TOURISM MANAGEMENT

Mini Project

21CSC205P -Database Management Systems

Presented By

RA2211003011034 – Diva Merja

RA2211003011046 – Sree Charanya

Guide

Dr.M.Kandan

Assistant Professor

Dept of CTech

Problem understanding, Identification of Entity and Relationships, Construction of DB using ER Model for the project

- **Problem understanding**

The problem statement for tourism management databases encompasses a myriad of challenges that demand strategic solutions for effective management and optimization of tourism operations. One of the primary challenges revolves around the fragmented nature of data sources within the tourism industry. Information pertinent to bookings, reservations, customer profiles, and inventory often exists in disparate systems, making it difficult to consolidate and utilize data efficiently. This fragmentation impedes seamless information flow, leading to inconsistencies and integrity issues within databases.

Customer Relationship Management (CRM) presents another critical challenge in tourism management databases. The tourism industry relies heavily on building and maintaining strong relationships with customers. This entails meticulous handling of sensitive customer data, tracking interactions across various touchpoints, and implementing targeted marketing strategies to enhance customer satisfaction and loyalty. However, ensuring data security, privacy, and compliance with regulations such as GDPR (General Data Protection Regulation) and CCPA (California Consumer Privacy Act) is paramount due to the sensitive nature of customer information.

Additionally, seamless integration with external systems is essential for tourism management databases to operate efficiently. This includes coordination with suppliers, such as hotels, airlines, and tour operators, to facilitate real-time bookings and updates. Integration with external booking platforms and distribution channels is also crucial for maximizing visibility and reaching a broader audience. Without seamless integration capabilities, tourism businesses may face difficulties in managing inventory, pricing, and availability across various channels.

Furthermore, robust reporting and analytics capabilities are indispensable for informed decision-making in the tourism sector. Access to timely and accurate data insights enables businesses to identify trends, monitor performance, and optimize strategies to stay competitive in a dynamic market environment. Analyzing customer behavior, booking patterns, and market trends empowers tourism businesses to make data-driven decisions that drive growth and profitability.

Problem understanding, Identification of Entity and Relationships, Construction of DB using ER Model for the project

Existing System

Existing solutions for tourism management databases often rely on Customer Relationship Management (CRM) and Enterprise Resource Planning (ERP) systems to streamline operations, manage reservations, and handle customer data. While these systems offer certain benefits, they also come with several disadvantages that can hinder their effectiveness in addressing the complex needs of the tourism industry.

One significant drawback of existing solutions is the high implementation costs associated with CRM and ERP systems. The initial investment required for software licenses, customization, and implementation services can be substantial, particularly for small and medium-sized tourism businesses with limited resources. Additionally, ongoing maintenance and support costs further contribute to the total cost of ownership, making these solutions financially prohibitive for some organizations.

Integration complexities present another challenge for existing tourism management databases. Integrating CRM and ERP systems with existing infrastructure, external booking platforms, and distribution channels can be a complex and time-consuming process. Compatibility issues, data migration challenges, and the need for custom development can prolong implementation timelines and increase project risks.

Problem understanding, Identification of Entity and Relationships, Construction of DB using ER Model for the project

Objective of the Project

The objective of this project is to develop an innovative and comprehensive tourism management database system that addresses the multifaceted challenges faced by the tourism industry. This system aims to streamline operations, enhance customer experiences, and optimize business processes through efficient handling of reservations, dynamic inventory management, and robust customer relationship management (CRM) capabilities.

The primary goal is to create a solution that overcomes the limitations of existing CRM and Enterprise Resource Planning (ERP) systems, offering a more cost-effective, flexible, and user-friendly alternative. By leveraging modern technologies such as cloud computing, artificial intelligence, and data analytics, the proposed system seeks to deliver superior performance and scalability while accommodating the evolving needs of tourism businesses.

Integration and Consolidation: Develop a unified platform that integrates fragmented data sources and consolidates information related to bookings, reservations, customer profiles, and inventory. This will facilitate seamless information flow, minimize data redundancy, and improve data integrity within the database.

Real-time Booking and Inventory Management: Implement functionalities for handling real-time bookings, coordinating with suppliers, and dynamically adjusting inventory based on demand fluctuations. This will help prevent overbooking situations, optimize resource utilization, and enhance operational efficiency.

Reporting and Analytics: Implement robust reporting and analytics capabilities to provide insights into customer behavior, booking patterns, and market trends. This will empower tourism businesses to make informed decisions, identify opportunities for growth, and stay ahead of competitors.

Overall, the project aims to deliver a cutting-edge tourism management database system that addresses the complex needs of the industry, fosters innovation, and drives sustainable growth in the tourism sector.

Identification of Entity and Relationships

Entity and Their AttributesLogin Entity

Contains login credentials for users.login_ID serves as the primary key.Includes Login_username, login_role_ID, and User_password attributes.Facilitates authentication and authorization processes.

User Entity:

It has UserId, Email, address, phone no, name. Name is composite attribute of firstname and lastname. Phone no is multi valued attribute. UserId is the Primary Key for User entity. It also has age attribute which is a derived attribute .

Hotel Entity:

Manages hotel booking information.Attributes include Htl_ID, Htl_rent, Htl_desc, Htl_name, and Htl_type.Provides comprehensive details about each hotel.Essential for tracking and managing hotel accommodations.

Booking Entity:

Handles booking information.Attributes include book_ID, book_title, book_desc, book_type, and book_date.Facilitates efficient tracking and management of bookings.Crucial for scheduling and organizing travel arrangements.

Travel Agent Entity:

Contains information about travel agents.TA_ID serves as the primary key.Includes TA_name and TA_desc attributes.Offers insights into services provided by travel agents.Integral for coordinating travel arrangements and services.

- **Identification of Entity and Relationships**

Relationships between Entities:

HAS:

Relationship between Login and User entities.

Explanation: Each login is associated with a user, indicating which user the login credentials belong to.

MANAGE:

Relationship between User and Booking entities.

Explanation: Each user manages one or more bookings, indicating the user's association with booking activities.

MANAGE:

Relationship between Booking and Hotel entities.

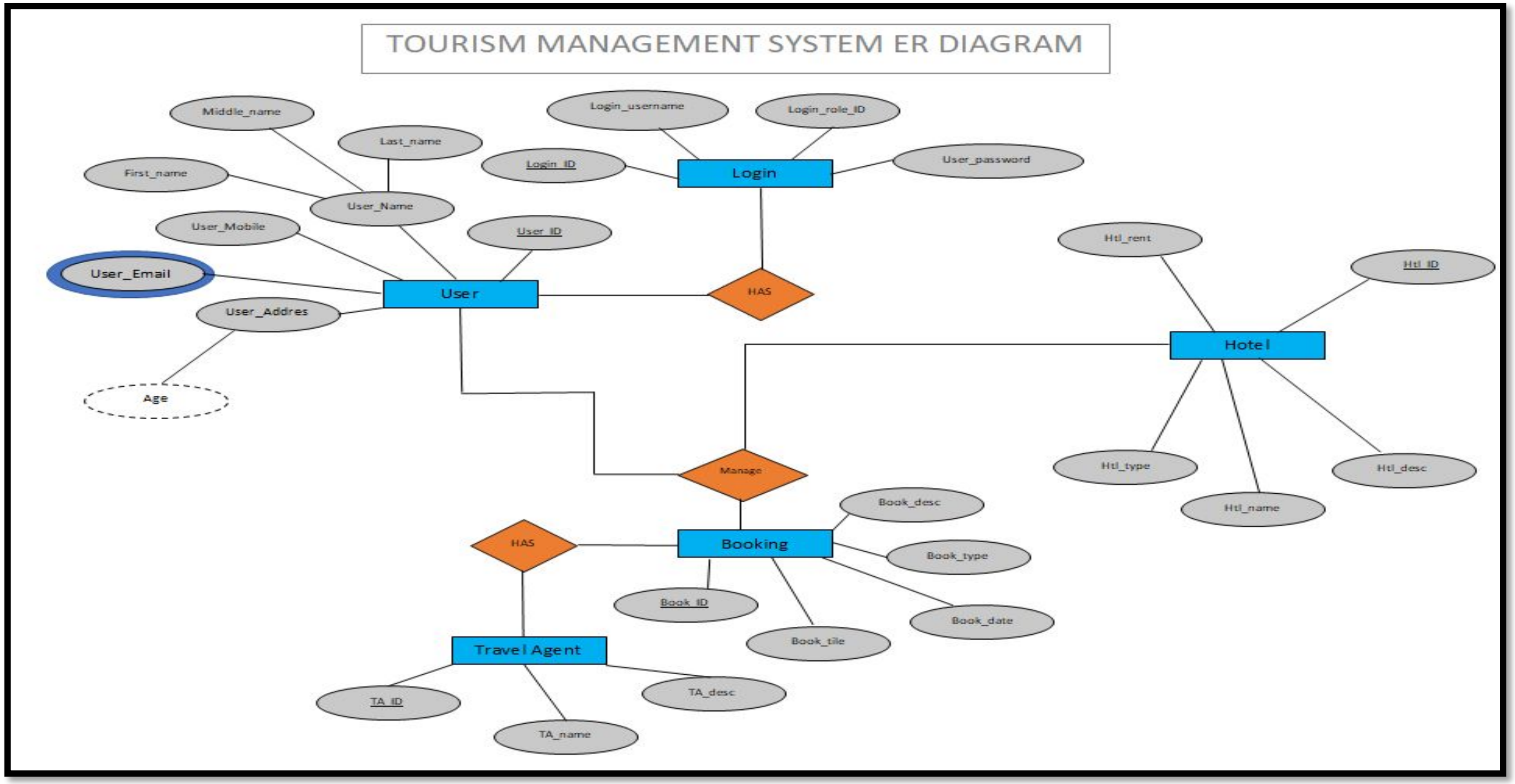
Explanation: Each booking is associated with a hotel, indicating the hotel where the booking is made.

HAS:

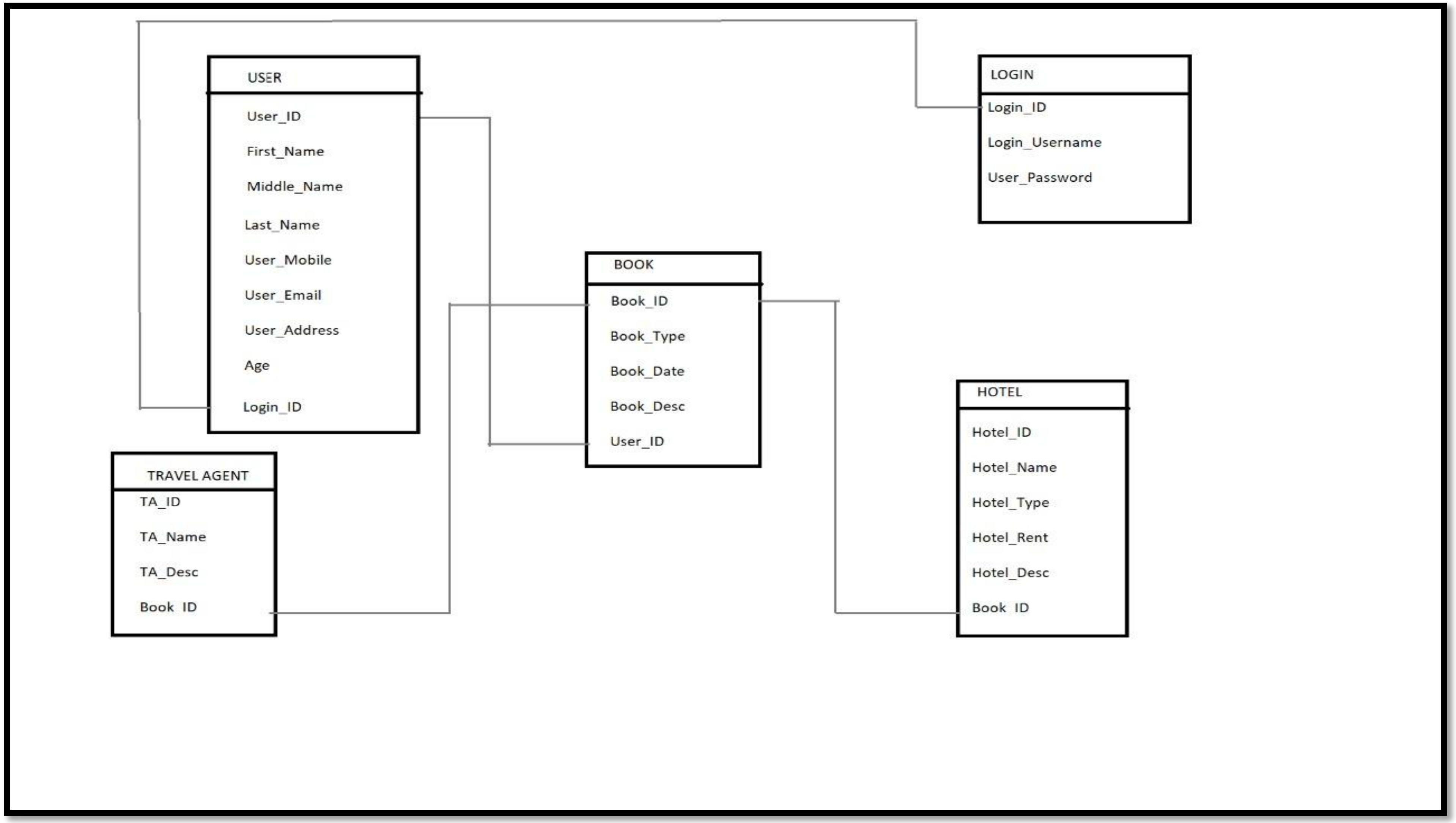
Relationship between Booking and TravelAgent entities.

Explanation: Each booking is associated with a travel agent, indicating the travel agent involved in arranging the booking.

Construction of DB using ER Model for the project



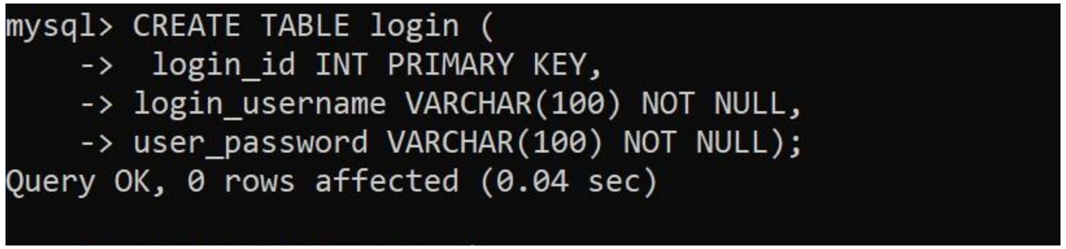
Design of Relational Schemas, Creation of Database Tables for the project



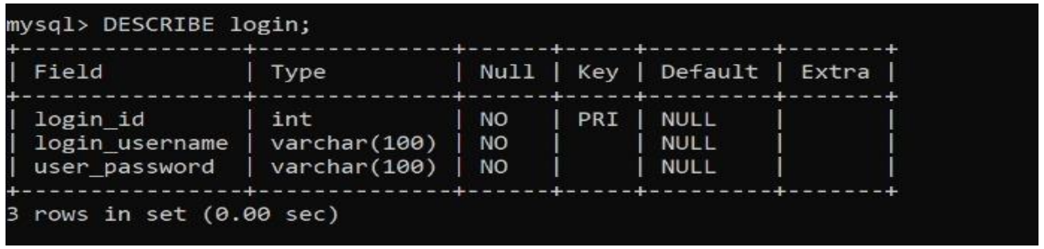
Design of Relational Schemas, Creation of Database Tables for the project (Cont..)

DDL Commands and Results

- **Creating the Login Table:**

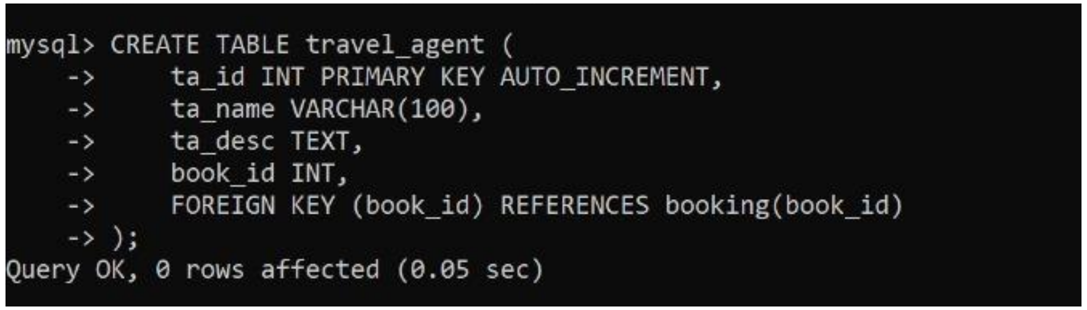


```
mysql> CREATE TABLE login (
    ->  login_id INT PRIMARY KEY,
    -> login_username VARCHAR(100) NOT NULL,
    -> user_password VARCHAR(100) NOT NULL);
Query OK, 0 rows affected (0.04 sec)
```

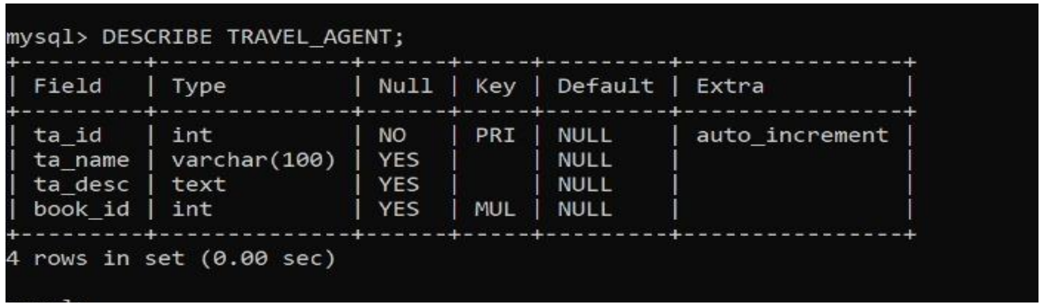


```
mysql> DESCRIBE login;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| login_id       | int          | NO   | PRI | NULL    |       |
| login_username | varchar(100) | NO   |     | NULL    |       |
| user_password  | varchar(100) | NO   |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

- **Creating the Login Table:**



```
mysql> CREATE TABLE travel_agent (
    ->     ta_id INT PRIMARY KEY AUTO_INCREMENT,
    ->     ta_name VARCHAR(100),
    ->     ta_desc TEXT,
    ->     book_id INT,
    ->     FOREIGN KEY (book_id) REFERENCES booking(book_id)
    -> );
Query OK, 0 rows affected (0.05 sec)
```

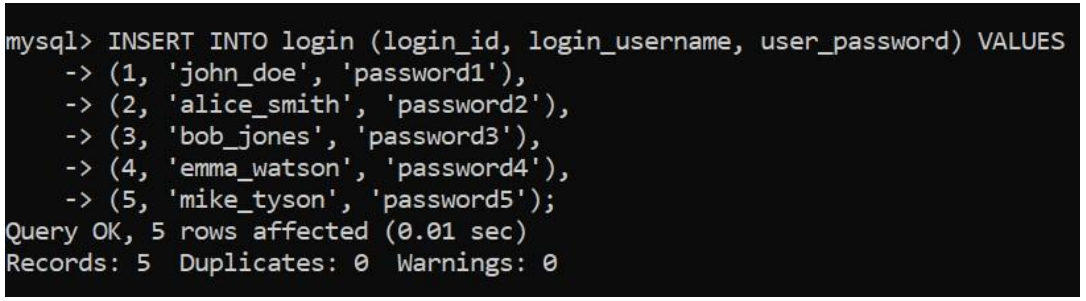


```
mysql> DESCRIBE TRAVEL_AGENT;
+---------+--------------+------+-----+---------+----------------+
| Field   | Type         | Null | Key | Default | Extra          |
+---------+--------------+------+-----+---------+----------------+
| ta_id   | int          | NO   | PRI | NULL    | auto_increment |
| ta_name | varchar(100) | YES  |     | NULL    |                |
| ta_desc | text         | YES  |     | NULL    |                |
| book_id | int          | YES  | MUL | NULL    |                |
+---------+--------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

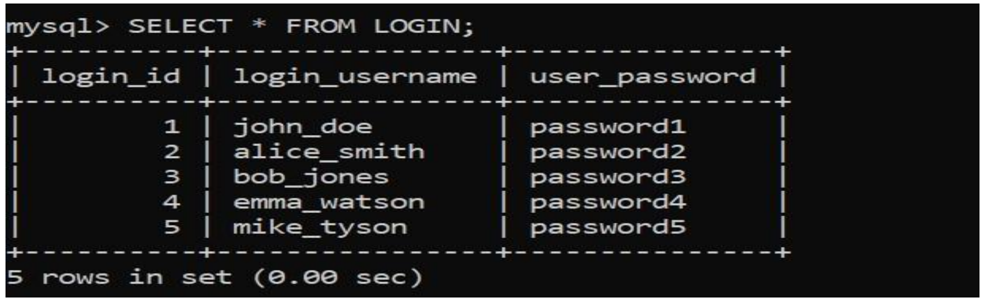
Design of Relational Schemas, Creation of Database Tables for the project (Cont..)

DML Commands (INSERT) and Results:

Insertion of data into login table:

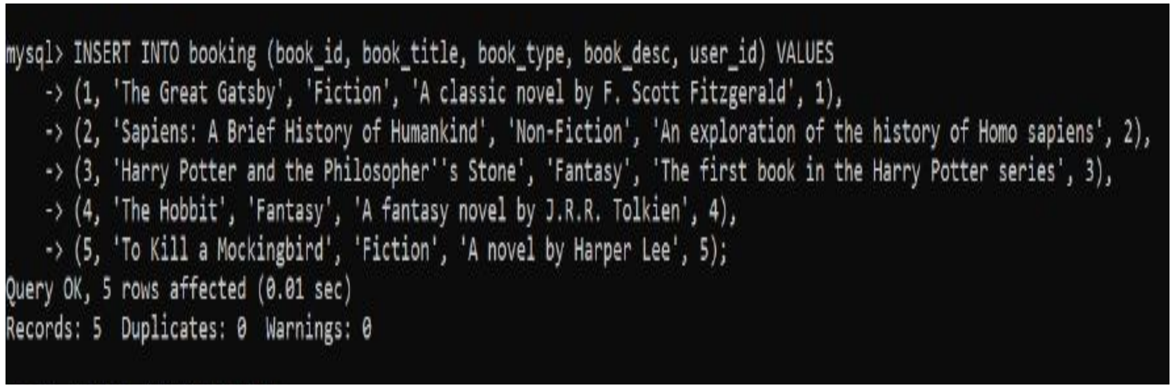


```
mysql> INSERT INTO login (login_id, login_username, user_password) VALUES
    -> (1, 'john_doe', 'password1'),
    -> (2, 'alice_smith', 'password2'),
    -> (3, 'bob_jones', 'password3'),
    -> (4, 'emma_watson', 'password4'),
    -> (5, 'mike_tyson', 'password5');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

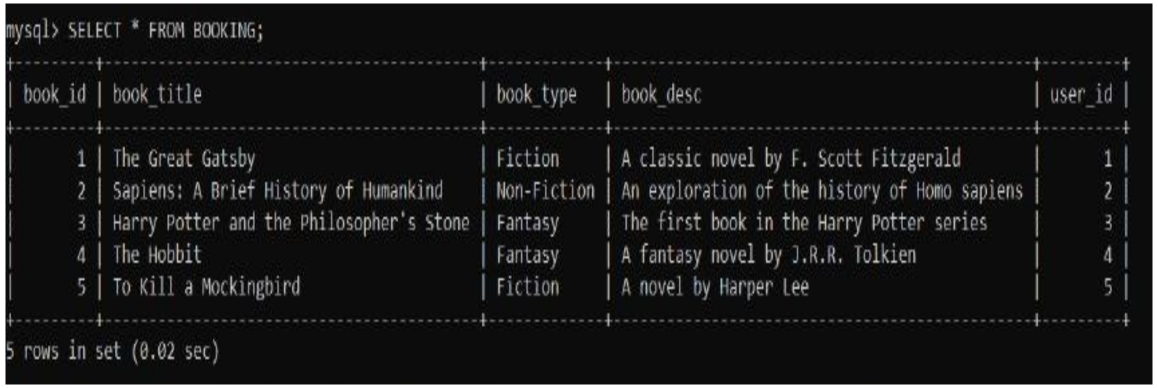


```
mysql> SELECT * FROM LOGIN;
+----------+----------------+---------------+
| login_id | login_username | user_password |
+----------+----------------+---------------+
|        1 | john_doe       | password1     |
|        2 | alice_smith    | password2     |
|        3 | bob_jones      | password3     |
|        4 | emma_watson    | password4     |
|        5 | mike_tyson     | password5     |
+----------+----------------+---------------+
5 rows in set (0.00 sec)
```

Insertion of data into booking table:



```
mysql> INSERT INTO booking (book_id, book_title, book_type, book_desc, user_id) VALUES
    -> (1, 'The Great Gatsby', 'Fiction', 'A classic novel by F. Scott Fitzgerald', 1),
    -> (2, 'Sapiens: A Brief History of Humankind', 'Non-Fiction', 'An exploration of the history of Homo sapiens', 2),
    -> (3, 'Harry Potter and the Philosopher''s Stone', 'Fantasy', 'The first book in the Harry Potter series', 3),
    -> (4, 'The Hobbit', 'Fantasy', 'A fantasy novel by J.R.R. Tolkien', 4),
    -> (5, 'To Kill a Mockingbird', 'Fiction', 'A novel by Harper Lee', 5);
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

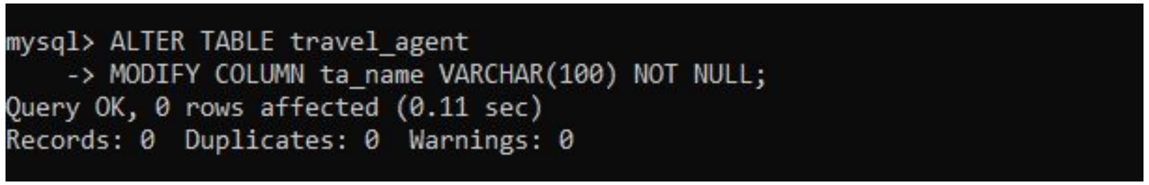


```
mysql> SELECT * FROM BOOKING;
+---------+------------------------------------------+-------------+-----------------------------------------------+---------+
| book_id | book_title                               | book_type   | book_desc                                     | user_id |
+---------+------------------------------------------+-------------+-----------------------------------------------+---------+
|       1 | The Great Gatsby                         | Fiction     | A classic novel by F. Scott Fitzgerald        |       1 |
|       2 | Sapiens: A Brief History of Humankind    | Non-Fiction | An exploration of the history of Homo sapiens |       2 |
|       3 | Harry Potter and the Philosopher's Stone | Fantasy     | The first book in the Harry Potter series     |       3 |
|       4 | The Hobbit                               | Fantasy     | A fantasy novel by J.R.R. Tolkien             |       4 |
|       5 | To Kill a Mockingbird                    | Fiction     | A novel by Harper Lee                         |       5 |
+---------+------------------------------------------+-------------+-----------------------------------------------+---------+
5 rows in set (0.02 sec)
```

Complex queries based on the concepts of constraints, sets, joins, views, Triggers and Cursors

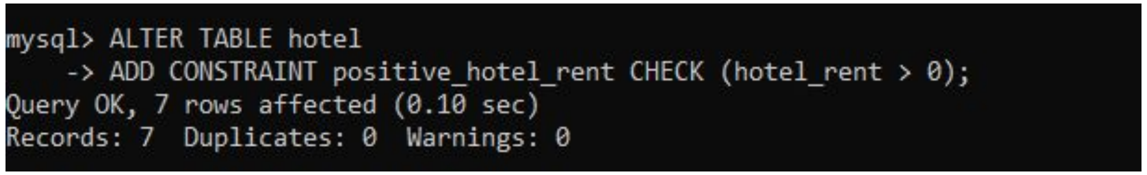
- Complex queries based on the concepts of constraints

- **Constraint to have ta_name for every tuple entry(not null).**



```
mysql> ALTER TABLE travel_agent
    -> MODIFY COLUMN ta_name VARCHAR(100) NOT NULL;
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

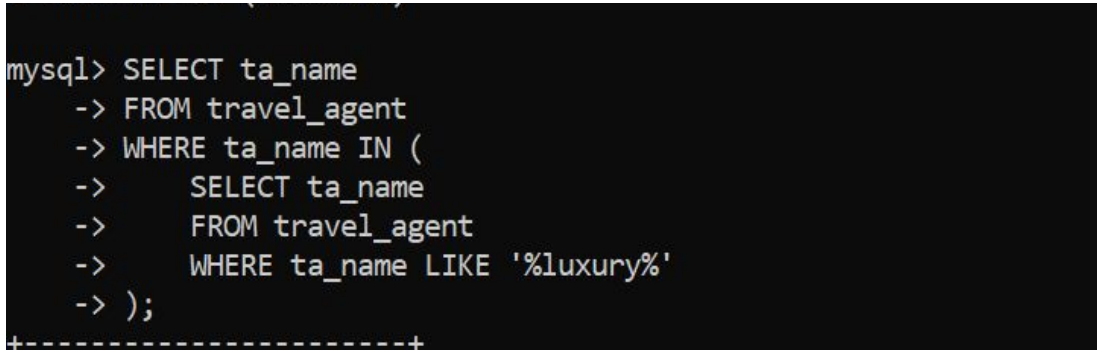
- **Constraint to have hotel rent higher than zero.**



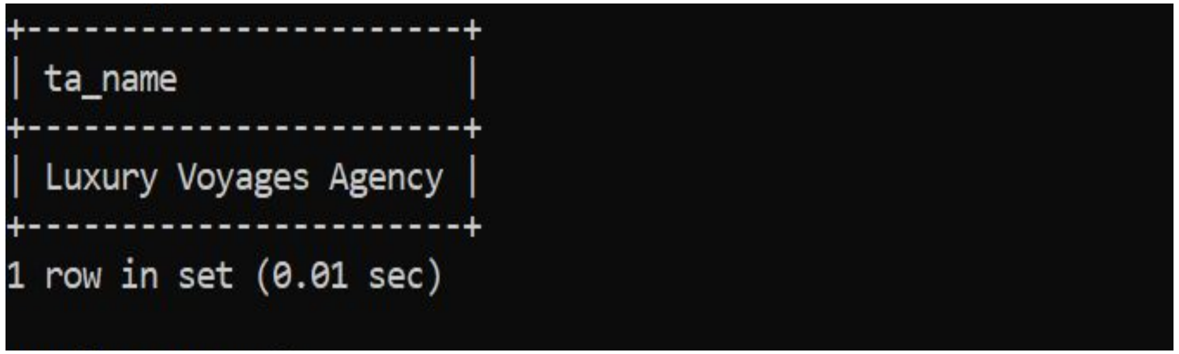
```
mysql> ALTER TABLE hotel
    -> ADD CONSTRAINT positive_hotel_rent CHECK (hotel_rent > 0);
Query OK, 7 rows affected (0.10 sec)
Records: 7  Duplicates: 0  Warnings: 0
```

Sets

- **Sub Query to get travel agent that is Luxury class.**

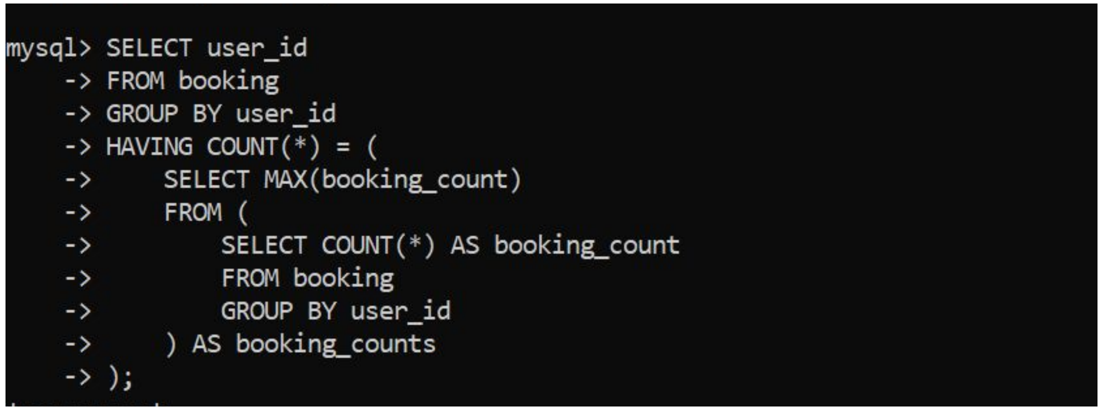


```
mysql> SELECT ta_name
    -> FROM travel_agent
    -> WHERE ta_name IN (
    ->     SELECT ta_name
    ->     FROM travel_agent
    ->     WHERE ta_name LIKE '%luxury%'
    -> );
```

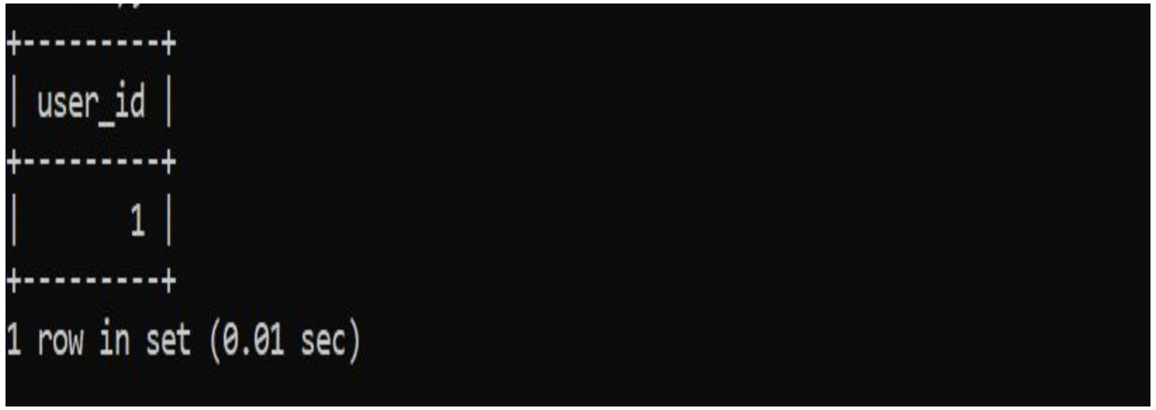


```
+-----------------------+
| ta_name               |
+-----------------------+
| Luxury Voyages Agency |
+-----------------------+
1 row in set (0.01 sec)
```

- **Sub Query to get user with highest number of booking.**



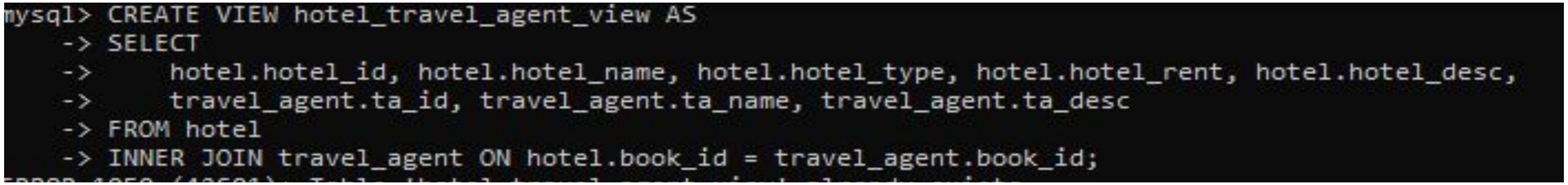
```
mysql> SELECT user_id
    -> FROM booking
    -> GROUP BY user_id
    -> HAVING COUNT(*) = (
    ->     SELECT MAX(booking_count)
    ->     FROM (
    ->         SELECT COUNT(*) AS booking_count
    ->         FROM booking
    ->         GROUP BY user_id
    ->     ) AS booking_counts
    -> );
```



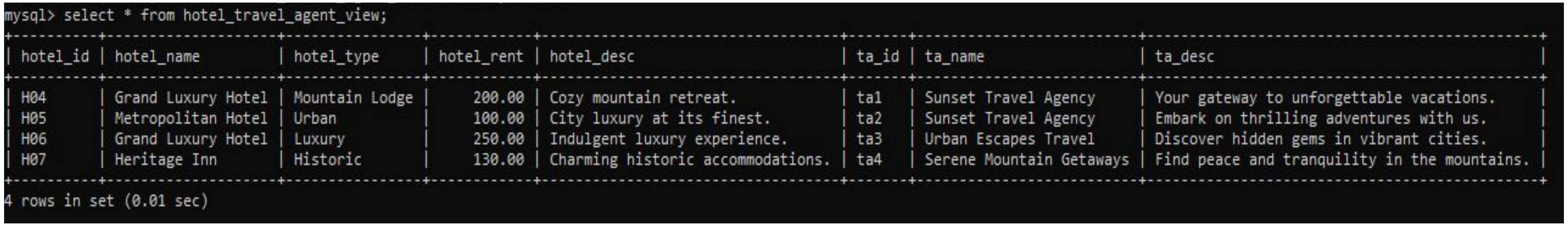
```
+---------+
| user_id |
+---------+
|       1 |
+---------+
1 row in set (0.01 sec)
```

Joins

- **Inner join for the hotel and travel_agent table.**

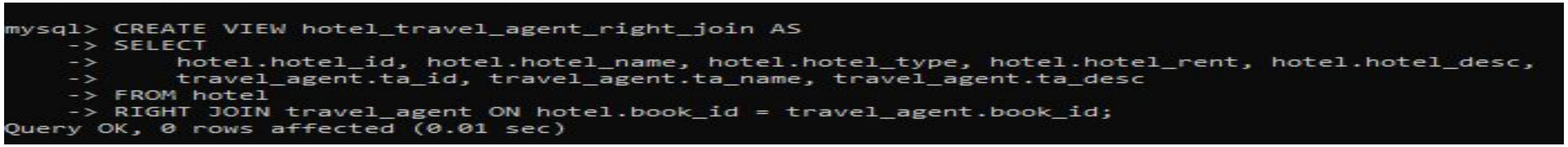


```
mysql> CREATE VIEW hotel_travel_agent_view AS
    -> SELECT
    ->     hotel.hotel_id, hotel.hotel_name, hotel.hotel_type, hotel.hotel_rent, hotel.hotel_desc,
    ->     travel_agent.ta_id, travel_agent.ta_name, travel_agent.ta_desc
    -> FROM hotel
    -> INNER JOIN travel_agent ON hotel.book_id = travel_agent.book_id;
[illegible]
```

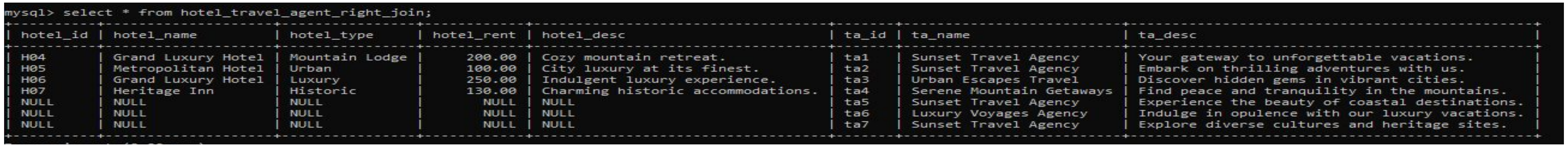


```
mysql> select * from hotel_travel_agent_view;
+----------+--------------------+-----------------+------------+-----------------------------------+-------+--------------------------+----------------------------------------------+
| hotel_id | hotel_name         | hotel_type      | hotel_rent | hotel_desc                        | ta_id | ta_name                  | ta_desc                                      |
+----------+--------------------+-----------------+------------+-----------------------------------+-------+--------------------------+----------------------------------------------+
| H04      | Grand Luxury Hotel | Mountain Lodge  |     200.00 | Cozy mountain retreat.            | ta1   | Sunset Travel Agency     | Your gateway to unforgettable vacations.     |
| H05      | Metropolitan Hotel | Urban           |     100.00 | City luxury at its finest.        | ta2   | Sunset Travel Agency     | Embark on thrilling adventures with us.      |
| H06      | Grand Luxury Hotel | Luxury          |     250.00 | Indulgent luxury experience.      | ta3   | Urban Escapes Travel     | Discover hidden gems in vibrant cities.      |
| H07      | Heritage Inn       | Historic        |     130.00 | Charming historic accommodations. | ta4   | Serene Mountain Getaways | Find peace and tranquility in the mountains. |
+----------+--------------------+-----------------+------------+-----------------------------------+-------+--------------------------+----------------------------------------------+
4 rows in set (0.01 sec)
```

- **Right Outer join for the hotel and travel_agent table.**



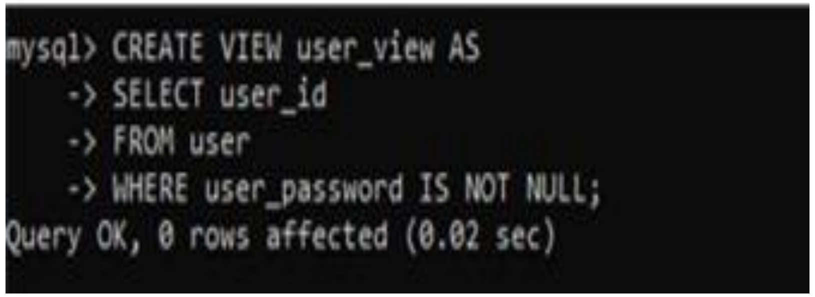
```
mysql> CREATE VIEW hotel_travel_agent_right_join AS
    -> SELECT
    ->     hotel.hotel_id, hotel.hotel_name, hotel.hotel_type, hotel.hotel_rent, hotel.hotel_desc,
    ->     travel_agent.ta_id, travel_agent.ta_name, travel_agent.ta_desc
    -> FROM hotel
    -> RIGHT JOIN travel_agent ON hotel.book_id = travel_agent.book_id;
Query OK, 0 rows affected (0.01 sec)
```



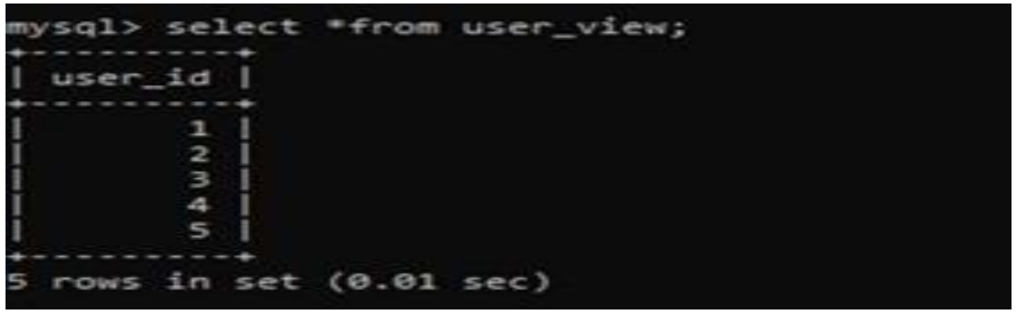
```
mysql> select * from hotel_travel_agent_right_join;
+----------+--------------------+-----------------+------------+-----------------------------------+-------+--------------------------+------------------------------------------------+
| hotel_id | hotel_name         | hotel_type      | hotel_rent | hotel_desc                        | ta_id | ta_name                  | ta_desc                                        |
+----------+--------------------+-----------------+------------+-----------------------------------+-------+--------------------------+------------------------------------------------+
| H04      | Grand Luxury Hotel | Mountain Lodge  |     200.00 | Cozy mountain retreat.            | ta1   | Sunset Travel Agency     | Your gateway to unforgettable vacations.       |
| H05      | Metropolitan Hotel | Urban           |     100.00 | City luxury at its finest.        | ta2   | Sunset Travel Agency     | Embark on thrilling adventures with us.        |
| H06      | Grand Luxury Hotel | Luxury          |     250.00 | Indulgent luxury experience.      | ta3   | Urban Escapes Travel     | Discover hidden gems in vibrant cities.        |
| H07      | Heritage Inn       | Historic        |     130.00 | Charming historic accommodations. | ta4   | Serene Mountain Getaways | Find peace and tranquility in the mountains.   |
| NULL     | NULL               | NULL            |       NULL | NULL                              | ta5   | Sunset Travel Agency     | Experience the beauty of coastal destinations. |
| NULL     | NULL               | NULL            |       NULL | NULL                              | ta6   | Luxury Voyages Agency    | Indulge in opulence with our luxury vacations. |
| NULL     | NULL               | NULL            |       NULL | NULL                              | ta7   | Sunset Travel Agency     | Explore diverse cultures and heritage sites.   |
+----------+--------------------+-----------------+------------+-----------------------------------+-------+--------------------------+------------------------------------------------+
```

Views

- View to list users who have given passwords.

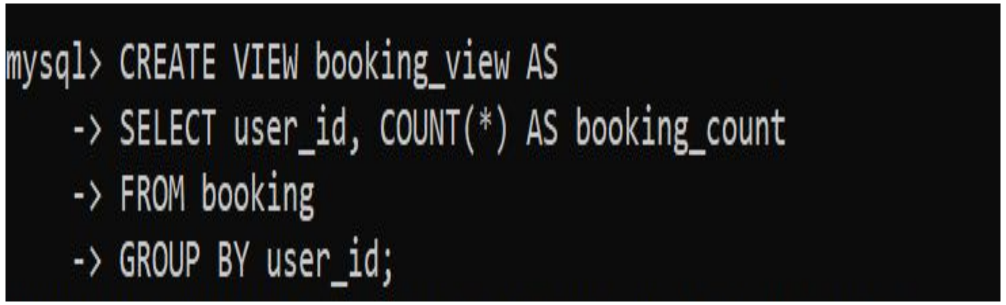


```
mysql> CREATE VIEW user_view AS
    -> SELECT user_id
    -> FROM user
    -> WHERE user_password IS NOT NULL;
Query OK, 0 rows affected (0.02 sec)
```

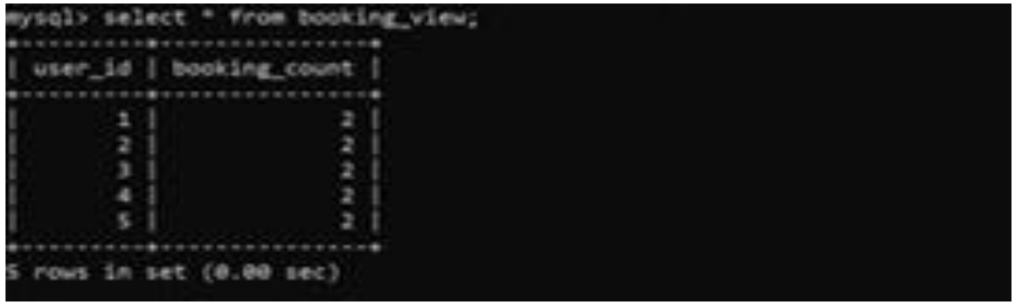


```
mysql> select *from user_view;
+---------+
| user_id |
+---------+
|       1 |
|       2 |
|       3 |
|       4 |
|       5 |
+---------+
5 rows in set (0.01 sec)
```

- View to list of users and their number of bookings.



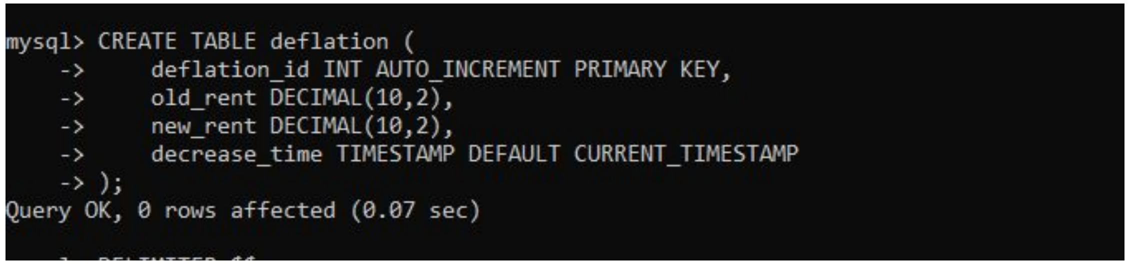
```
mysql> CREATE VIEW booking_view AS
    -> SELECT user_id, COUNT(*) AS booking_count
    -> FROM booking
    -> GROUP BY user_id;
```



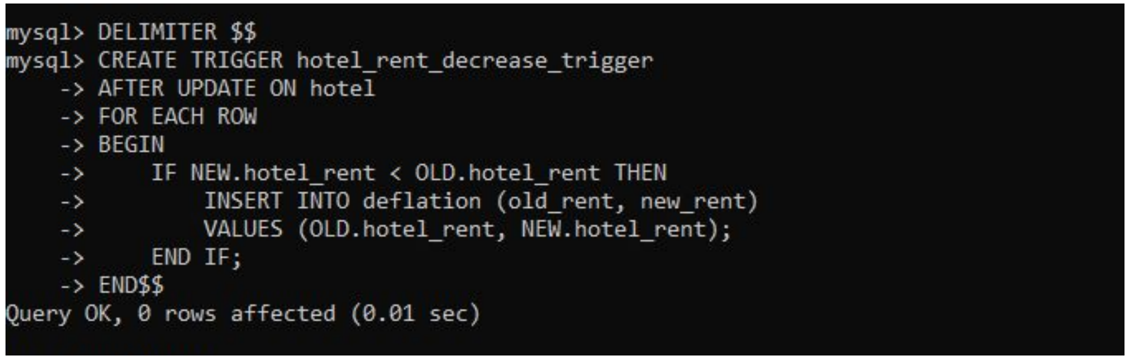
```
mysql> select * from booking_view;
+---------+---------------+
| user_id | booking_count |
+---------+---------------+
|       1 |             2 |
|       2 |             2 |
|       3 |             2 |
|       4 |             2 |
|       5 |             2 |
+---------+---------------+
5 rows in set (0.00 sec)
```

Triggers

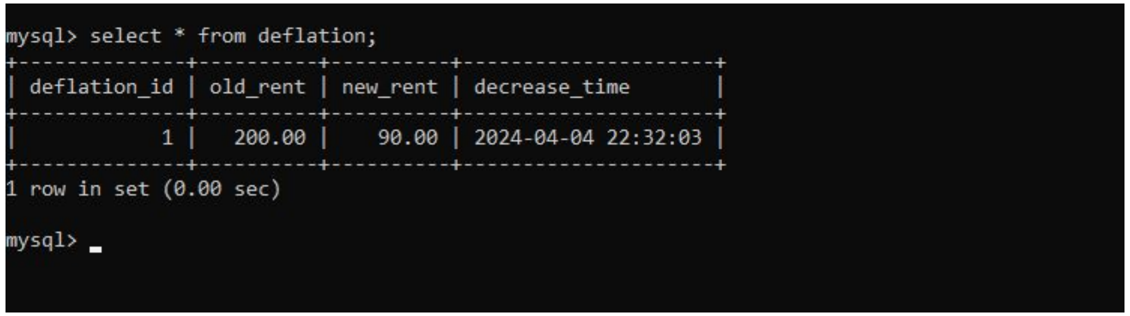
- **Trigger to alert decrease in hotel rent.**



```
mysql> CREATE TABLE deflation (
    ->     deflation_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     old_rent DECIMAL(10,2),
    ->     new_rent DECIMAL(10,2),
    ->     decrease_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.07 sec)
```



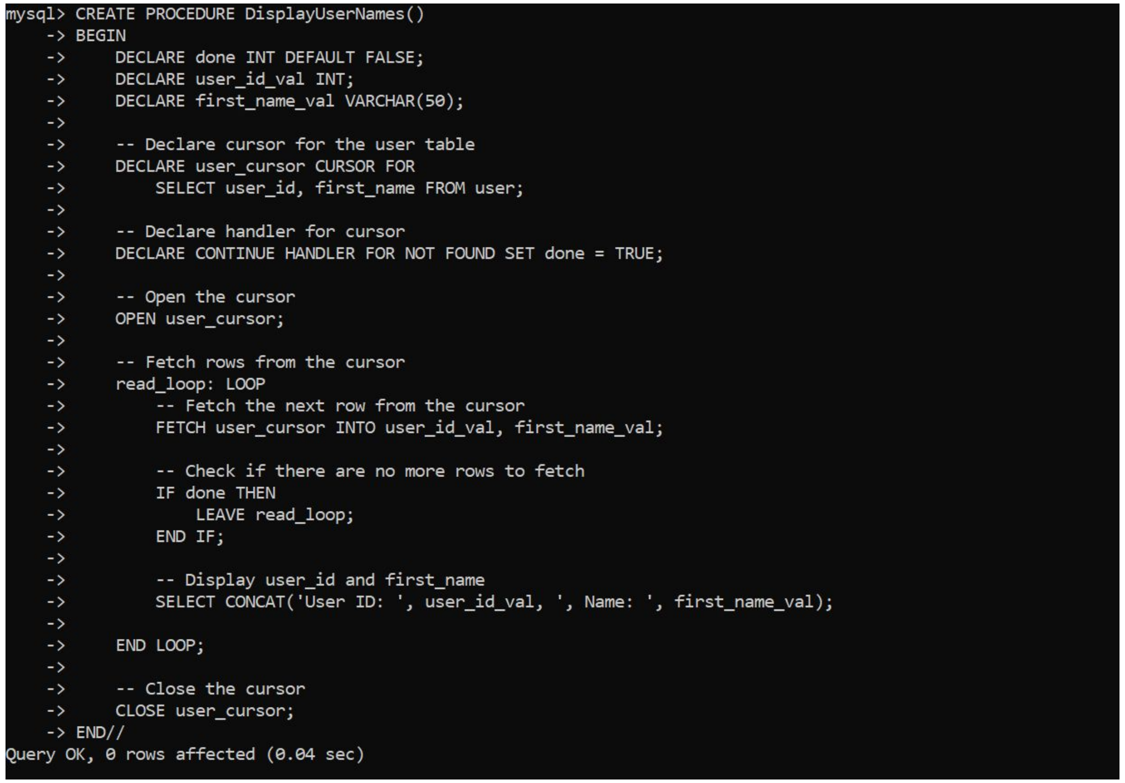
```
mysql> DELIMITER $$
mysql> CREATE TRIGGER hotel_rent_decrease_trigger
    -> AFTER UPDATE ON hotel
    -> FOR EACH ROW
    -> BEGIN
    ->     IF NEW.hotel_rent < OLD.hotel_rent THEN
    ->         INSERT INTO deflation (old_rent, new_rent)
    ->         VALUES (OLD.hotel_rent, NEW.hotel_rent);
    ->     END IF;
    -> END$$
Query OK, 0 rows affected (0.01 sec)
```



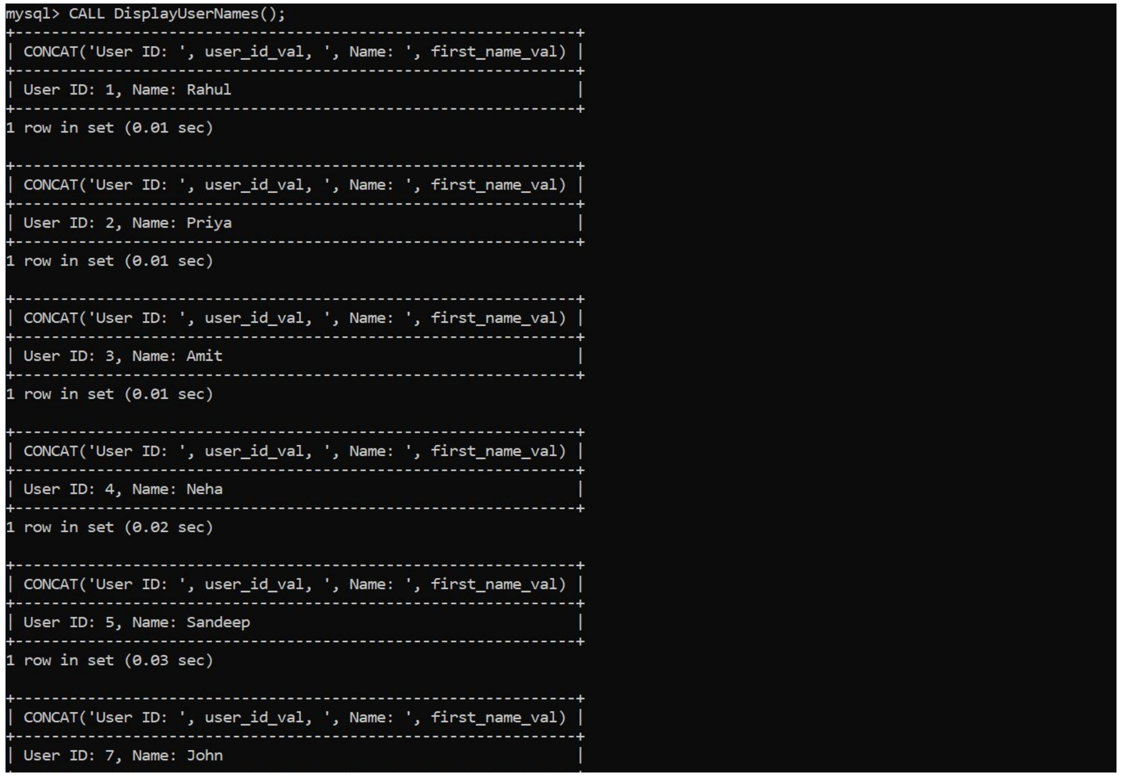
```
mysql> select * from deflation;
+--------------+----------+----------+---------------------+
| deflation_id | old_rent | new_rent | decrease_time       |
+--------------+----------+----------+---------------------+
|            1 |   200.00 |    90.00 | 2024-04-04 22:32:03 |
+--------------+----------+----------+---------------------+
1 row in set (0.00 sec)

mysql> _
```

Cursors

- **Cursor to declare Usernames.**



```
mysql> CREATE PROCEDURE DisplayUserNames()
    -> BEGIN
    ->     DECLARE done INT DEFAULT FALSE;
    ->     DECLARE user_id_val INT;
    ->     DECLARE first_name_val VARCHAR(50);
    ->
    ->     -- Declare cursor for the user table
    ->     DECLARE user_cursor CURSOR FOR
    ->         SELECT user_id, first_name FROM user;
    ->
    ->     -- Declare handler for cursor
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    ->
    ->     -- Open the cursor
    ->     OPEN user_cursor;
    ->
    ->     -- Fetch rows from the cursor
    ->     read_loop: LOOP
    ->         -- Fetch the next row from the cursor
    ->         FETCH user_cursor INTO user_id_val, first_name_val;
    ->
    ->         -- Check if there are no more rows to fetch
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         -- Display user_id and first_name
    ->         SELECT CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val);
    ->
    ->     END LOOP;
    ->
    ->     -- Close the cursor
    ->     CLOSE user_cursor;
    -> END//
Query OK, 0 rows affected (0.04 sec)
```



```
mysql> CALL DisplayUserNames();
+-----------------------------------------------------------+
| CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val) |
+-----------------------------------------------------------+
| User ID: 1, Name: Rahul                                   |
+-----------------------------------------------------------+
1 row in set (0.01 sec)

+-----------------------------------------------------------+
| CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val) |
+-----------------------------------------------------------+
| User ID: 2, Name: Priya                                   |
+-----------------------------------------------------------+
1 row in set (0.01 sec)

+-----------------------------------------------------------+
| CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val) |
+-----------------------------------------------------------+
| User ID: 3, Name: Amit                                    |
+-----------------------------------------------------------+
1 row in set (0.01 sec)

+-----------------------------------------------------------+
| CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val) |
+-----------------------------------------------------------+
| User ID: 4, Name: Neha                                    |
+-----------------------------------------------------------+
1 row in set (0.02 sec)

+-----------------------------------------------------------+
| CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val) |
+-----------------------------------------------------------+
| User ID: 5, Name: Sandeep                                 |
+-----------------------------------------------------------+
1 row in set (0.03 sec)

+-----------------------------------------------------------+
| CONCAT('User ID: ', user_id_val, ', Name: ', first_name_val) |
+-----------------------------------------------------------+
| User ID: 7, Name: John                                    |
```

Analyzing the pitfalls, identifying the dependencies, and applying normalizations

Pitfalls:

- Redundancy: In my database tables, I've strived to minimize redundancy by avoiding storing the same data in multiple places. However, there's still a risk of redundancy if data is duplicated unintentionally across different tables. Even without explicit dependencies, redundant data can lead to inconsistencies and inefficiencies if not managed properly.

- Inefficiency: While I've attempted to design my tables efficiently, inefficiencies can still arise if the schema lacks proper normalization or if data retrieval mechanisms are not optimized. Without explicit dependencies, it's essential to ensure that storage and retrieval processes are streamlined to maintain performance.

- Complexity: Despite the absence of explicit dependencies, complexity can creep into the schema if it's not carefully designed. Complexity can make the database harder to understand and maintain, increasing the likelihood of errors and hindering future development efforts. Simplifying the schema through normalization can help mitigate these complexities and improve overall manageability.

Dependencies:

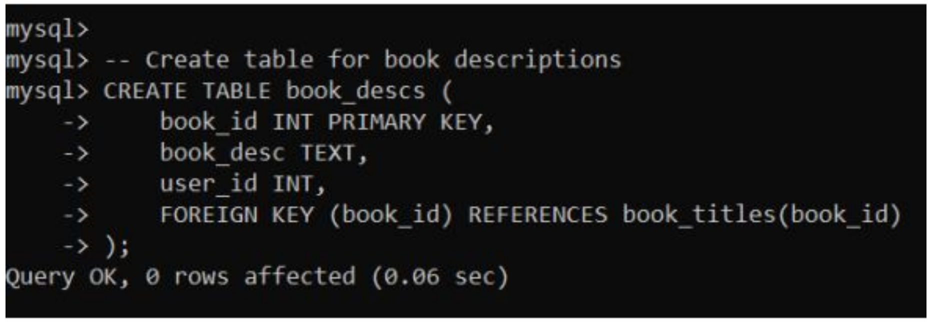
No dependencies were identified in the database schema. Each table appears to be independent of each other without any partial or transitivity dependencies.

Normalization:

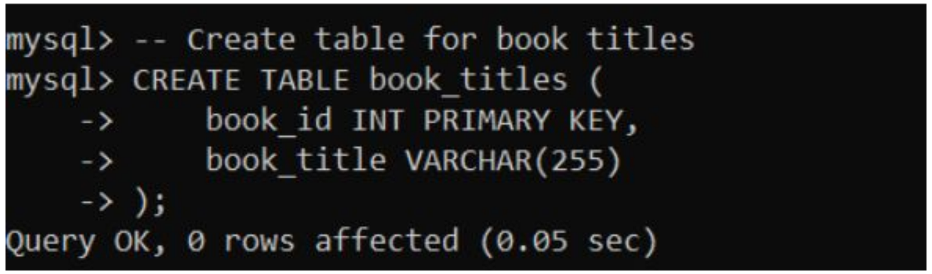
Since no dependencies were identified and the database schema does not exhibit any issues related to redundancy, inefficiency, or complexity, normalization beyond 1NF may not be necessary. However, ensuring that the database design adheres to the principles of 1NF, such as atomicity and uniqueness of values, is essential.

Analyzing the pitfalls, identifying the dependencies, and applying normalizations (Cont..)

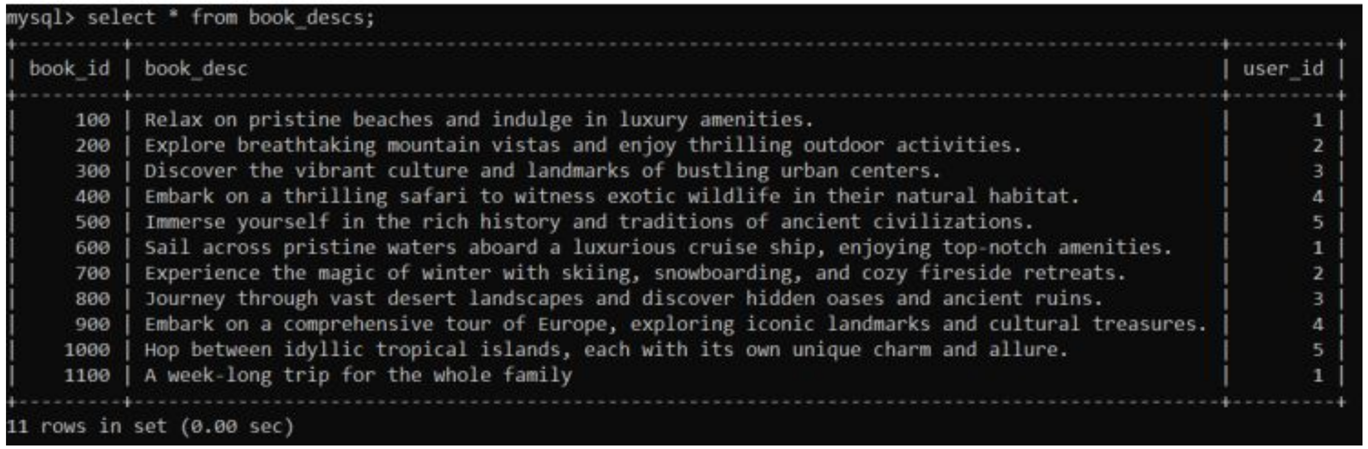
- Normalization:(2NF) In this setup, book_titles contains only the book_id and book_title, while book_descs contains book_id, book_desc, and user_id. This separation adheres to 2NF by removing any partial dependencies.



```
mysql>
mysql> -- Create table for book descriptions
mysql> CREATE TABLE book_descs (
    ->     book_id INT PRIMARY KEY,
    ->     book_desc TEXT,
    ->     user_id INT,
    ->     FOREIGN KEY (book_id) REFERENCES book_titles(book_id)
    -> );
Query OK, 0 rows affected (0.06 sec)
```



```
mysql> -- Create table for book titles
mysql> CREATE TABLE book_titles (
    ->     book_id INT PRIMARY KEY,
    ->     book_title VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.05 sec)
```



```
mysql> select * from book_descs;
+---------+-----------------------------------------------------------------------------------+---------+
| book_id | book_desc                                                                         | user_id |
+---------+-----------------------------------------------------------------------------------+---------+
|     100 | Relax on pristine beaches and indulge in luxury amenities.                        |       1 |
|     200 | Explore breathtaking mountain vistas and enjoy thrilling outdoor activities.      |       2 |
|     300 | Discover the vibrant culture and landmarks of bustling urban centers.             |       3 |
|     400 | Embark on a thrilling safari to witness exotic wildlife in their natural habitat. |       4 |
|     500 | Immerse yourself in the rich history and traditions of ancient civilizations.     |       5 |
|     600 | Sail across pristine waters aboard a luxurious cruise ship, enjoying top-notch amenities. |       1 |
|     700 | Experience the magic of winter with skiing, snowboarding, and cozy fireside retreats. |       2 |
|     800 | Journey through vast desert landscapes and discover hidden oases and ancient ruins. |       3 |
|     900 | Embark on a comprehensive tour of Europe, exploring iconic landmarks and cultural treasures. |       4 |
|    1000 | Hop between idyllic tropical islands, each with its own unique charm and allure.  |       5 |
|    1100 | A week-long trip for the whole family                                             |       1 |
+---------+-----------------------------------------------------------------------------------+---------+
11 rows in set (0.00 sec)
```

Implementation of concurrency control and recovery mechanisms

- **Atomicity**

All provided SQL transactions start with `START TRANSACTION` and end with either `COMMIT` or `ROLLBACK`. This ensures that either all the operations within the transaction are completed successfully (committed) or none of them are (rolled back). For example:

- When updating `book_title`, `hotel_rent`, or `ta_desc`, each operation is part of a transaction and will be either fully completed or fully rolled back.

- If any error occurs during the execution of an operation, the transaction can be rolled back to the savepoint, ensuring that the database remains in a consistent state.

Consistency

The operations maintain the consistency of the database by following predefined constraints and rules. For example:

- The update operations (`book_title` and `hotel_rent`) ensure that the updated data remains valid according to the schema and any business rules.

- The insert operation (`ta_desc`) ensures that new data adheres to the constraints defined for the `travel_agent` table.

Isolation

Each transaction operates independently of other transactions. Transactions are executed in isolation, meaning that the intermediate states of transactions are not visible to other transactions until they are committed. For example:

- The savepoints provide a mechanism for breaking transactions into smaller parts, allowing for finer control over transaction boundaries and rollback points without affecting other transactions.

Durability

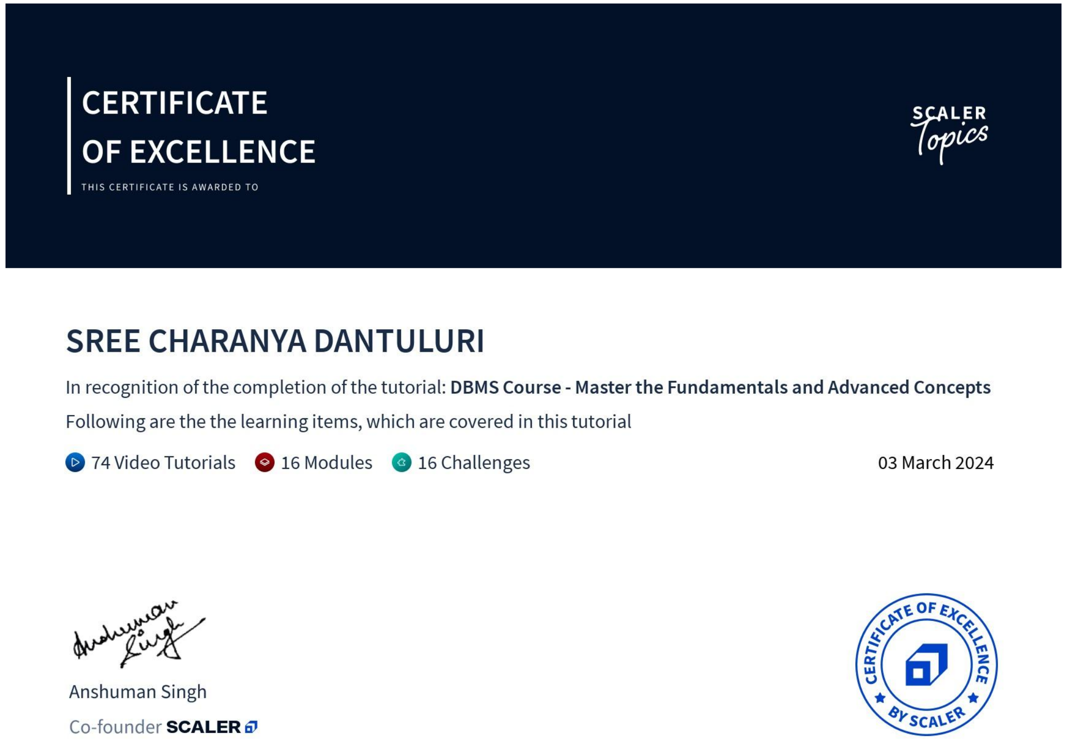
Once a transaction is committed, its changes are permanently saved in the database, even in the event of system failure. For example:

- After a successful `COMMIT`, any changes made to the database (such as updates to `book_title` and `hotel_rent` or insertions into `travel_agent`) are durably persisted.

- If a transaction is rolled back due to an error or explicit rollback command, changes are not persisted, ensuring that the database remains in a consistent state.

Overall, the provided data operations demonstrate adherence to the ACID properties, ensuring reliability, consistency, and integrity of the database transactions.

Attach the Real Time project certificate / Online course certificate


CERTIFICATE
OF EXCELLENCE
THIS CERTIFICATE IS AWARDED TO

SCALER Topics

SREE CHARANYA DANTULURI

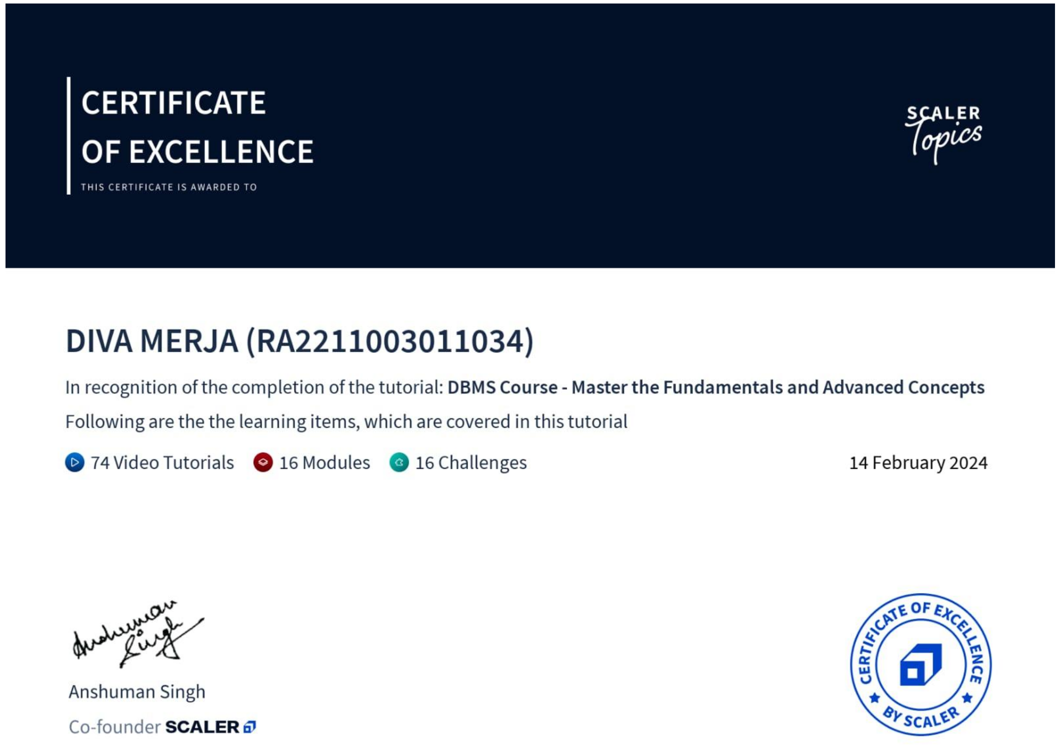
In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges 03 March 2024

Anshuman Singh
Co-founder **SCALER**

CERTIFICATE OF EXCELLENCE BY SCALER


CERTIFICATE
OF EXCELLENCE
THIS CERTIFICATE IS AWARDED TO

SCALER Topics

DIVA MERJA (RA2211003011034)

In recognition of the completion of the tutorial: **DBMS Course - Master the Fundamentals and Advanced Concepts**

Following are the the learning items, which are covered in this tutorial

74 Video Tutorials 16 Modules 16 Challenges 14 February 2024

Anshuman Singh
Co-founder **SCALER**

CERTIFICATE OF EXCELLENCE BY SCALER