSERT ON bookings
    -> FOR EACH ROW
    -> BEGIN
    ->   IF CURTIME() > '19:00:00' THEN
    ->     SIGNAL SQLSTATE '45000'
    ->       SET MESSAGE_TEXT = 'Inserts are not allowed after 7
PM';
    ->   END IF;
    -> END$$
Query OK, 0 rows affected (0.06 sec)
```

Trigger to prevent duplicate phone numbers:-

```
mysql> CREATE TRIGGER prevent_duplicate_phone_insert
    -> BEFORE INSERT ON guests
    -> FOR EACH ROW
    -> BEGIN
    ->     IF EXISTS (
    ->         SELECT 1 FROM guests
    ->         WHERE ContactNo = NEW.ContactNo
    ->     ) THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Duplicate phone number not al
lowed';
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.06 sec)
```

Trigger to perform ON DELETE CASCADE manually:

```
mysql> DELIMITER $$
mysql>
mysql> CREATE TRIGGER trg_delete_guest
    -> BEFORE DELETE ON guests
    -> FOR EACH ROW
    -> BEGIN
    ->     DELETE FROM guests_details
    ->     WHERE GuestID = OLD.GuestID;
    -> END$$
Query OK, 0 rows affected (0.05 sec)
```

Trigger to restrict delete on Sunday:-

```
mysql> CREATE TRIGGER trg_delete_guest_combined
    -> BEFORE DELETE ON guests
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Restrict deletion on Sunday
    ->     IF DAYNAME(CURDATE()) = 'Sunday' THEN
    ->         SIGNAL SQLSTATE '45000'
    ->         SET MESSAGE_TEXT = 'Deletion is not allowed on Sunday';
    ->     END IF;
    ->
    ->     -- Cascade delete to guests_details
    ->     DELETE FROM guests_details
    ->     WHERE GuestID = OLD.GuestID;
    -> END$$
Query OK, 0 rows affected (0.05 sec)
```