**SVKM's NMIMS**

**School of Technology Management & Engineering (Indore Campus)**

**Computer Engineering Department (B Tech/MBATech CE and B Tech AIDS Sem IV)**

**Database Management System**

# Project Report

| | | |
|---|---|---|
| Program | B-Tech (Computer Engineering) | |
| Semester | 4th | |
| Name of the Project: | Travel Itinerary Database | |
| | | |
| Details of Project Members | | |
| Batch | Roll No. | Name |
| 2 | D-47 | Mahi Jaiswal |
| 2 | D-68 | Rahul Sinha |
| 2 | D-74 | Rohit Sinha |
| Date of Submission: 11-04-25 | | |

**Contribution of each project Members:**

| Roll No. | Name: | Contribution |
|---|---|---|
| D-047 | Mahi Jaiswal | Documentation and Performing SQL queries |
| D-068 | Rahul Sinha | ER Diagram, Creating Database and Performing SQL queries |
| D-074 | Rohit Sinha | ER Diagram to Relational Schema Conversion and Performing SQL queries |

**Github link of your project:** **https://github.com/mahijaiswal-10/DBMS-PROJECT-**

**Note:**

1. Create a readme file if you have multiple files
2. All files must be properly named (Example:R004_DBMSProject)
3. Submit all relevant files of your work ( Report, all SQL files, Any other files)

4. **Plagiarism is highly discouraged (Your report will be checked for plagiarism)**

**Rubrics for the Project evaluation:**

| | |
|---|---|
| First phase of evaluation:<br>Innovative Ideas (5 Marks)<br>Design and Partial implementation (5 Marks) | 10 marks |
| Final phase of evaluation<br>Implementation, presentation and viva,<br>Self-Learning and Learning Beyond classroom | 10 marks |

# Project Report

## Selected Topic

**by**

**Mahi Jaiswal, Roll number: D47**

**Rahul Sinha, Roll number: D68**

**Rohit Sinha, Roll number: D74**

## Course: DBMS

# AY: 2024-25

## Table of Contents

# I. Storyline

Our project is a database management system designed to support an interactive travel planning platform that allows users to design and customize their travel experiences. The system efficiently stores, manages, and retrieves data related to travel packages, personalized itineraries, and user-generated blogs. Each user has a profile within the database, which records their activities, including writing blogs, creating itineraries, and selecting travel packages. Additionally, the database tracks user interactions, such as pages visited and engagement metrics, to enable personalized recommendations and insights. Built with a focus on data integrity, optimized query performance, and scalability, the system ensures reliable and efficient handling of increasing user interactions. It is structured to support seamless data operations while maintaining consistency and accuracy, allowing the platform to deliver a responsive and dynamic user experience.

# II. Components of Database Design

Entities and Attributes

**1. Users**

This part stores information about people using the platform:

- A unique ID for each user
- Their username (what they go by on the platform)
- Their email address
- A safe, hidden version of their password
- A profile picture
- The date they created their account

---

**2. Blogs**

This part keeps track of blogs written by users:

- A unique ID for each blog
- The user who wrote it
- The title and full content of the blog
- The date the blog was created and last updated

---

**3. Itineraries**

This part saves travel plans made by users:

- A unique ID for each itinerary
- The user who made it
- The name and a short summary of the trip
- When it was created and last updated

---

**4. Itinerary Items**

These are the detailed steps or activities inside a travel plan:

- A unique ID for each activity
- Which itinerary it belongs to
- The order in which the activity happens
- A short description of the activity
- Where and when it takes place

---

**5. Packages**

These are travel offers or plans available for users:

- A unique ID for each package
- The name and detailed description
- The price and number of days
- The dates when this package can be booked

---

**6. Destinations**

This part stores details about travel places:

- A unique ID for each destination
- The name of the place
- What type of place it is (like a city or adventure spot)
- A description and a picture of it

---

**7. User Visits**

This keeps a record when a user checks or visits a destination:

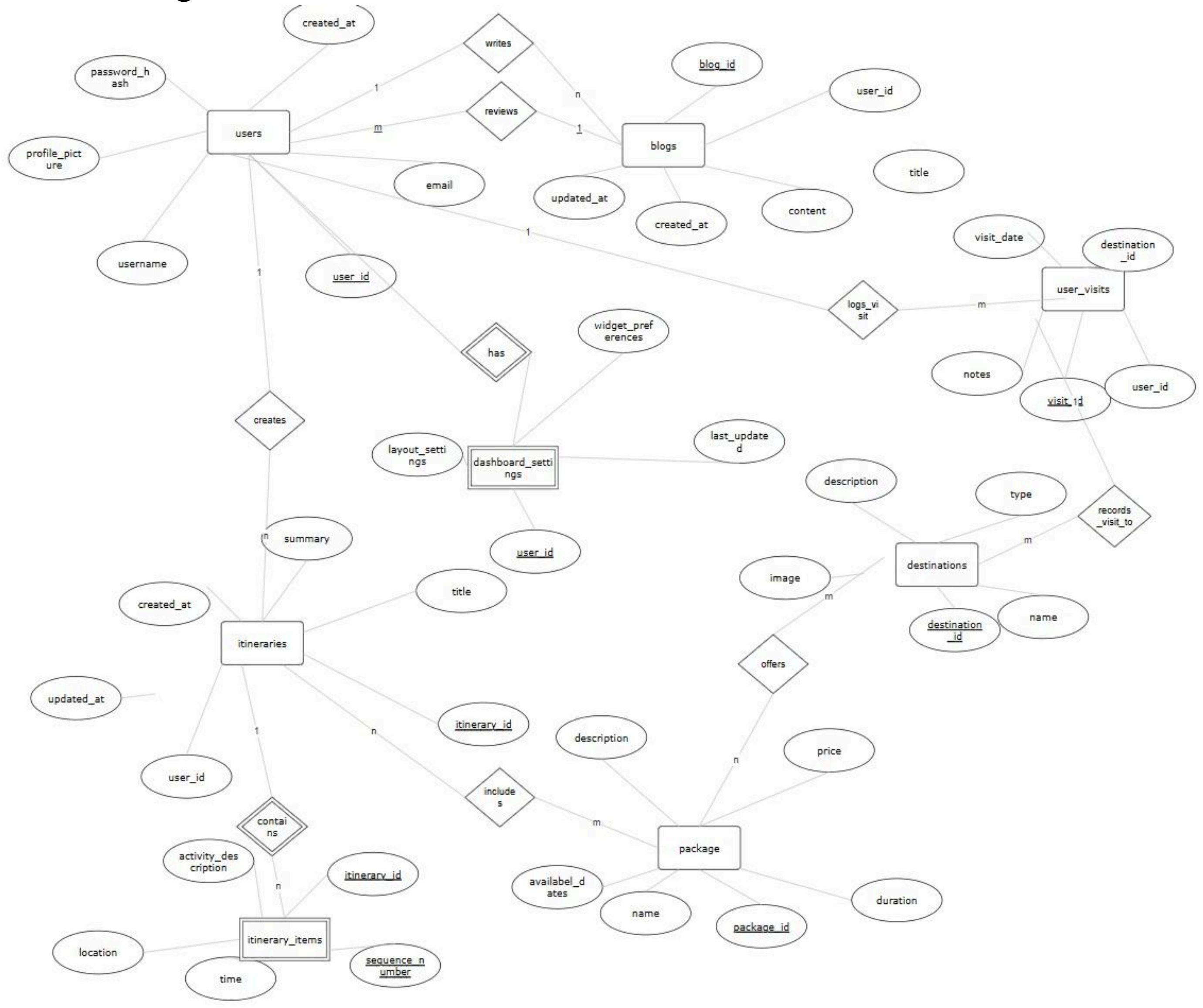
- A unique ID for each visit
- The user and the place they visited
- The date of the visit
- Any notes they added

Relationships and Cardinalities

- **Users** can **write** multiple **blogs** (1:N).
- **Users** can **create** multiple **itineraries** (1:N).
- **Each itinerary** contains **multiple itinerary items** (1:N).
- **Users** can have **one** dashboard setting (1:1).
- **Users** can **log visits** to multiple destinations (M:N).
- **Packages** can **include** multiple destinations (M:N).

## III. Entity Relationship Diagram

Our ER Diagram is as follows:



## IV. Relational Model

Based on the ER diagram, we derive the following relational schema that structures the data efficiently while maintaining relationships between different entities.

1. **Users Table (Strong Entity)**

- **Attributes:**
  Users(user_id (PK), username, email, password_hash, profile_picture, created_at)

**2. Blogs Table (Entity with Relationship to Users)**

- **Attributes:**
  Blogs(blog_id (PK), user_id (FK), title, content, created_at, updated_at)
- **Foreign Key:**
  user_id references Users(user_id)
- **Mapping Relationship:**
  - **(1,N) Relationship**: Each **User** can write **many Blogs**, but each **Blog** is written by one **User**.

**3. Itineraries Table (Entity with Relationship to Users)**

- **Attributes:**
  Itineraries(itinerary_id (PK), user_id (FK), title, summary, created_at, updated_at)
- **Foreign Key:**
  user_id references Users(user_id)
- **Mapping Relationship:**
  - **(1,N) Relationship**: Each **User** can create **multiple Itineraries**, but each **Itinerary** belongs to one **User**.

**4. Itinerary Items Table (Weak Entity, Dependent on Itineraries)**

- **Attributes:**
  Itinerary_Items(item_id (PK), itinerary_id (FK), sequence_number, activity_description, location, time)
- **Foreign Key:**
  itinerary_id references Itineraries(itinerary_id)
- **Mapping Relationship:**
  - **(1,N) Relationship**: Each **Itinerary** can have **multiple Items**, but each **Item** belongs to one **Itinerary**.

**5. Packages Table (Independent Entity)**

- **Attributes:**
  Packages(package_id (PK), name, description, price, duration, available_dates)

**6. Package Destinations Table (Many-to-Many Relationship Resolver)**

- **Attributes:**
  Package_Destinations(package_id (PK, FK), destination_id (PK, FK))
- **Foreign Keys:**
  - package_id references Packages(package_id)
  - destination_id references Destinations(destination_id)
- **Mapping Relationship:**

- **(M,N) Relationship**: A **Package** can include **multiple Destinations**, and a **Destination** can be part of **multiple Packages**.

**7. Destinations Table (Independent Entity)**

- **Attributes:**
  Destinations(destination_id (PK), name, type, description, image)

**8. User Visits Table (Many-to-Many Relationship Resolver)**

- **Attributes:**
  User_Visits(visit_id (PK), user_id (FK), destination_id (FK), visit_date, notes)
- **Foreign Keys:**
  - user_id references Users(user_id)
  - destination_id references Destinations(destination_id)
- **Mapping Relationship:**
  - **(M,N) Relationship**: A **User** can visit **multiple Destinations**, and a **Destination** can be visited by **multiple Users**.

**9. Dashboard Settings Table (Dependent on Users)**

- **Attributes:**
  Dashboard_Settings(dashboard_id (PK), user_id (FK), layout_settings, widget_preferences, last_updated)
- **Foreign Key:**
  user_id references Users(user_id)
- **Mapping Relationship:**
  - **(1,1) Relationship**: Each **User** has exactly **one Dashboard Setting**.

# V. Normalization

## 1NF:

As Packages table may contain multiple dates, we decompose this table as following:

**Packages Table**

- Initial Attributes:
  Packages(package_id (PK), name, description, price, duration, available_dates)
- Problem Identified:
  available_dates contains multiple values
- Decomposition:

1. Removed available_dates
2. Created Package_Dates table

- Final Attributes:
  Packages(package_id (PK), name, description, price, duration)

**Package_Dates Table  (new table)**

- Attributes:
  Package_Dates(package_id (PK, FK), available_date (PK))
- Foreign Keys:
  package_id references Packages(package_id)
- Mapping Relationship:
  (1,N) Relationship: Each Package can have multiple available dates

## 2NF:

All tables are already in 2NF.

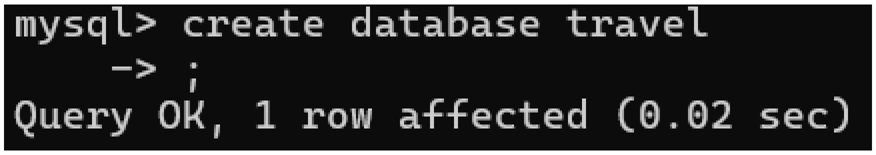
## 3NF:

All tables are already in 3NF.
*Notes/Assumptions:*
***Username is not unique** by design, as it's treated like a display name. Multiple users can share the same username (e.g., "IndoreTraveler"), while* user_id *serves as the true unique identifier.*

***Email can be reused** during testing to simplify development. For example, using* test@example.com *for multiple accounts speeds up testing. Uniqueness can be enforced later in production.*

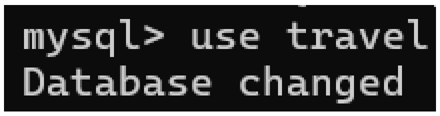
# VI. SQL Queries

**Create Database:**

1. 

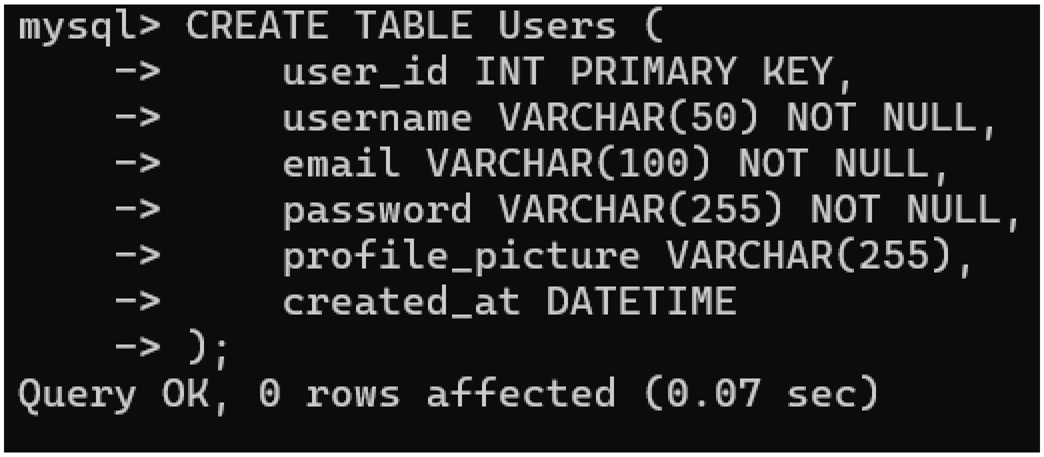
```
mysql> create database travel
    -> ;
Query OK, 1 row affected (0.02 sec)
```

**Use Database:**

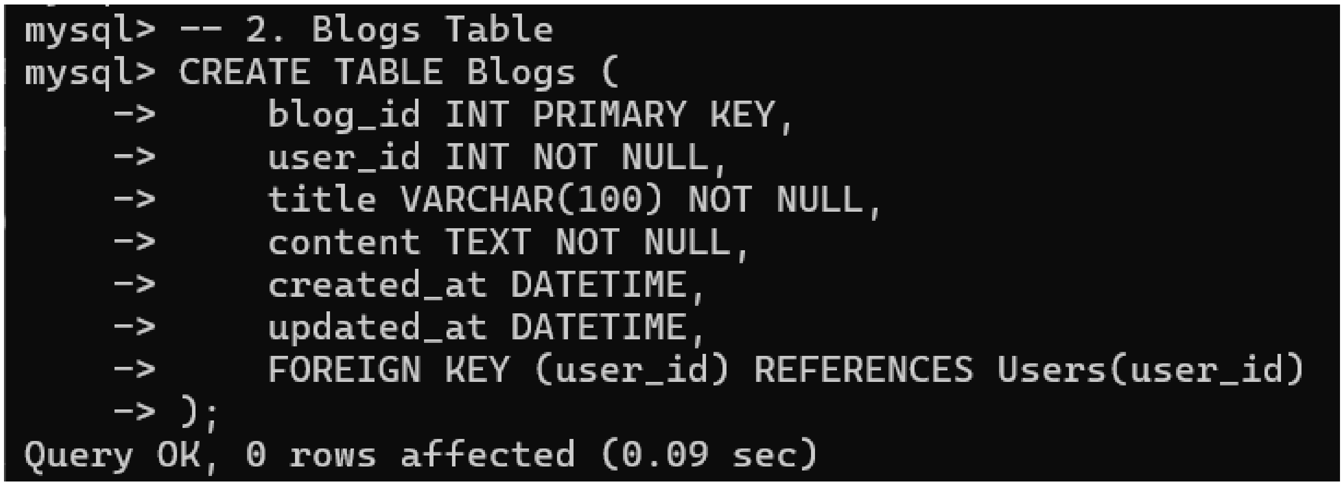
2. 

```
mysql> use travel
Database changed
```

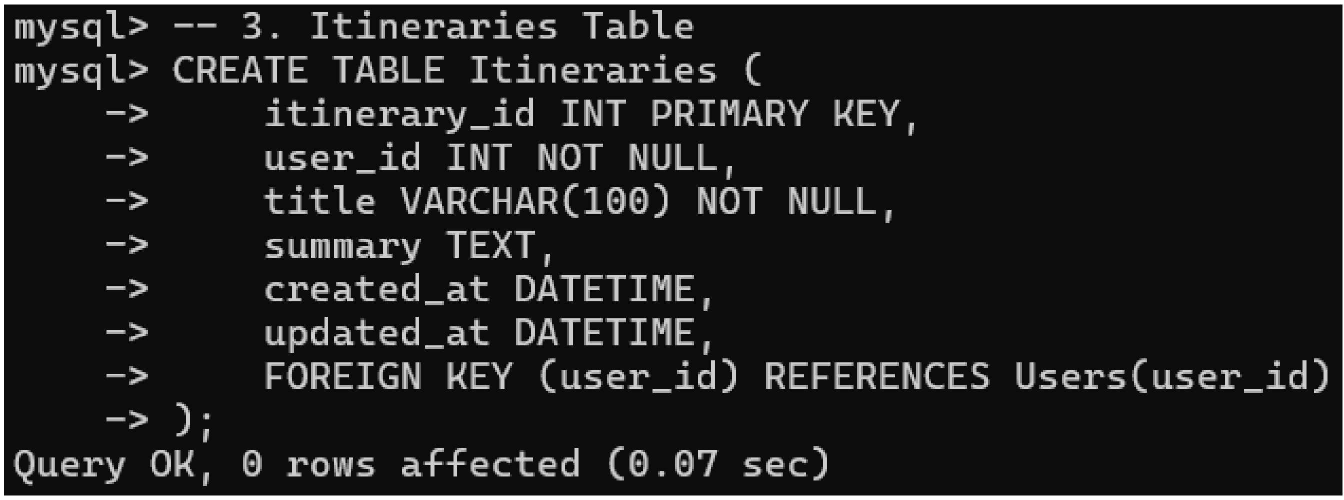
**Create tables:**

3. 

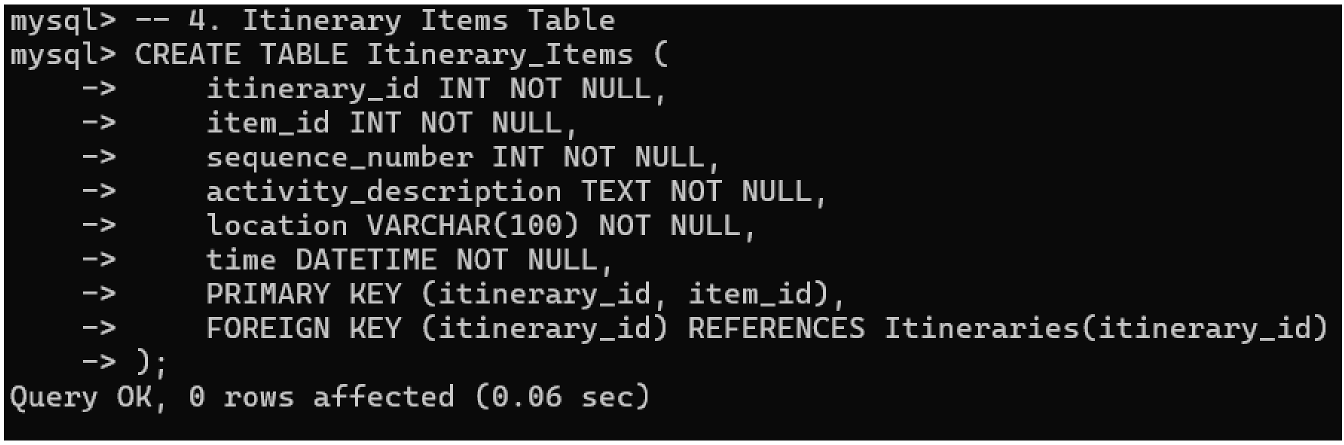
```
mysql> CREATE TABLE Users (
    ->     user_id INT PRIMARY KEY,
    ->     username VARCHAR(50) NOT NULL,
    ->     email VARCHAR(100) NOT NULL,
    ->     password VARCHAR(255) NOT NULL,
    ->     profile_picture VARCHAR(255),
    ->     created_at DATETIME
    -> );
Query OK, 0 rows affected (0.07 sec)
```

4. 

```
mysql> -- 2. Blogs Table
mysql> CREATE TABLE Blogs (
    ->     blog_id INT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     title VARCHAR(100) NOT NULL,
    ->     content TEXT NOT NULL,
    ->     created_at DATETIME,
    ->     updated_at DATETIME,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.09 sec)
```

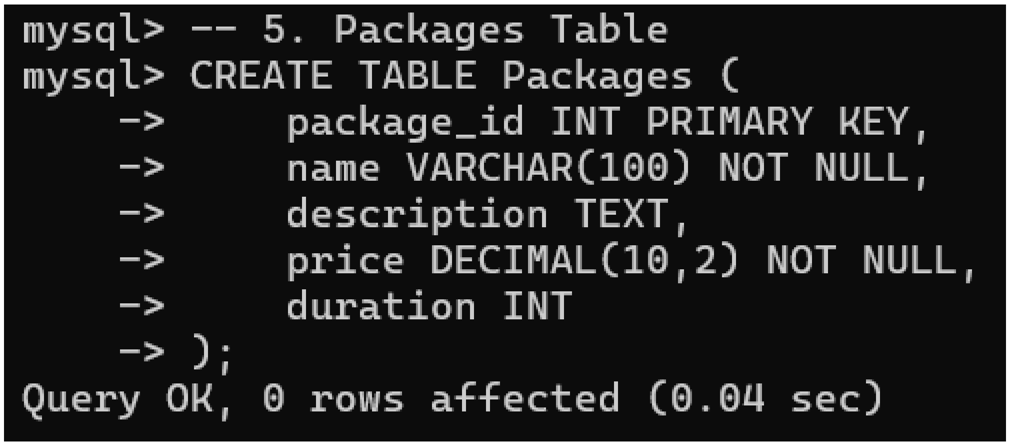
5. 

```
mysql> -- 3. Itineraries Table
mysql> CREATE TABLE Itineraries (
    ->     itinerary_id INT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     title VARCHAR(100) NOT NULL,
    ->     summary TEXT,
    ->     created_at DATETIME,
    ->     updated_at DATETIME,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.07 sec)
```



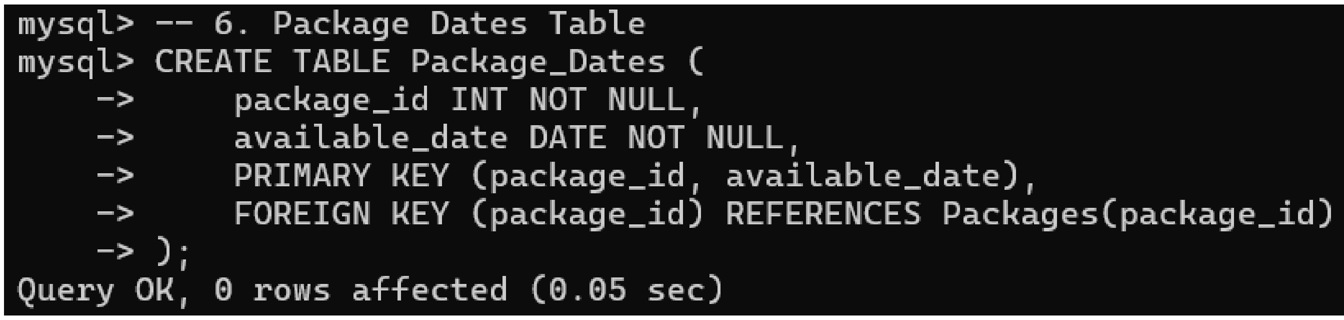
```
mysql> -- 4. Itinerary Items Table
mysql> CREATE TABLE Itinerary_Items (
    ->     itinerary_id INT NOT NULL,
    ->     item_id INT NOT NULL,
    ->     sequence_number INT NOT NULL,
    ->     activity_description TEXT NOT NULL,
    ->     location VARCHAR(100) NOT NULL,
    ->     time DATETIME NOT NULL,
    ->     PRIMARY KEY (itinerary_id, item_id),
    ->     FOREIGN KEY (itinerary_id) REFERENCES Itineraries(itinerary_id)
    -> );
Query OK, 0 rows affected (0.06 sec)
```

6.



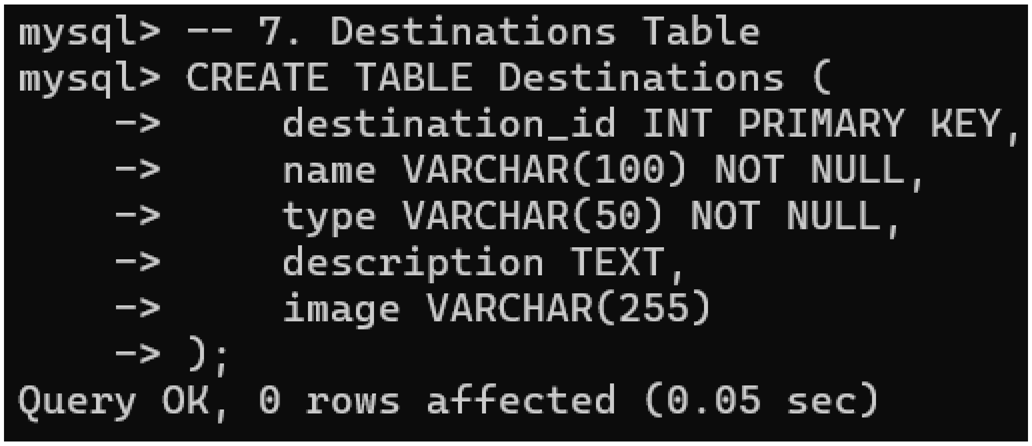
```
mysql> -- 5. Packages Table
mysql> CREATE TABLE Packages (
    ->     package_id INT PRIMARY KEY,
    ->     name VARCHAR(100) NOT NULL,
    ->     description TEXT,
    ->     price DECIMAL(10,2) NOT NULL,
    ->     duration INT
    -> );
Query OK, 0 rows affected (0.04 sec)
```

7.



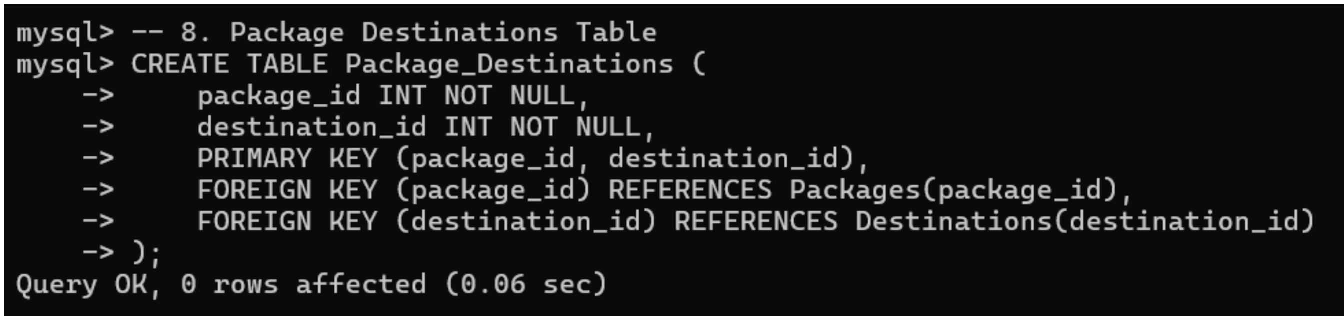
```
mysql> -- 6. Package Dates Table
mysql> CREATE TABLE Package_Dates (
    ->     package_id INT NOT NULL,
    ->     available_date DATE NOT NULL,
    ->     PRIMARY KEY (package_id, available_date),
    ->     FOREIGN KEY (package_id) REFERENCES Packages(package_id)
    -> );
Query OK, 0 rows affected (0.05 sec)
```

8.

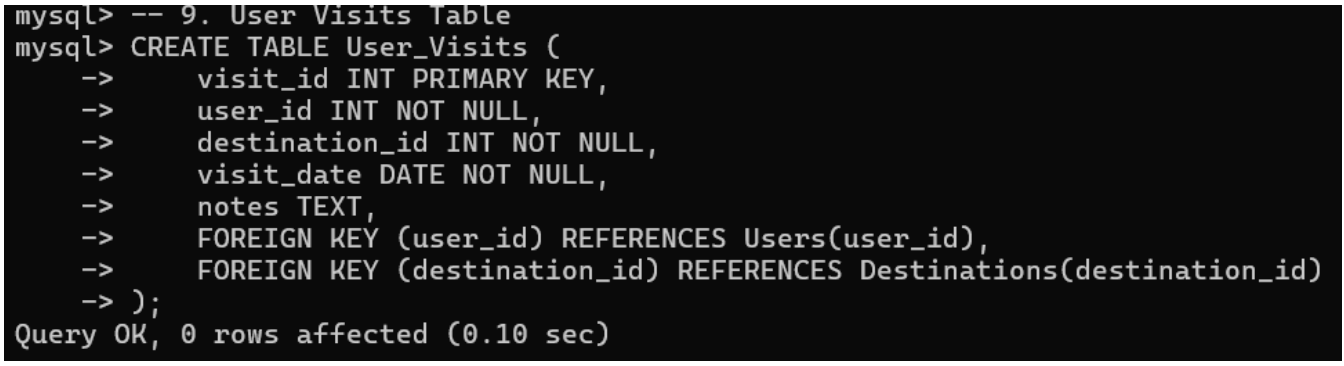


```
mysql> -- 7. Destinations Table
mysql> CREATE TABLE Destinations (
    ->     destination_id INT PRIMARY KEY,
    ->     name VARCHAR(100) NOT NULL,
    ->     type VARCHAR(50) NOT NULL,
    ->     description TEXT,
    ->     image VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.05 sec)
```

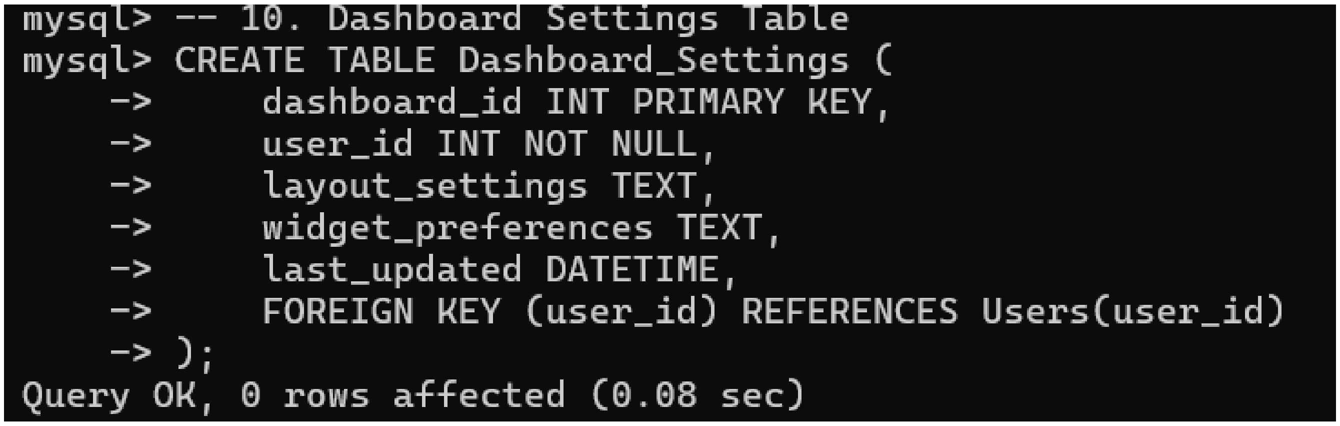
9.

10. 

```
mysql> -- 8. Package Destinations Table
mysql> CREATE TABLE Package_Destinations (
    ->     package_id INT NOT NULL,
    ->     destination_id INT NOT NULL,
    ->     PRIMARY KEY (package_id, destination_id),
    ->     FOREIGN KEY (package_id) REFERENCES Packages(package_id),
    ->     FOREIGN KEY (destination_id) REFERENCES Destinations(destination_id)
    -> );
Query OK, 0 rows affected (0.06 sec)
```

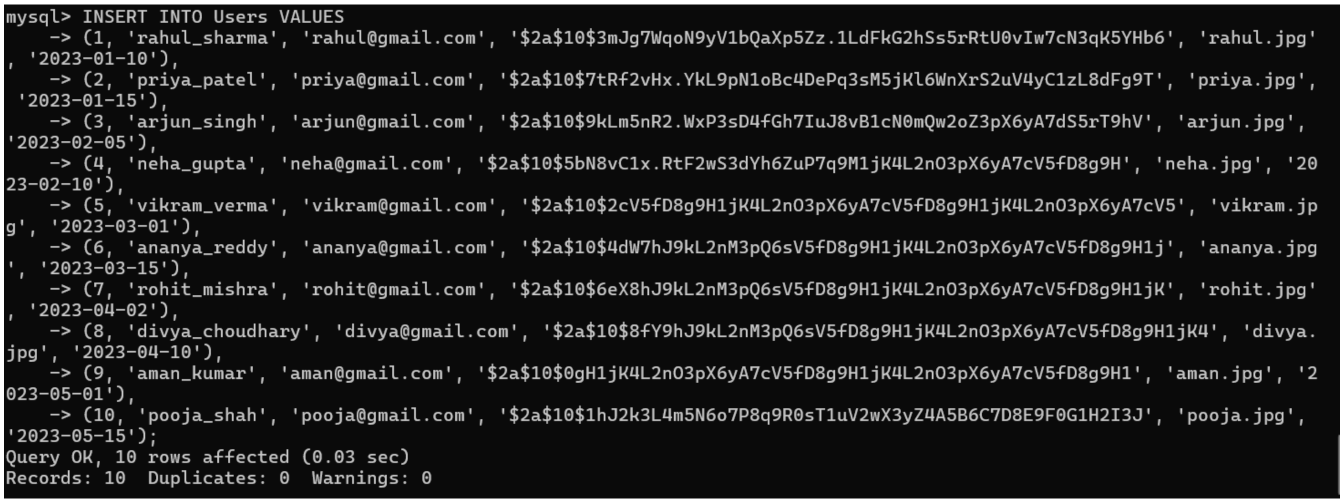
11. 

```
mysql> -- 9. User Visits Table
mysql> CREATE TABLE User_Visits (
    ->     visit_id INT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     destination_id INT NOT NULL,
    ->     visit_date DATE NOT NULL,
    ->     notes TEXT,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id),
    ->     FOREIGN KEY (destination_id) REFERENCES Destinations(destination_id)
    -> );
Query OK, 0 rows affected (0.10 sec)
```

12. 

```
mysql> -- 10. Dashboard Settings Table
mysql> CREATE TABLE Dashboard_Settings (
    ->     dashboard_id INT PRIMARY KEY,
    ->     user_id INT NOT NULL,
    ->     layout_settings TEXT,
    ->     widget_preferences TEXT,
    ->     last_updated DATETIME,
    ->     FOREIGN KEY (user_id) REFERENCES Users(user_id)
    -> );
Query OK, 0 rows affected (0.08 sec)
```

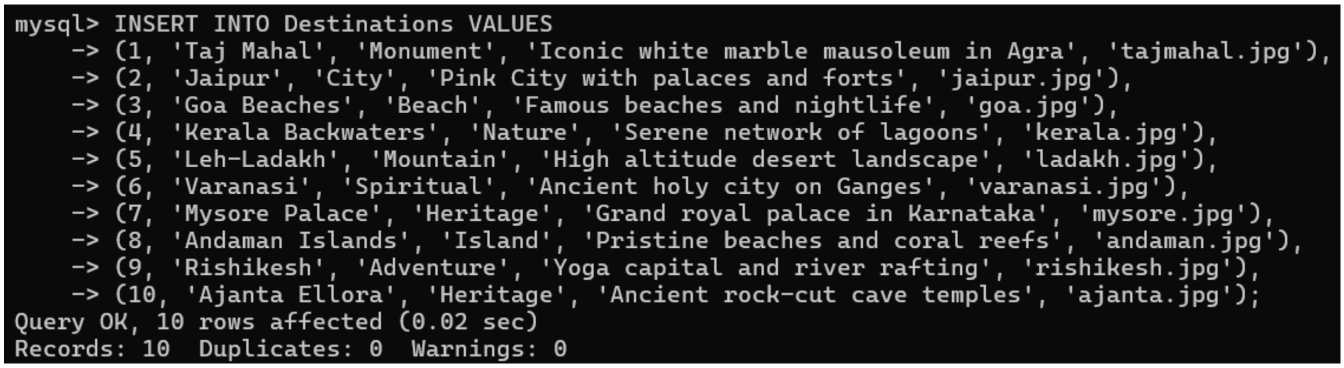
**Inserting values:**
**Users table:**



```
mysql> INSERT INTO Users VALUES
    -> (1, 'rahul_sharma', 'rahul@gmail.com', '$2a$10$3mJg7WqoN9yV1bQaXp5Zz.1LdFkG2hSs5rRtU0vIw7cN3qK5YHb6', 'rahul.jpg'
, '2023-01-10'),
    -> (2, 'priya_patel', 'priya@gmail.com', '$2a$10$7tRf2vHx.YkL9pN1oBc4DePq3sM5jKl6WnXrS2uV4yC1zL8dFg9T', 'priya.jpg',
 '2023-01-15'),
    -> (3, 'arjun_singh', 'arjun@gmail.com', '$2a$10$9kLm5nR2.WxP3sD4fGh7IuJ8vB1cN0mQw2oZ3pX6yA7dS5rT9hV', 'arjun.jpg',
'2023-02-05'),
    -> (4, 'neha_gupta', 'neha@gmail.com', '$2a$10$5bN8vC1x.RtF2wS3dYh6ZuP7q9M1jK4L2nO3pX6yA7cV5fD8g9H', 'neha.jpg', '20
23-02-10'),
    -> (5, 'vikram_verma', 'vikram@gmail.com', '$2a$10$2cV5fD8g9H1jK4L2nO3pX6yA7cV5fD8g9H1jK4L2nO3pX6yA7cV5', 'vikram.jp
g', '2023-03-01'),
    -> (6, 'ananya_reddy', 'ananya@gmail.com', '$2a$10$4dW7hJ9kL2nM3pQ6sV5fD8g9H1jK4L2nO3pX6yA7cV5fD8g9H1j', 'ananya.jpg
', '2023-03-15'),
    -> (7, 'rohit_mishra', 'rohit@gmail.com', '$2a$10$6eX8hJ9kL2nM3pQ6sV5fD8g9H1jK4L2nO3pX6yA7cV5fD8g9H1jK', 'rohit.jpg'
, '2023-04-02'),
    -> (8, 'divya_choudhary', 'divya@gmail.com', '$2a$10$8fY9hJ9kL2nM3pQ6sV5fD8g9H1jK4L2nO3pX6yA7cV5fD8g9H1jK4', 'divya.
jpg', '2023-04-10'),
    -> (9, 'aman_kumar', 'aman@gmail.com', '$2a$10$0gH1jK4L2nO3pX6yA7cV5fD8g9H1jK4L2nO3pX6yA7cV5fD8g9H1', 'aman.jpg', '2
023-05-01'),
    -> (10, 'pooja_shah', 'pooja@gmail.com', '$2a$10$1hJ2k3L4m5N6o7P8q9R0sT1uV2wX3yZ4A5B6C7D8E9F0G1H2I3J', 'pooja.jpg',
'2023-05-15');
Query OK, 10 rows affected (0.03 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

13.

**Destinations table:**

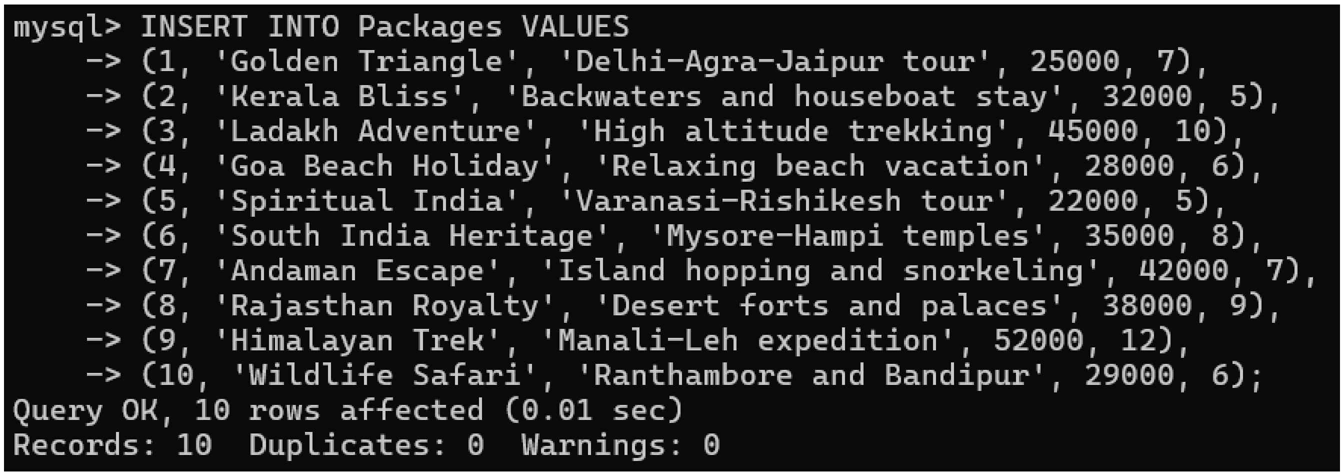


```
mysql> INSERT INTO Destinations VALUES
    -> (1, 'Taj Mahal', 'Monument', 'Iconic white marble mausoleum in Agra', 'tajmahal.jpg'),
    -> (2, 'Jaipur', 'City', 'Pink City with palaces and forts', 'jaipur.jpg'),
    -> (3, 'Goa Beaches', 'Beach', 'Famous beaches and nightlife', 'goa.jpg'),
    -> (4, 'Kerala Backwaters', 'Nature', 'Serene network of lagoons', 'kerala.jpg'),
    -> (5, 'Leh-Ladakh', 'Mountain', 'High altitude desert landscape', 'ladakh.jpg'),
    -> (6, 'Varanasi', 'Spiritual', 'Ancient holy city on Ganges', 'varanasi.jpg'),
    -> (7, 'Mysore Palace', 'Heritage', 'Grand royal palace in Karnataka', 'mysore.jpg'),
    -> (8, 'Andaman Islands', 'Island', 'Pristine beaches and coral reefs', 'andaman.jpg'),
    -> (9, 'Rishikesh', 'Adventure', 'Yoga capital and river rafting', 'rishikesh.jpg'),
    -> (10, 'Ajanta Ellora', 'Heritage', 'Ancient rock-cut cave temples', 'ajanta.jpg');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

14.

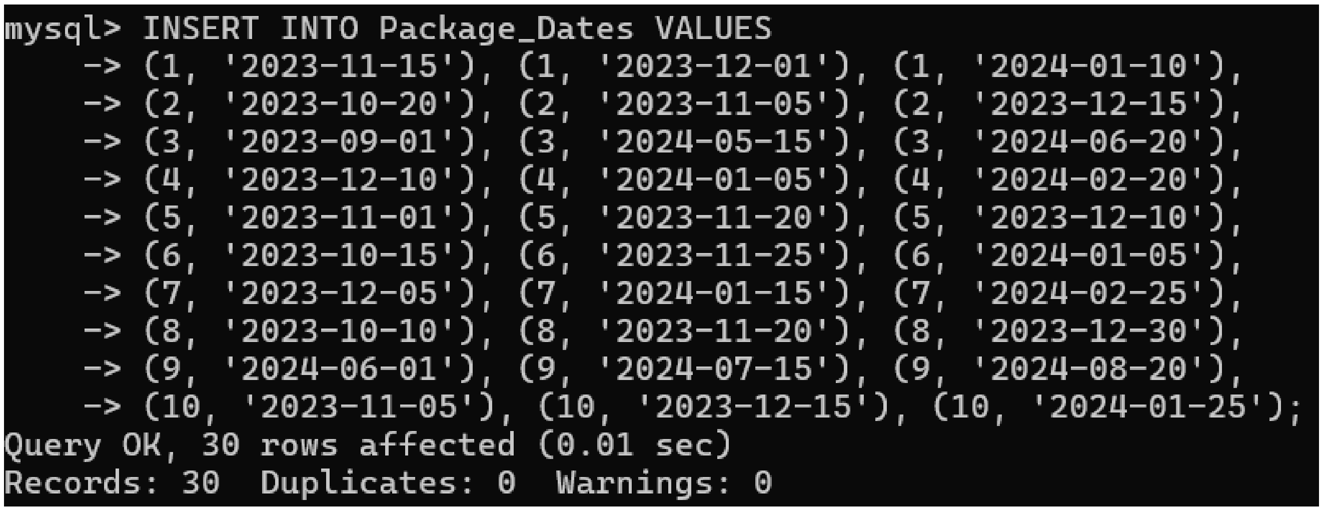
**Packages table:**

15.



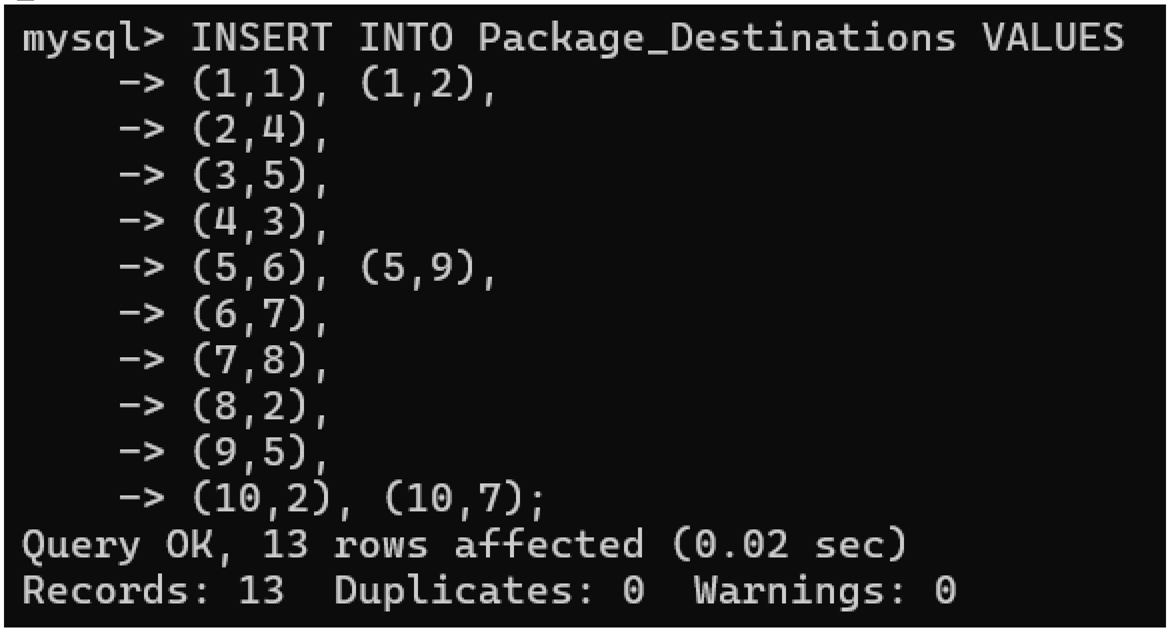
```
mysql> INSERT INTO Packages VALUES
    -> (1, 'Golden Triangle', 'Delhi-Agra-Jaipur tour', 25000, 7),
    -> (2, 'Kerala Bliss', 'Backwaters and houseboat stay', 32000, 5),
    -> (3, 'Ladakh Adventure', 'High altitude trekking', 45000, 10),
    -> (4, 'Goa Beach Holiday', 'Relaxing beach vacation', 28000, 6),
    -> (5, 'Spiritual India', 'Varanasi-Rishikesh tour', 22000, 5),
    -> (6, 'South India Heritage', 'Mysore-Hampi temples', 35000, 8),
    -> (7, 'Andaman Escape', 'Island hopping and snorkeling', 42000, 7),
    -> (8, 'Rajasthan Royalty', 'Desert forts and palaces', 38000, 9),
    -> (9, 'Himalayan Trek', 'Manali-Leh expedition', 52000, 12),
    -> (10, 'Wildlife Safari', 'Ranthambore and Bandipur', 29000, 6);
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

**Packages_Dates table:**

16. 

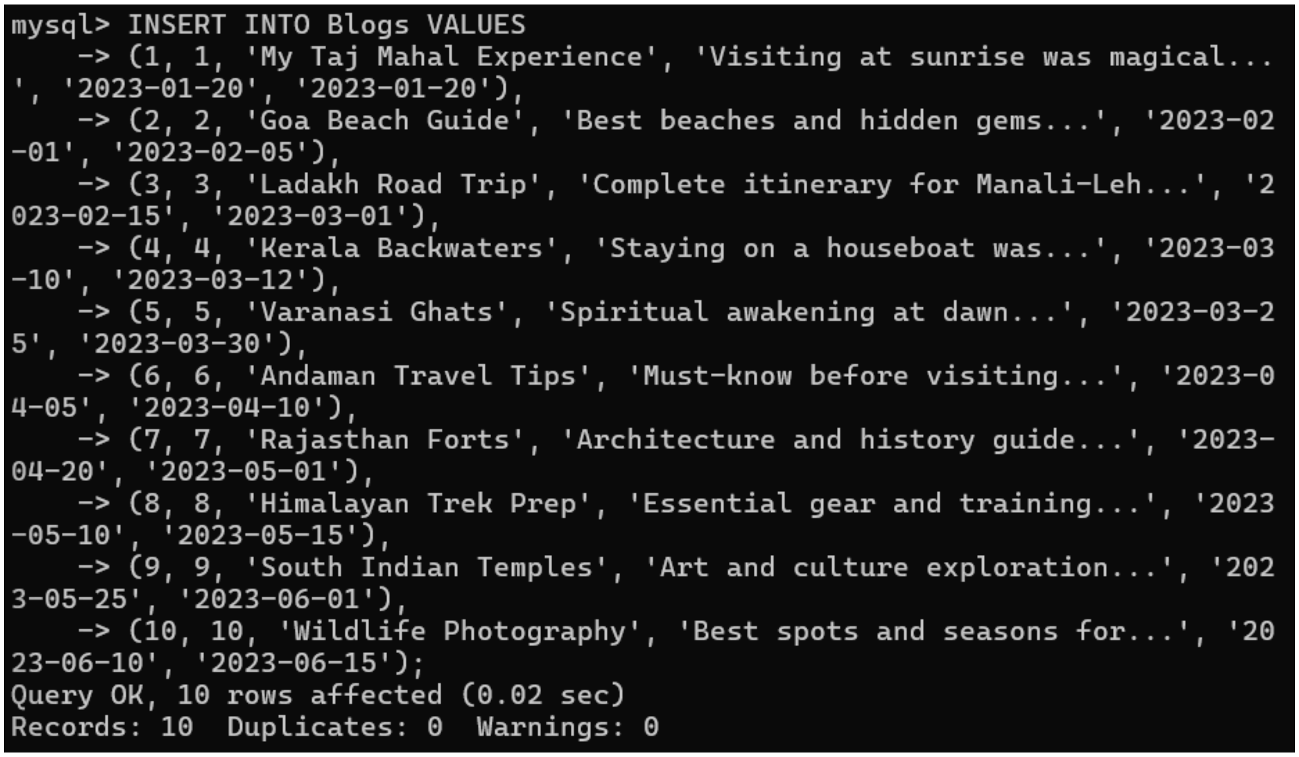
```
mysql> INSERT INTO Package_Dates VALUES
    -> (1, '2023-11-15'), (1, '2023-12-01'), (1, '2024-01-10'),
    -> (2, '2023-10-20'), (2, '2023-11-05'), (2, '2023-12-15'),
    -> (3, '2023-09-01'), (3, '2024-05-15'), (3, '2024-06-20'),
    -> (4, '2023-12-10'), (4, '2024-01-05'), (4, '2024-02-20'),
    -> (5, '2023-11-01'), (5, '2023-11-20'), (5, '2023-12-10'),
    -> (6, '2023-10-15'), (6, '2023-11-25'), (6, '2024-01-05'),
    -> (7, '2023-12-05'), (7, '2024-01-15'), (7, '2024-02-25'),
    -> (8, '2023-10-10'), (8, '2023-11-20'), (8, '2023-12-30'),
    -> (9, '2024-06-01'), (9, '2024-07-15'), (9, '2024-08-20'),
    -> (10, '2023-11-05'), (10, '2023-12-15'), (10, '2024-01-25');
Query OK, 30 rows affected (0.01 sec)
Records: 30  Duplicates: 0  Warnings: 0
```

**Package_Destinations table:**

**17.** 

```
mysql> INSERT INTO Package_Destinations VALUES
    -> (1,1), (1,2),
    -> (2,4),
    -> (3,5),
    -> (4,3),
    -> (5,6), (5,9),
    -> (6,7),
    -> (7,8),
    -> (8,2),
    -> (9,5),
    -> (10,2), (10,7);
Query OK, 13 rows affected (0.02 sec)
Records: 13  Duplicates: 0  Warnings: 0
```

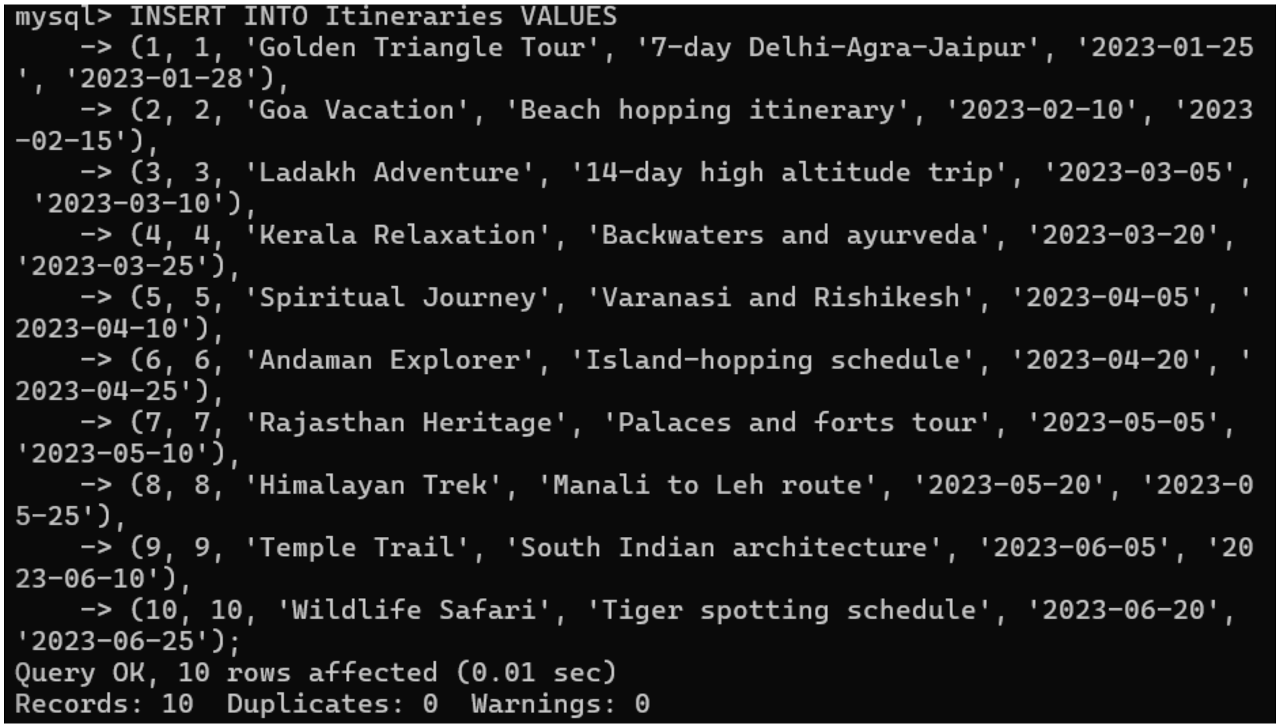
**Blogs table:**



```
mysql> INSERT INTO Blogs VALUES
    -> (1, 1, 'My Taj Mahal Experience', 'Visiting at sunrise was magical...
', '2023-01-20', '2023-01-20'),
    -> (2, 2, 'Goa Beach Guide', 'Best beaches and hidden gems...', '2023-02
-01', '2023-02-05'),
    -> (3, 3, 'Ladakh Road Trip', 'Complete itinerary for Manali-Leh...', '2
023-02-15', '2023-03-01'),
    -> (4, 4, 'Kerala Backwaters', 'Staying on a houseboat was...', '2023-03
-10', '2023-03-12'),
    -> (5, 5, 'Varanasi Ghats', 'Spiritual awakening at dawn...', '2023-03-2
5', '2023-03-30'),
    -> (6, 6, 'Andaman Travel Tips', 'Must-know before visiting...', '2023-0
4-05', '2023-04-10'),
    -> (7, 7, 'Rajasthan Forts', 'Architecture and history guide...', '2023-
04-20', '2023-05-01'),
    -> (8, 8, 'Himalayan Trek Prep', 'Essential gear and training...', '2023
-05-10', '2023-05-15'),
    -> (9, 9, 'South Indian Temples', 'Art and culture exploration...', '202
3-05-25', '2023-06-01'),
    -> (10, 10, 'Wildlife Photography', 'Best spots and seasons for...', '20
23-06-10', '2023-06-15');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

18.

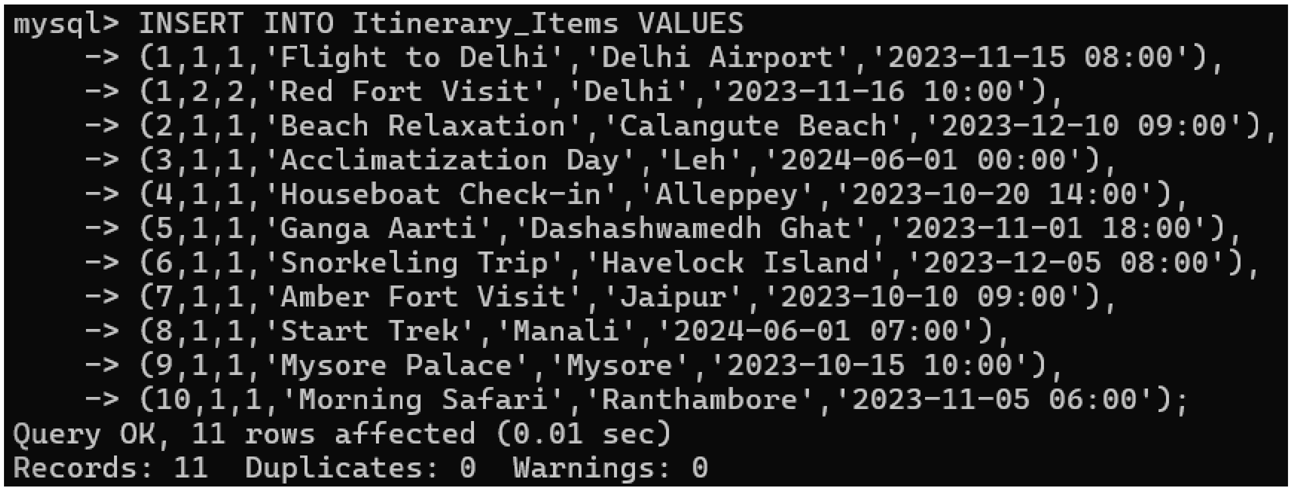
**Itineraries table:**



```
mysql> INSERT INTO Itineraries VALUES
    -> (1, 1, 'Golden Triangle Tour', '7-day Delhi-Agra-Jaipur', '2023-01-25
', '2023-01-28'),
    -> (2, 2, 'Goa Vacation', 'Beach hopping itinerary', '2023-02-10', '2023
-02-15'),
    -> (3, 3, 'Ladakh Adventure', '14-day high altitude trip', '2023-03-05',
 '2023-03-10'),
    -> (4, 4, 'Kerala Relaxation', 'Backwaters and ayurveda', '2023-03-20',
'2023-03-25'),
    -> (5, 5, 'Spiritual Journey', 'Varanasi and Rishikesh', '2023-04-05', '
2023-04-10'),
    -> (6, 6, 'Andaman Explorer', 'Island-hopping schedule', '2023-04-20', '
2023-04-25'),
    -> (7, 7, 'Rajasthan Heritage', 'Palaces and forts tour', '2023-05-05',
'2023-05-10'),
    -> (8, 8, 'Himalayan Trek', 'Manali to Leh route', '2023-05-20', '2023-0
5-25'),
    -> (9, 9, 'Temple Trail', 'South Indian architecture', '2023-06-05', '20
23-06-10'),
    -> (10, 10, 'Wildlife Safari', 'Tiger spotting schedule', '2023-06-20',
'2023-06-25');
Query OK, 10 rows affected (0.01 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

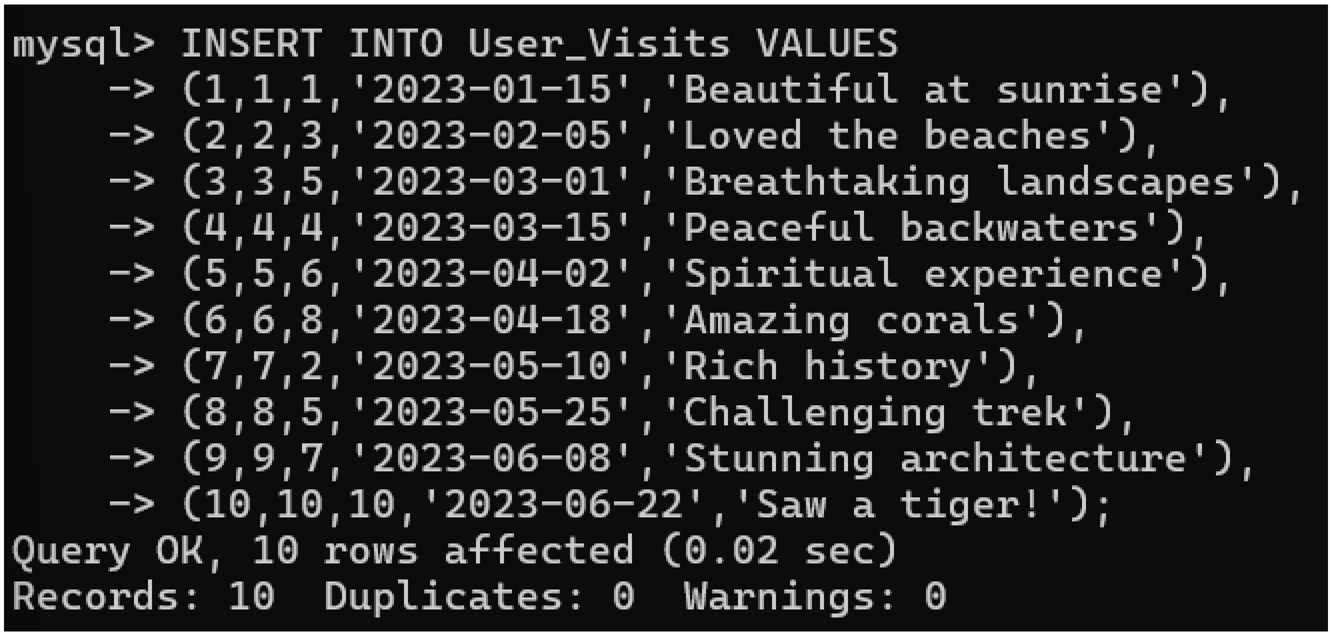
19.

**Itinerary_Items table:**

20. 

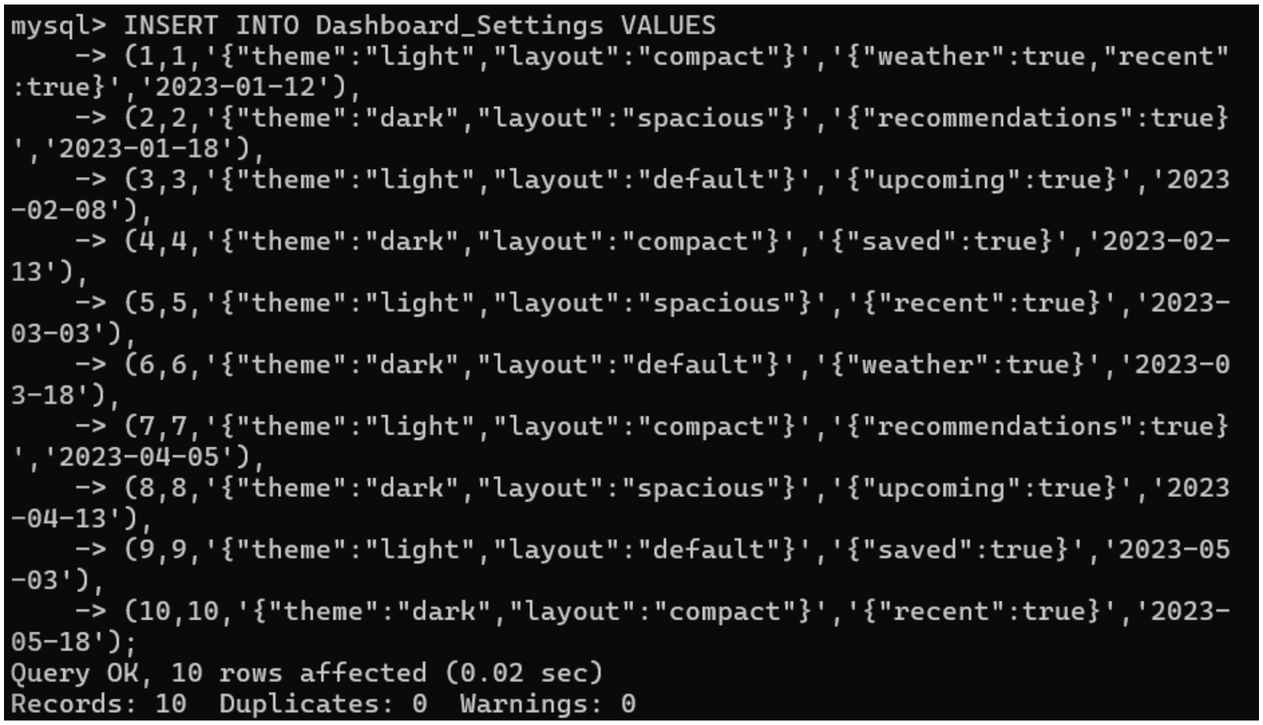
```
mysql> INSERT INTO Itinerary_Items VALUES
    -> (1,1,1,'Flight to Delhi','Delhi Airport','2023-11-15 08:00'),
    -> (1,2,2,'Red Fort Visit','Delhi','2023-11-16 10:00'),
    -> (2,1,1,'Beach Relaxation','Calangute Beach','2023-12-10 09:00'),
    -> (3,1,1,'Acclimatization Day','Leh','2024-06-01 00:00'),
    -> (4,1,1,'Houseboat Check-in','Alleppey','2023-10-20 14:00'),
    -> (5,1,1,'Ganga Aarti','Dashashwamedh Ghat','2023-11-01 18:00'),
    -> (6,1,1,'Snorkeling Trip','Havelock Island','2023-12-05 08:00'),
    -> (7,1,1,'Amber Fort Visit','Jaipur','2023-10-10 09:00'),
    -> (8,1,1,'Start Trek','Manali','2024-06-01 07:00'),
    -> (9,1,1,'Mysore Palace','Mysore','2023-10-15 10:00'),
    -> (10,1,1,'Morning Safari','Ranthambore','2023-11-05 06:00');
Query OK, 11 rows affected (0.01 sec)
Records: 11  Duplicates: 0  Warnings: 0
```

**User_Visits table:**

21. 

```
mysql> INSERT INTO User_Visits VALUES
    -> (1,1,1,'2023-01-15','Beautiful at sunrise'),
    -> (2,2,3,'2023-02-05','Loved the beaches'),
    -> (3,3,5,'2023-03-01','Breathtaking landscapes'),
    -> (4,4,4,'2023-03-15','Peaceful backwaters'),
    -> (5,5,6,'2023-04-02','Spiritual experience'),
    -> (6,6,8,'2023-04-18','Amazing corals'),
    -> (7,7,2,'2023-05-10','Rich history'),
    -> (8,8,5,'2023-05-25','Challenging trek'),
    -> (9,9,7,'2023-06-08','Stunning architecture'),
    -> (10,10,10,'2023-06-22','Saw a tiger!');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

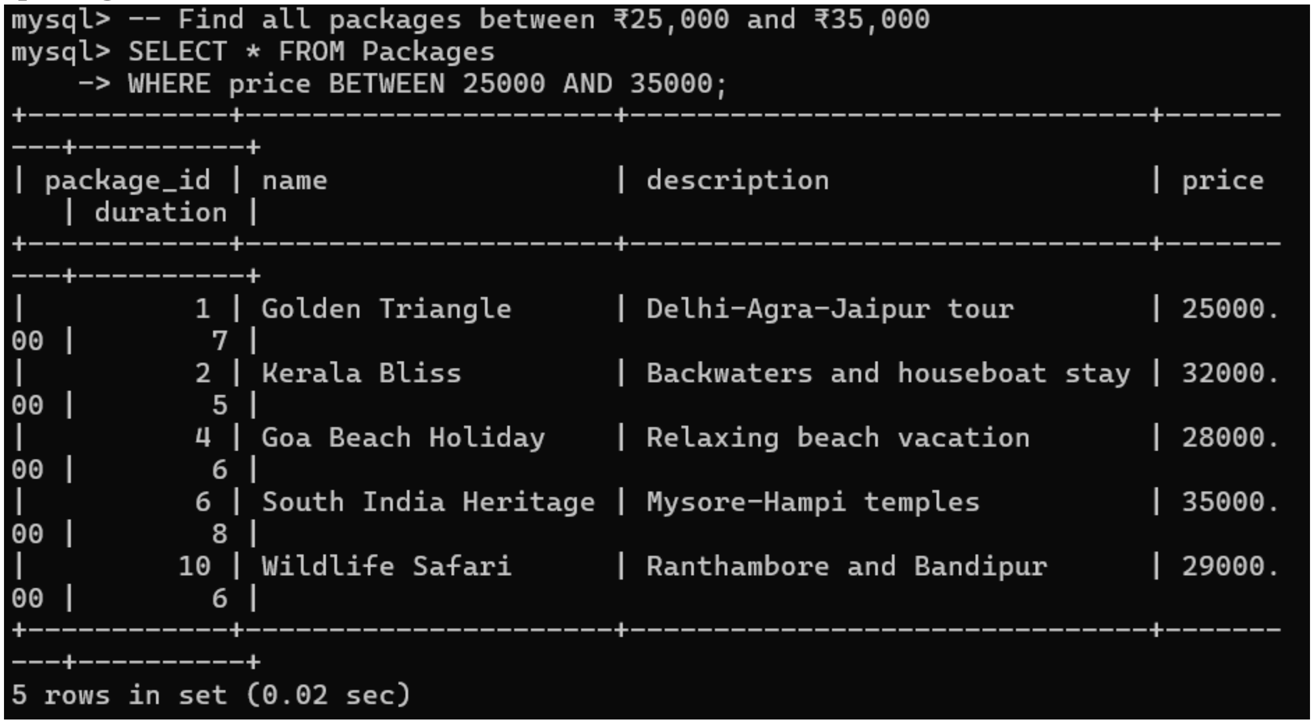
**Dashboard_Settings:**



```
mysql> INSERT INTO Dashboard_Settings VALUES
    -> (1,1,'{"theme":"light","layout":"compact"}','{"weather":true,"recent"
:true}','2023-01-12'),
    -> (2,2,'{"theme":"dark","layout":"spacious"}','{"recommendations":true}
','2023-01-18'),
    -> (3,3,'{"theme":"light","layout":"default"}','{"upcoming":true}','2023
-02-08'),
    -> (4,4,'{"theme":"dark","layout":"compact"}','{"saved":true}','2023-02-
13'),
    -> (5,5,'{"theme":"light","layout":"spacious"}','{"recent":true}','2023-
03-03'),
    -> (6,6,'{"theme":"dark","layout":"default"}','{"weather":true}','2023-0
3-18'),
    -> (7,7,'{"theme":"light","layout":"compact"}','{"recommendations":true}
','2023-04-05'),
    -> (8,8,'{"theme":"dark","layout":"spacious"}','{"upcoming":true}','2023
-04-13'),
    -> (9,9,'{"theme":"light","layout":"default"}','{"saved":true}','2023-05
-03'),
    -> (10,10,'{"theme":"dark","layout":"compact"}','{"recent":true}','2023-
05-18');
Query OK, 10 rows affected (0.02 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

22.

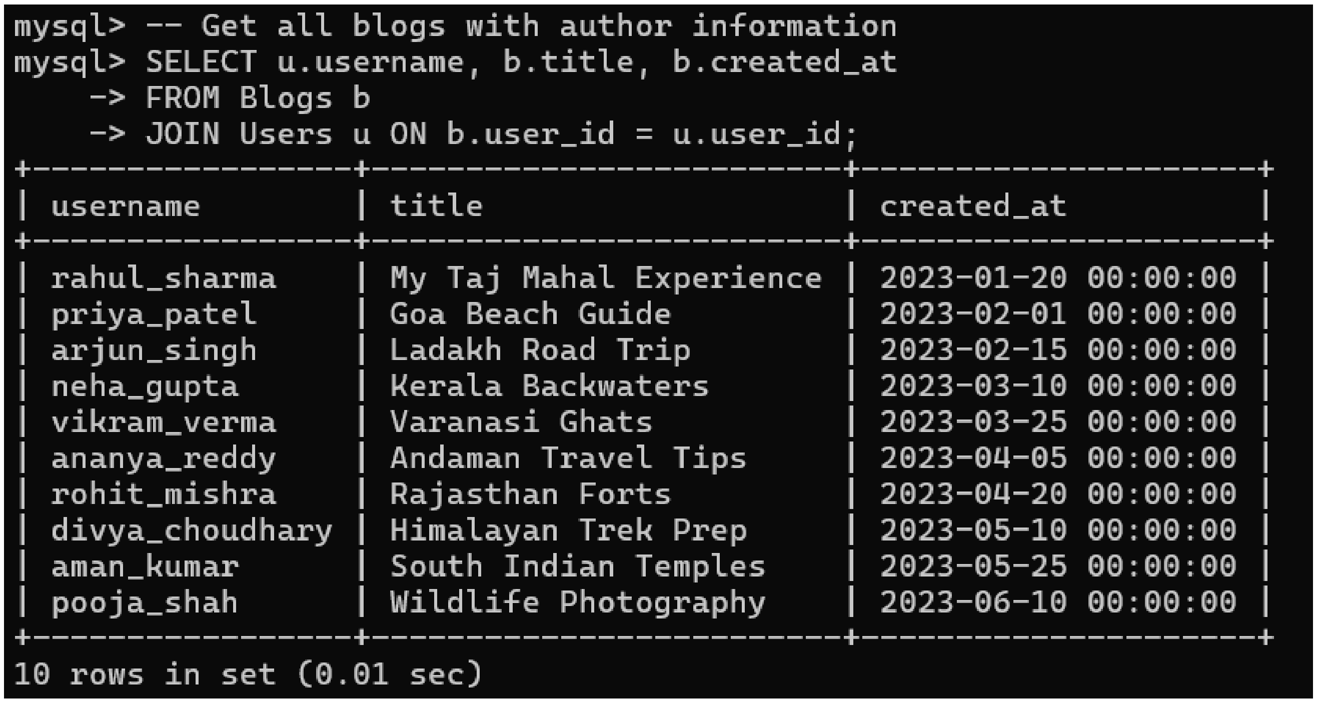
**Find all packages between 25,000 and 35,000:**



```
mysql> -- Find all packages between ₹25,000 and ₹35,000
mysql> SELECT * FROM Packages
    -> WHERE price BETWEEN 25000 AND 35000;
+------------+----------------------+--------------------------------+-------
---+----------+
| package_id | name                 | description                    | price
   | duration |
+------------+----------------------+--------------------------------+-------
---+----------+
|          1 | Golden Triangle      | Delhi-Agra-Jaipur tour         | 25000.
00 |        7 |
|          2 | Kerala Bliss         | Backwaters and houseboat stay  | 32000.
00 |        5 |
|          4 | Goa Beach Holiday    | Relaxing beach vacation        | 28000.
00 |        6 |
|          6 | South India Heritage | Mysore-Hampi temples           | 35000.
00 |        8 |
|         10 | Wildlife Safari      | Ranthambore and Bandipur       | 29000.
00 |        6 |
+------------+----------------------+--------------------------------+-------
---+----------+
5 rows in set (0.02 sec)
```

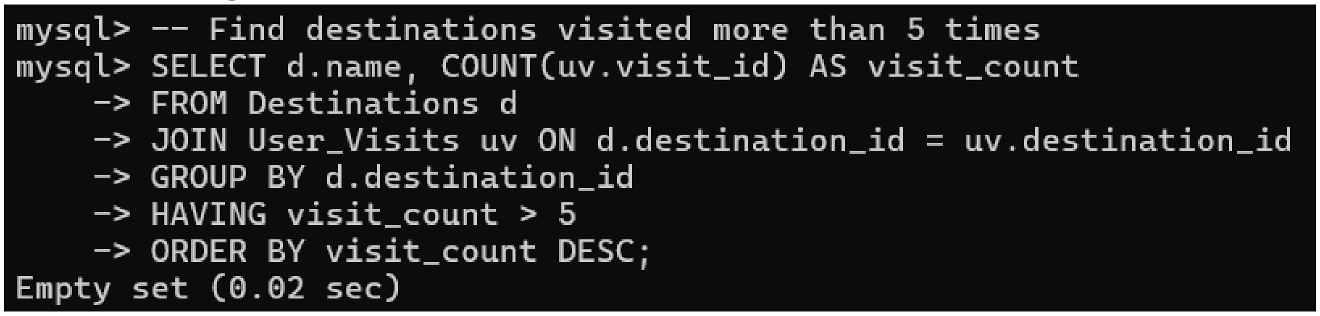
23.

**Get blogs with author informations:**

24. 

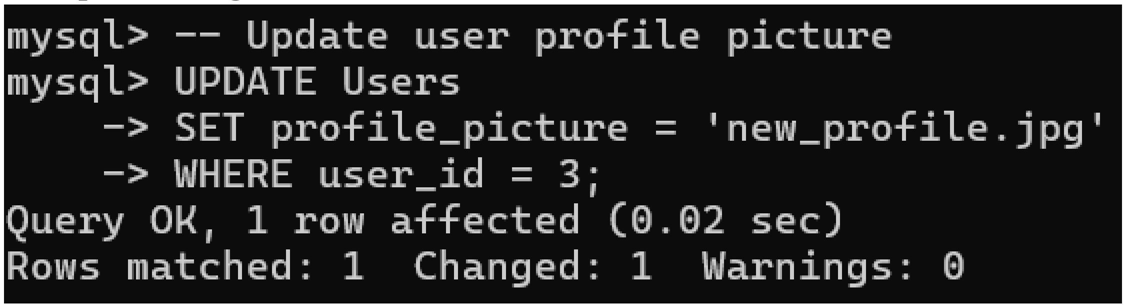
```
mysql> -- Get all blogs with author information
mysql> SELECT u.username, b.title, b.created_at
    -> FROM Blogs b
    -> JOIN Users u ON b.user_id = u.user_id;
+------------------+--------------------------+---------------------+
| username         | title                    | created_at          |
+------------------+--------------------------+---------------------+
| rahul_sharma     | My Taj Mahal Experience  | 2023-01-20 00:00:00 |
| priya_patel      | Goa Beach Guide          | 2023-02-01 00:00:00 |
| arjun_singh      | Ladakh Road Trip         | 2023-02-15 00:00:00 |
| neha_gupta       | Kerala Backwaters        | 2023-03-10 00:00:00 |
| vikram_verma     | Varanasi Ghats           | 2023-03-25 00:00:00 |
| ananya_reddy     | Andaman Travel Tips      | 2023-04-05 00:00:00 |
| rohit_mishra     | Rajasthan Forts          | 2023-04-20 00:00:00 |
| divya_choudhary  | Himalayan Trek Prep      | 2023-05-10 00:00:00 |
| aman_kumar       | South Indian Temples     | 2023-05-25 00:00:00 |
| pooja_shah       | Wildlife Photography     | 2023-06-10 00:00:00 |
+------------------+--------------------------+---------------------+
10 rows in set (0.01 sec)
```

**Find destinations having more than 5 visits:**

25. 

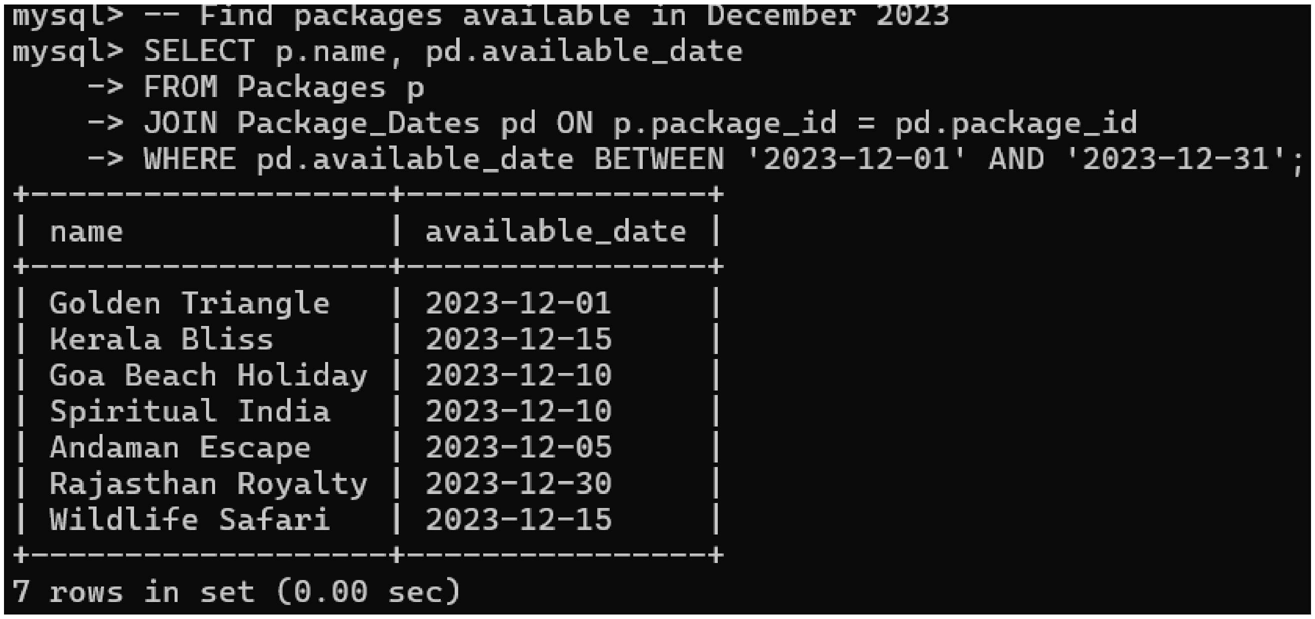
```
mysql> -- Find destinations visited more than 5 times
mysql> SELECT d.name, COUNT(uv.visit_id) AS visit_count
    -> FROM Destinations d
    -> JOIN User_Visits uv ON d.destination_id = uv.destination_id
    -> GROUP BY d.destination_id
    -> HAVING visit_count > 5
    -> ORDER BY visit_count DESC;
Empty set (0.02 sec)
```

**Update user profile image:**

26. 

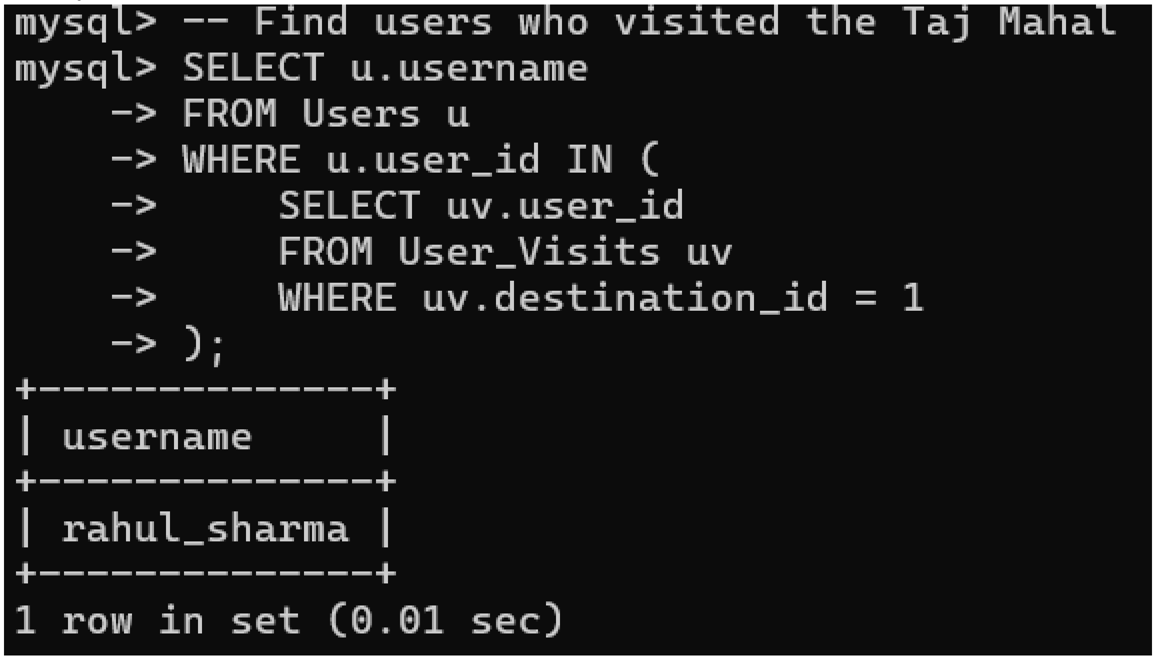
```
mysql> -- Update user profile picture
mysql> UPDATE Users
    -> SET profile_picture = 'new_profile.jpg'
    -> WHERE user_id = 3;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

**Find packages available in December:**

27. 

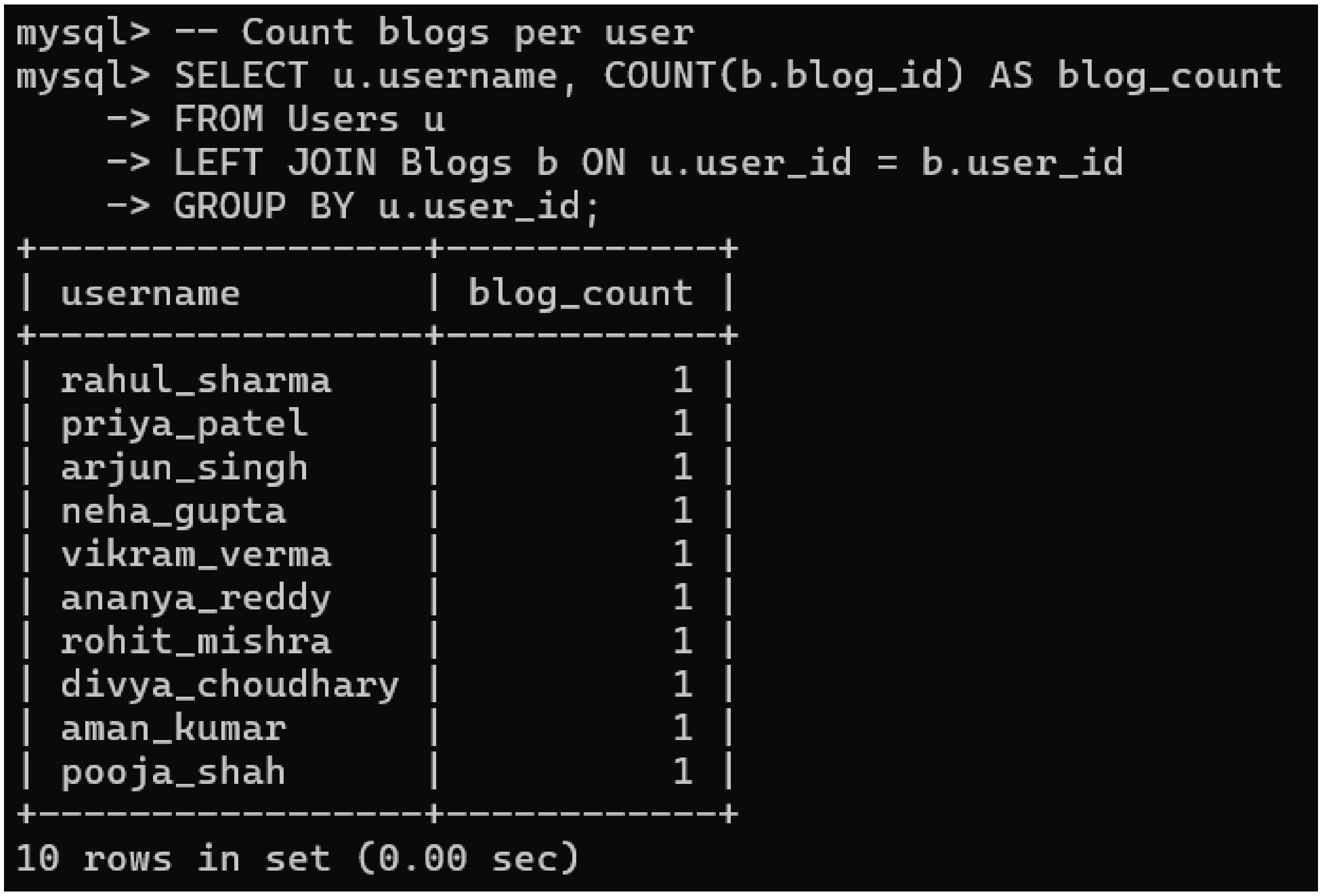
```
mysql> -- Find packages available in December 2023
mysql> SELECT p.name, pd.available_date
    -> FROM Packages p
    -> JOIN Package_Dates pd ON p.package_id = pd.package_id
    -> WHERE pd.available_date BETWEEN '2023-12-01' AND '2023-12-31';
+-------------------+----------------+
| name              | available_date |
+-------------------+----------------+
| Golden Triangle   | 2023-12-01     |
| Kerala Bliss      | 2023-12-15     |
| Goa Beach Holiday | 2023-12-10     |
| Spiritual India   | 2023-12-10     |
| Andaman Escape    | 2023-12-05     |
| Rajasthan Royalty | 2023-12-30     |
| Wildlife Safari   | 2023-12-15     |
+-------------------+----------------+
7 rows in set (0.00 sec)
```

**Nested Query:**

28. 

```
mysql> -- Find users who visited the Taj Mahal
mysql> SELECT u.username
    -> FROM Users u
    -> WHERE u.user_id IN (
    ->     SELECT uv.user_id
    ->     FROM User_Visits uv
    ->     WHERE uv.destination_id = 1
    -> );
+--------------+
| username     |
+--------------+
| rahul_sharma |
+--------------+
1 row in set (0.01 sec)
```

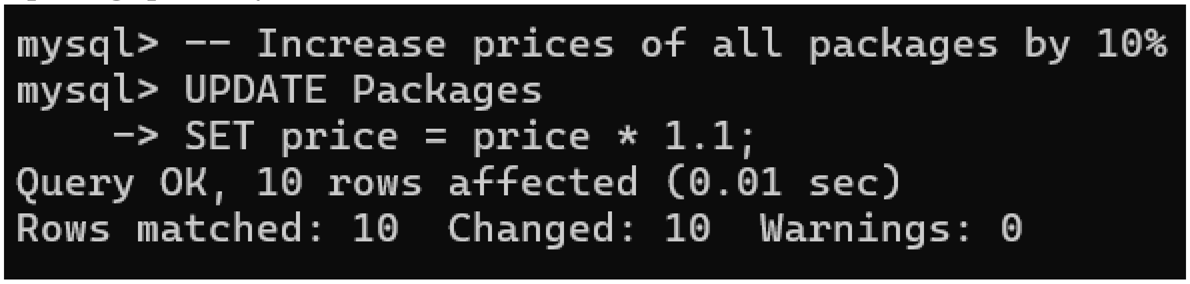
**No. of blogs written per user (Aggregate Function):**



```
mysql> -- Count blogs per user
mysql> SELECT u.username, COUNT(b.blog_id) AS blog_count
    -> FROM Users u
    -> LEFT JOIN Blogs b ON u.user_id = b.user_id
    -> GROUP BY u.user_id;
+-----------------+------------+
| username        | blog_count |
+-----------------+------------+
| rahul_sharma    |          1 |
| priya_patel     |          1 |
| arjun_singh     |          1 |
| neha_gupta      |          1 |
| vikram_verma    |          1 |
| ananya_reddy    |          1 |
| rohit_mishra    |          1 |
| divya_choudhary |          1 |
| aman_kumar      |          1 |
| pooja_shah      |          1 |
+-----------------+------------+
10 rows in set (0.00 sec)
```

29.

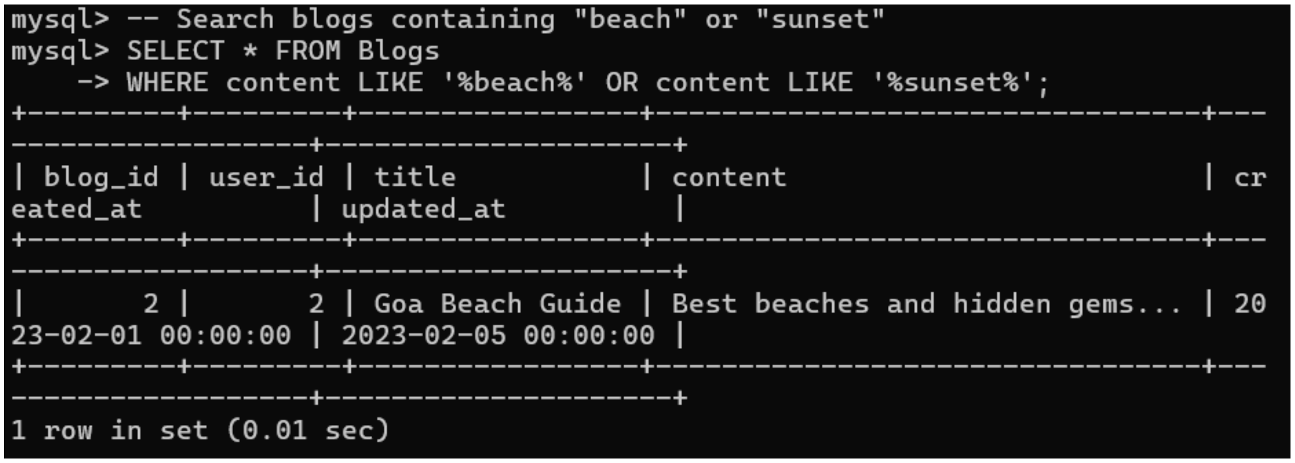
**Increase package prices by 10%:**



```
mysql> -- Increase prices of all packages by 10%
mysql> UPDATE Packages
    -> SET price = price * 1.1;
Query OK, 10 rows affected (0.01 sec)
Rows matched: 10  Changed: 10  Warnings: 0
```

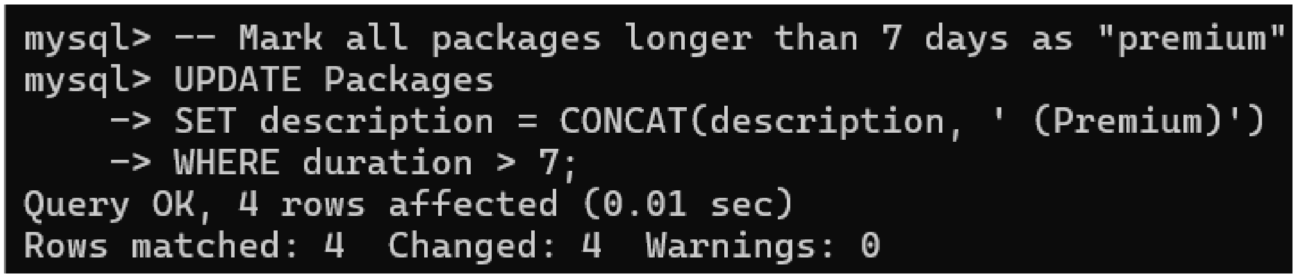
30.

**Using LIKE keyword:**

31. 

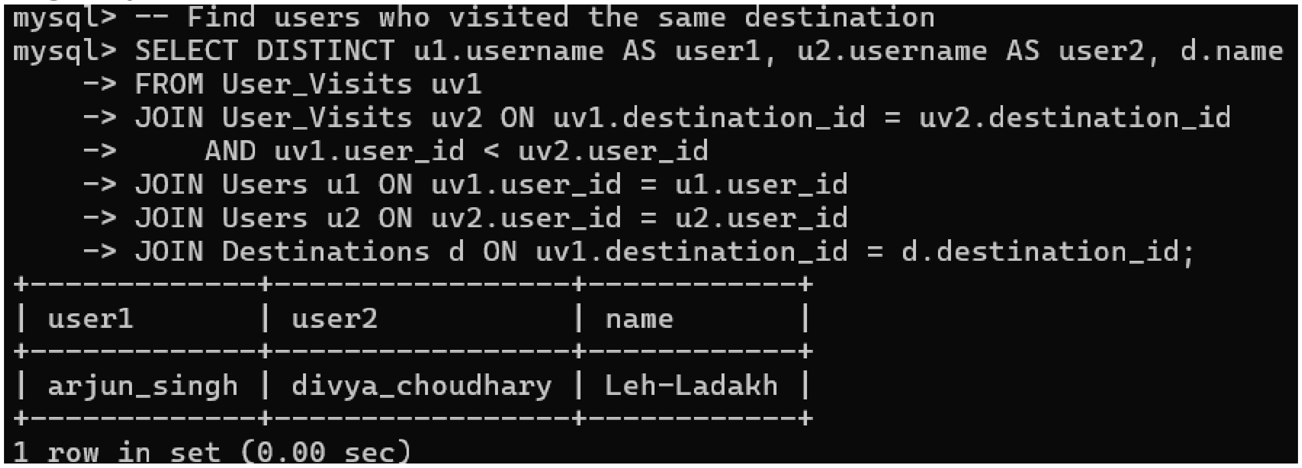
```
mysql> -- Search blogs containing "beach" or "sunset"
mysql> SELECT * FROM Blogs
    -> WHERE content LIKE '%beach%' OR content LIKE '%sunset%';
+---------+---------+-----------------+----------------------------------+---
------------------+---------------------+
| blog_id | user_id | title           | content                          | cr
eated_at          | updated_at          |
+---------+---------+-----------------+----------------------------------+---
------------------+---------------------+
|       2 |       2 | Goa Beach Guide | Best beaches and hidden gems...  | 20
23-02-01 00:00:00 | 2023-02-05 00:00:00 |
+---------+---------+-----------------+----------------------------------+---
------------------+---------------------+
1 row in set (0.01 sec)
```

**Update with condition:**

32. 

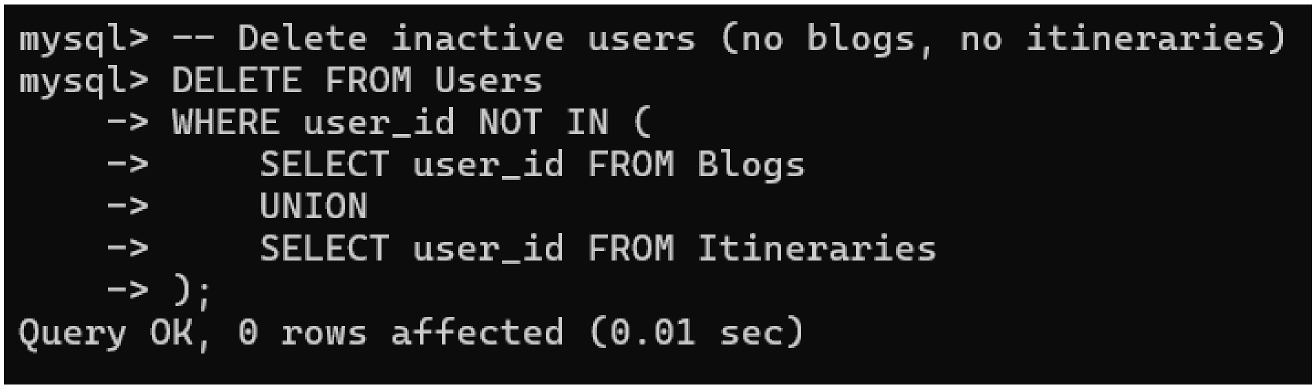
```
mysql> -- Mark all packages longer than 7 days as "premium"
mysql> UPDATE Packages
    -> SET description = CONCAT(description, ' (Premium)')
    -> WHERE duration > 7;
Query OK, 4 rows affected (0.01 sec)
Rows matched: 4  Changed: 4  Warnings: 0
```

**Performing self-join:**

33. 

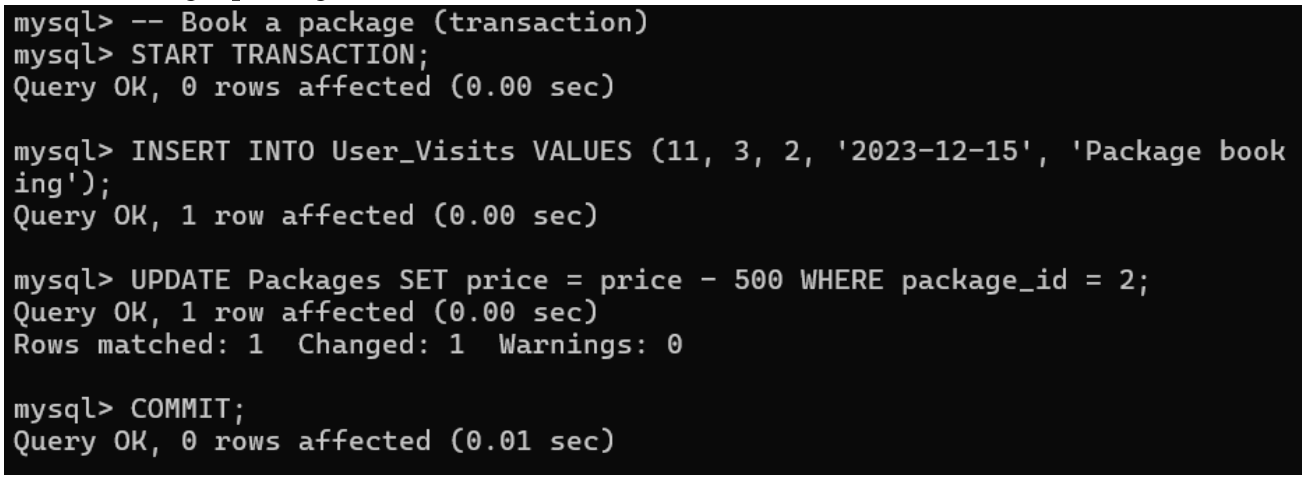
```
mysql> -- Find users who visited the same destination
mysql> SELECT DISTINCT u1.username AS user1, u2.username AS user2, d.name
    -> FROM User_Visits uv1
    -> JOIN User_Visits uv2 ON uv1.destination_id = uv2.destination_id
    ->     AND uv1.user_id < uv2.user_id
    -> JOIN Users u1 ON uv1.user_id = u1.user_id
    -> JOIN Users u2 ON uv2.user_id = u2.user_id
    -> JOIN Destinations d ON uv1.destination_id = d.destination_id;
+-------------+-----------------+------------+
| user1       | user2           | name       |
+-------------+-----------------+------------+
| arjun_singh | divya_choudhary | Leh-Ladakh |
+-------------+-----------------+------------+
1 row in set (0.00 sec)
```

**Checks and Deletes in-active users:**

34. 

```
mysql> -- Delete inactive users (no blogs, no itineraries)
mysql> DELETE FROM Users
    -> WHERE user_id NOT IN (
    ->     SELECT user_id FROM Blogs
    ->     UNION
    ->     SELECT user_id FROM Itineraries
    -> );
Query OK, 0 rows affected (0.01 sec)
```

**Transaction (Booking a package):**

35. 

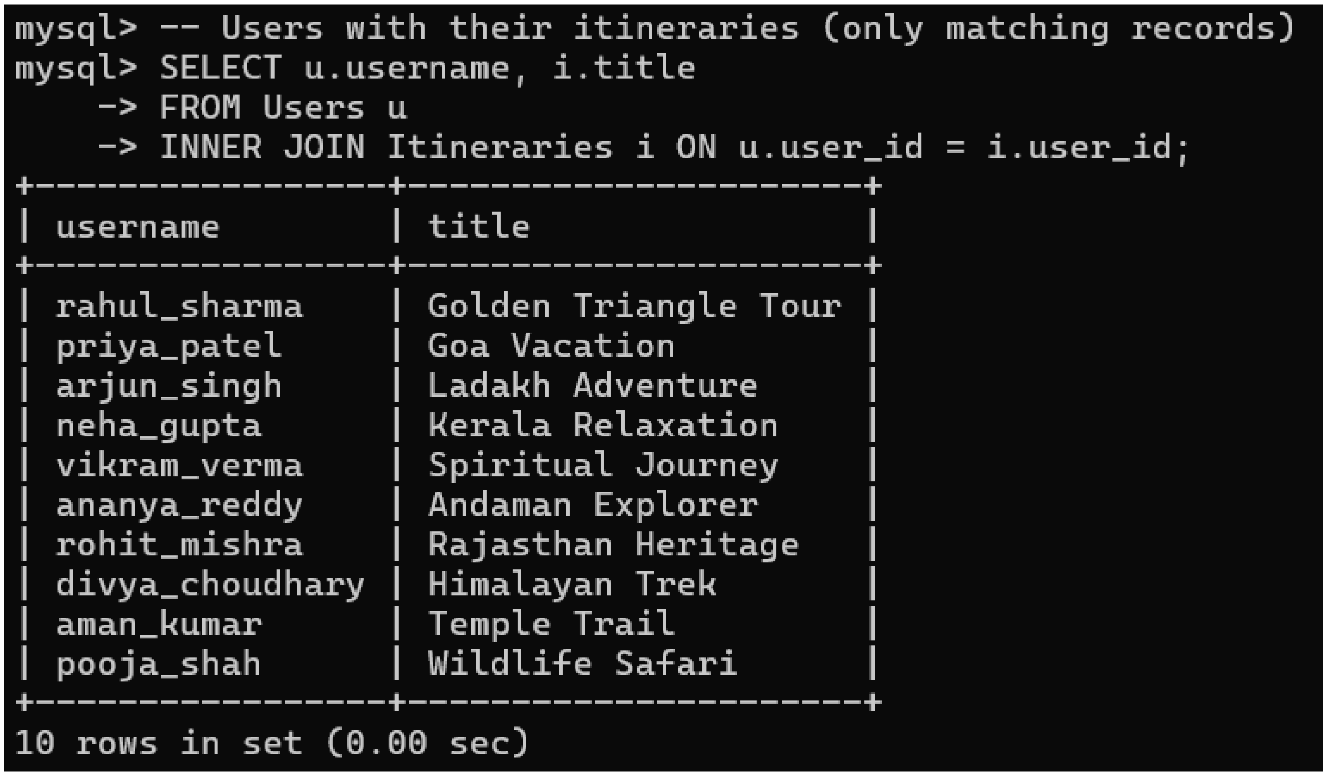
```
mysql> -- Book a package (transaction)
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO User_Visits VALUES (11, 3, 2, '2023-12-15', 'Package book
ing');
Query OK, 1 row affected (0.00 sec)

mysql> UPDATE Packages SET price = price - 500 WHERE package_id = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)
```

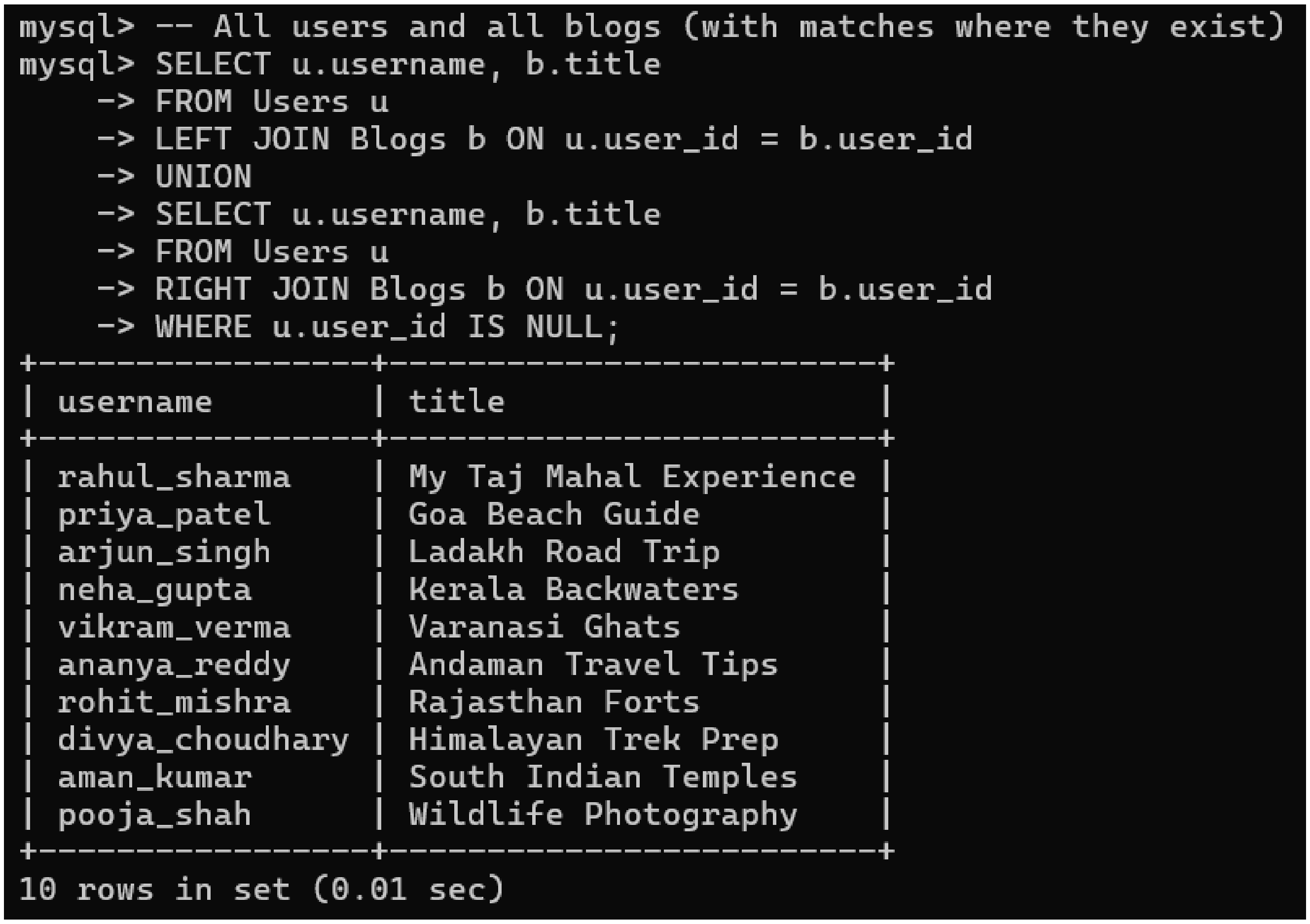
**Find users who have itineraries:**



```
mysql> -- Users with their itineraries (only matching records)
mysql> SELECT u.username, i.title
    -> FROM Users u
    -> INNER JOIN Itineraries i ON u.user_id = i.user_id;
+-----------------+----------------------+
| username        | title                |
+-----------------+----------------------+
| rahul_sharma    | Golden Triangle Tour |
| priya_patel     | Goa Vacation         |
| arjun_singh     | Ladakh Adventure     |
| neha_gupta      | Kerala Relaxation    |
| vikram_verma    | Spiritual Journey    |
| ananya_reddy    | Andaman Explorer     |
| rohit_mishra    | Rajasthan Heritage   |
| divya_choudhary | Himalayan Trek       |
| aman_kumar      | Temple Trail         |
| pooja_shah      | Wildlife Safari      |
+-----------------+----------------------+
10 rows in set (0.00 sec)
```

36.

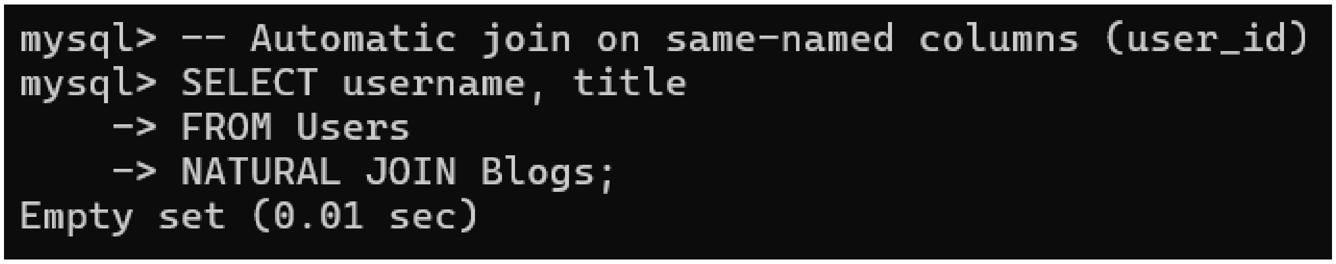
**Full Outer Join:**



```
mysql> -- All users and all blogs (with matches where they exist)
mysql> SELECT u.username, b.title
    -> FROM Users u
    -> LEFT JOIN Blogs b ON u.user_id = b.user_id
    -> UNION
    -> SELECT u.username, b.title
    -> FROM Users u
    -> RIGHT JOIN Blogs b ON u.user_id = b.user_id
    -> WHERE u.user_id IS NULL;
+-----------------+------------------------+
| username        | title                  |
+-----------------+------------------------+
| rahul_sharma    | My Taj Mahal Experience |
| priya_patel     | Goa Beach Guide        |
| arjun_singh     | Ladakh Road Trip       |
| neha_gupta      | Kerala Backwaters      |
| vikram_verma    | Varanasi Ghats         |
| ananya_reddy    | Andaman Travel Tips    |
| rohit_mishra    | Rajasthan Forts        |
| divya_choudhary | Himalayan Trek Prep    |
| aman_kumar      | South Indian Temples   |
| pooja_shah      | Wildlife Photography   |
+-----------------+------------------------+
10 rows in set (0.01 sec)
```

37.

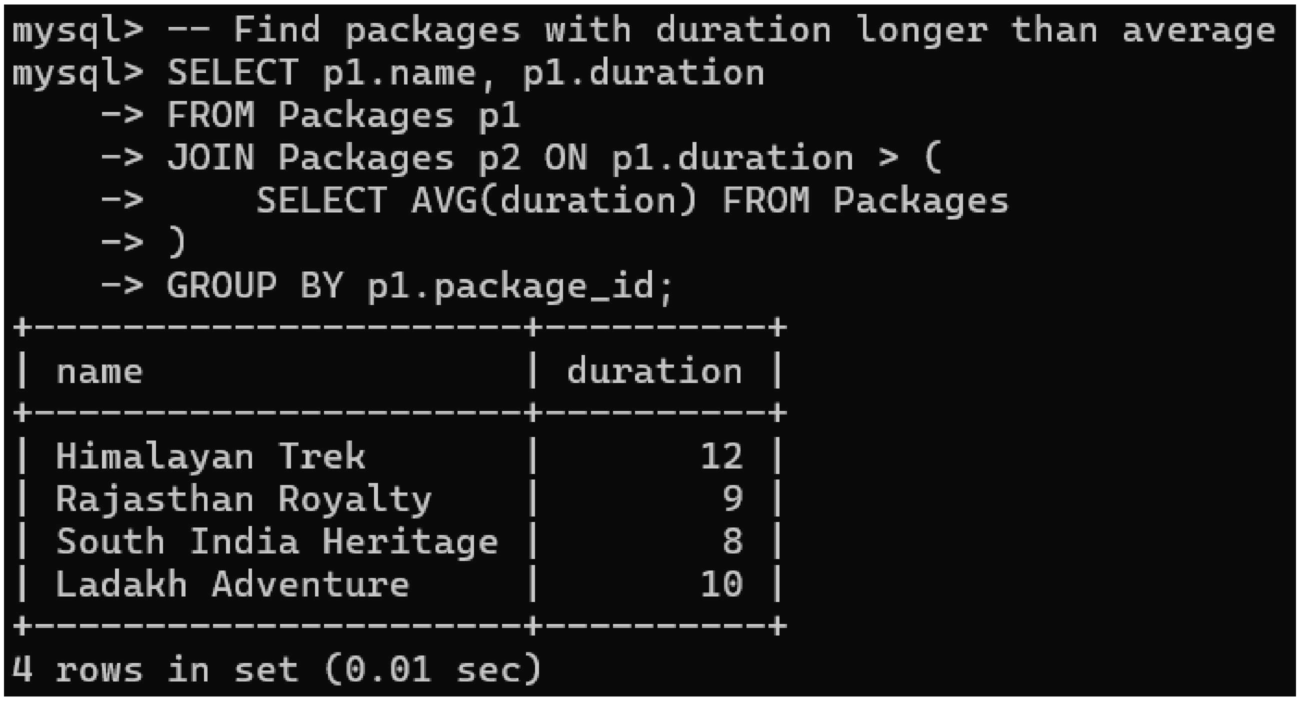
**Natural Join:**



```
mysql> -- Automatic join on same-named columns (user_id)
mysql> SELECT username, title
    -> FROM Users
    -> NATURAL JOIN Blogs;
Empty set (0.01 sec)
```

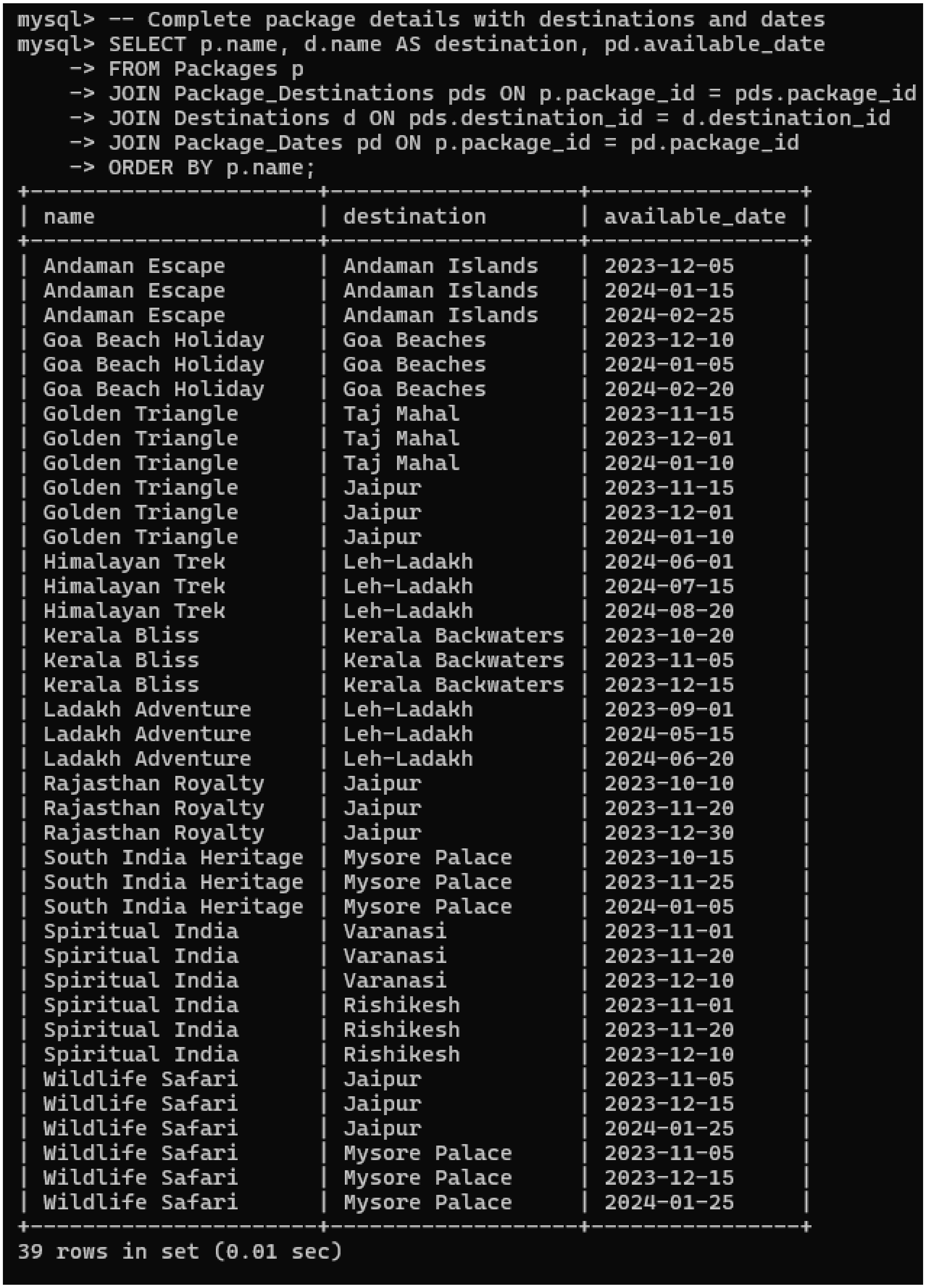
38.

**Theta-Join:**

39. 

```
mysql> -- Find packages with duration longer than average
mysql> SELECT p1.name, p1.duration
    -> FROM Packages p1
    -> JOIN Packages p2 ON p1.duration > (
    ->     SELECT AVG(duration) FROM Packages
    -> )
    -> GROUP BY p1.package_id;
+---------------------+----------+
| name                | duration |
+---------------------+----------+
| Himalayan Trek      |       12 |
| Rajasthan Royalty   |        9 |
| South India Heritage |       8 |
| Ladakh Adventure    |       10 |
+---------------------+----------+
4 rows in set (0.01 sec)
```

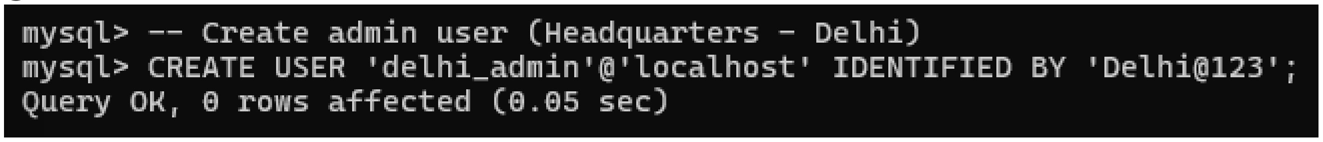
**Multiple Joins (Complete Package detail with dates):**



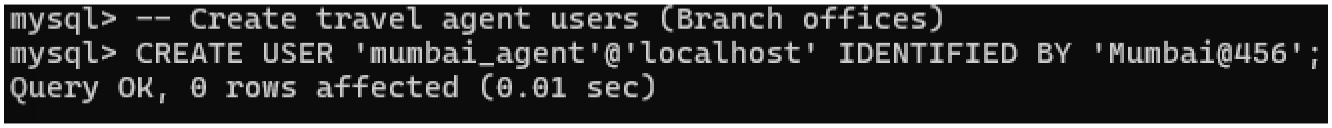
```
mysql> -- Complete package details with destinations and dates
mysql> SELECT p.name, d.name AS destination, pd.available_date
    -> FROM Packages p
    -> JOIN Package_Destinations pds ON p.package_id = pds.package_id
    -> JOIN Destinations d ON pds.destination_id = d.destination_id
    -> JOIN Package_Dates pd ON p.package_id = pd.package_id
    -> ORDER BY p.name;
+-----------------------+--------------------+----------------+
| name                  | destination        | available_date |
+-----------------------+--------------------+----------------+
| Andaman Escape        | Andaman Islands    | 2023-12-05     |
| Andaman Escape        | Andaman Islands    | 2024-01-15     |
| Andaman Escape        | Andaman Islands    | 2024-02-25     |
| Goa Beach Holiday     | Goa Beaches        | 2023-12-10     |
| Goa Beach Holiday     | Goa Beaches        | 2024-01-05     |
| Goa Beach Holiday     | Goa Beaches        | 2024-02-20     |
| Golden Triangle       | Taj Mahal          | 2023-11-15     |
| Golden Triangle       | Taj Mahal          | 2023-12-01     |
| Golden Triangle       | Taj Mahal          | 2024-01-10     |
| Golden Triangle       | Jaipur             | 2023-11-15     |
| Golden Triangle       | Jaipur             | 2023-12-01     |
| Golden Triangle       | Jaipur             | 2024-01-10     |
| Himalayan Trek        | Leh-Ladakh         | 2024-06-01     |
| Himalayan Trek        | Leh-Ladakh         | 2024-07-15     |
| Himalayan Trek        | Leh-Ladakh         | 2024-08-20     |
| Kerala Bliss          | Kerala Backwaters  | 2023-10-20     |
| Kerala Bliss          | Kerala Backwaters  | 2023-11-05     |
| Kerala Bliss          | Kerala Backwaters  | 2023-12-15     |
| Ladakh Adventure      | Leh-Ladakh         | 2023-09-01     |
| Ladakh Adventure      | Leh-Ladakh         | 2024-05-15     |
| Ladakh Adventure      | Leh-Ladakh         | 2024-06-20     |
| Rajasthan Royalty     | Jaipur             | 2023-10-10     |
| Rajasthan Royalty     | Jaipur             | 2023-11-20     |
| Rajasthan Royalty     | Jaipur             | 2023-12-30     |
| South India Heritage  | Mysore Palace      | 2023-10-15     |
| South India Heritage  | Mysore Palace      | 2023-11-25     |
| South India Heritage  | Mysore Palace      | 2024-01-05     |
| Spiritual India       | Varanasi           | 2023-11-01     |
| Spiritual India       | Varanasi           | 2023-11-20     |
| Spiritual India       | Varanasi           | 2023-12-10     |
| Spiritual India       | Rishikesh          | 2023-11-01     |
| Spiritual India       | Rishikesh          | 2023-11-20     |
| Spiritual India       | Rishikesh          | 2023-12-10     |
| Wildlife Safari       | Jaipur             | 2023-11-05     |
| Wildlife Safari       | Jaipur             | 2023-12-15     |
| Wildlife Safari       | Jaipur             | 2024-01-25     |
| Wildlife Safari       | Mysore Palace      | 2023-11-05     |
| Wildlife Safari       | Mysore Palace      | 2023-12-15     |
| Wildlife Safari       | Mysore Palace      | 2024-01-25     |
+-----------------------+--------------------+----------------+
39 rows in set (0.01 sec)
```

**40.**

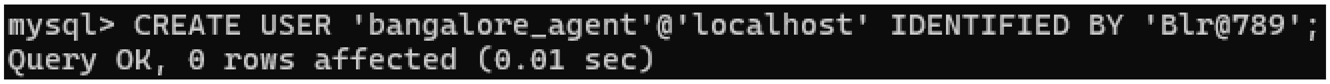
**Creating Users:**

41. 

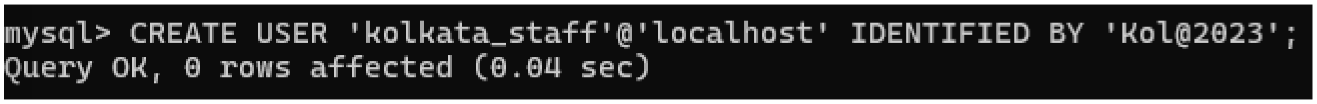
```
mysql> -- Create admin user (Headquarters - Delhi)
mysql> CREATE USER 'delhi_admin'@'localhost' IDENTIFIED BY 'Delhi@123';
Query OK, 0 rows affected (0.05 sec)
```

42. 

```
mysql> -- Create travel agent users (Branch offices)
mysql> CREATE USER 'mumbai_agent'@'localhost' IDENTIFIED BY 'Mumbai@456';
Query OK, 0 rows affected (0.01 sec)
```

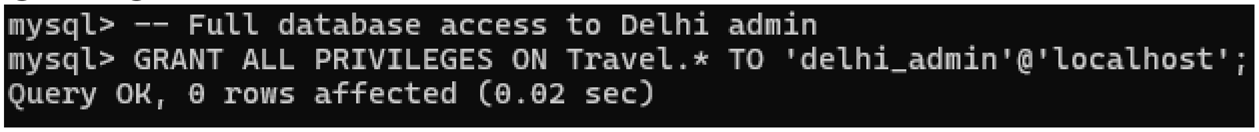
43. 

```
mysql> CREATE USER 'bangalore_agent'@'localhost' IDENTIFIED BY 'Blr@789';
Query OK, 0 rows affected (0.01 sec)
```

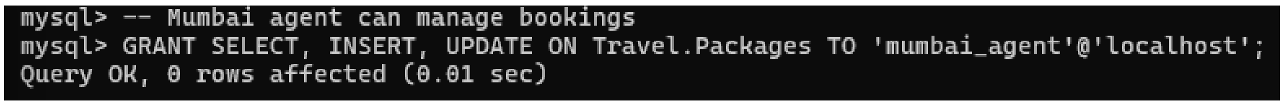
44. 

```
mysql> CREATE USER 'kolkata_staff'@'localhost' IDENTIFIED BY 'Kol@2023';
Query OK, 0 rows affected (0.04 sec)
```

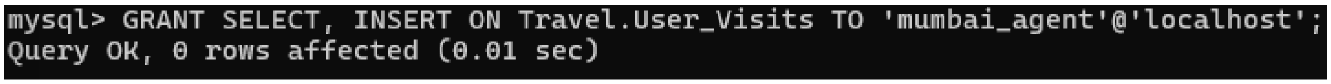
**Granting Privileges:**

45. 

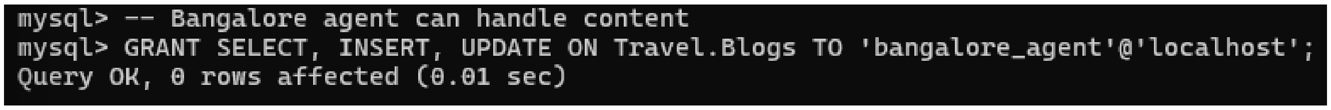
```
mysql> -- Full database access to Delhi admin
mysql> GRANT ALL PRIVILEGES ON Travel.* TO 'delhi_admin'@'localhost';
Query OK, 0 rows affected (0.02 sec)
```

46. 

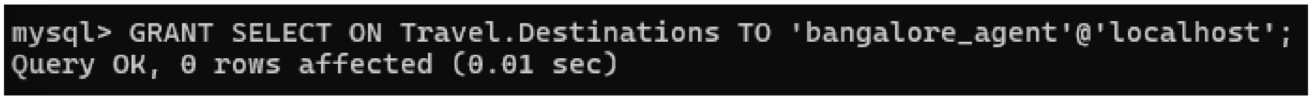
```
mysql> -- Mumbai agent can manage bookings
mysql> GRANT SELECT, INSERT, UPDATE ON Travel.Packages TO 'mumbai_agent'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

47. 

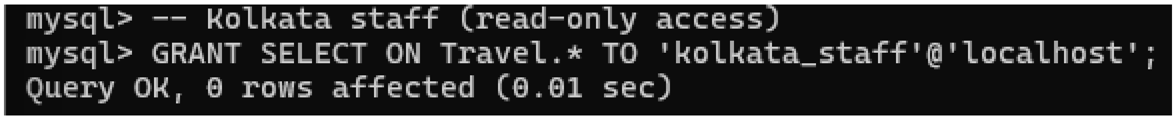
```
mysql> GRANT SELECT, INSERT ON Travel.User_Visits TO 'mumbai_agent'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

48. 

```
mysql> -- Bangalore agent can handle content
mysql> GRANT SELECT, INSERT, UPDATE ON Travel.Blogs TO 'bangalore_agent'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

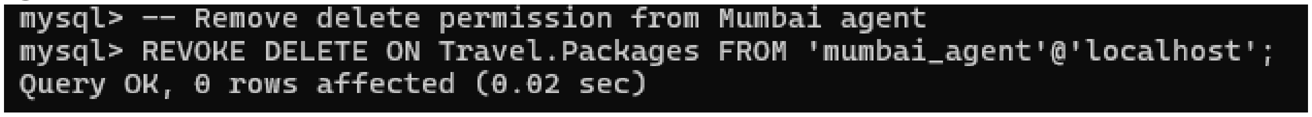
49. 

```
mysql> GRANT SELECT ON Travel.Destinations TO 'bangalore_agent'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

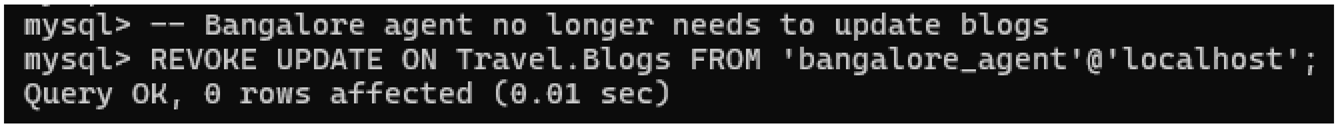
50. 

```
mysql> -- Kolkata staff (read-only access)
mysql> GRANT SELECT ON Travel.* TO 'kolkata_staff'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

**Revoking Access:**

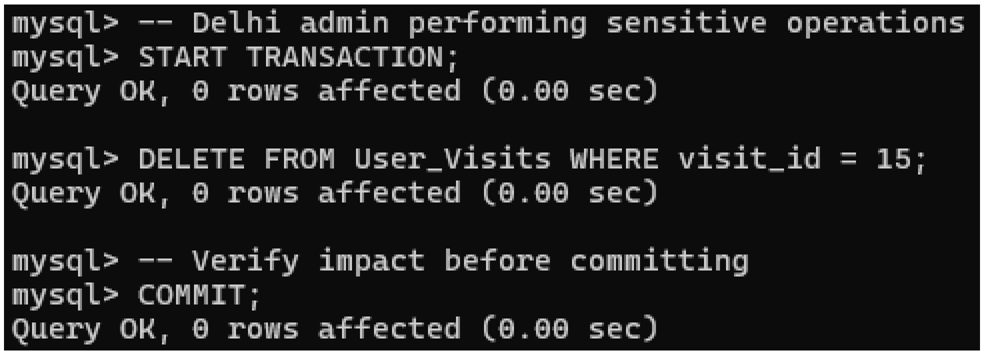
51. 

```
mysql> -- Remove delete permission from Mumbai agent
mysql> REVOKE DELETE ON Travel.Packages FROM 'mumbai_agent'@'localhost';
Query OK, 0 rows affected (0.02 sec)
```

52. 

```
mysql> -- Bangalore agent no longer needs to update blogs
mysql> REVOKE UPDATE ON Travel.Blogs FROM 'bangalore_agent'@'localhost';
Query OK, 0 rows affected (0.01 sec)
```

**Setting transaction for sensitive information:**



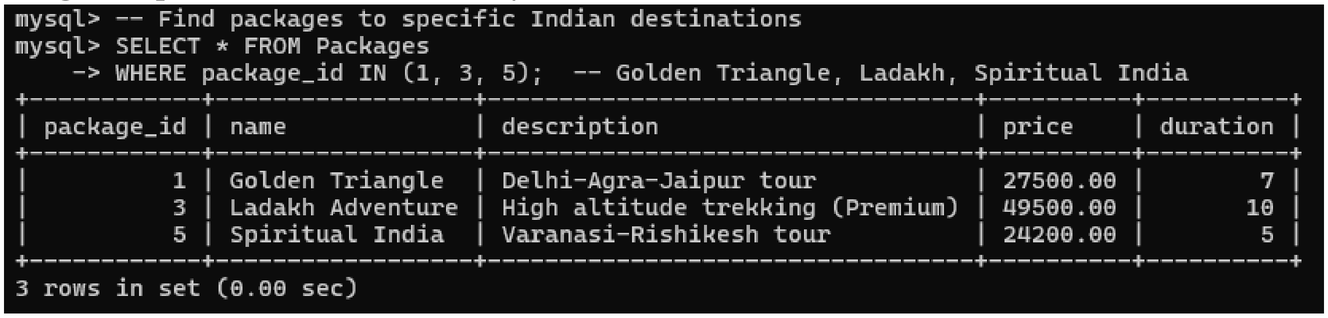
```
mysql> -- Delhi admin performing sensitive operations
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> DELETE FROM User_Visits WHERE visit_id = 15;
Query OK, 0 rows affected (0.00 sec)

mysql> -- Verify impact before committing
mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
```

53.

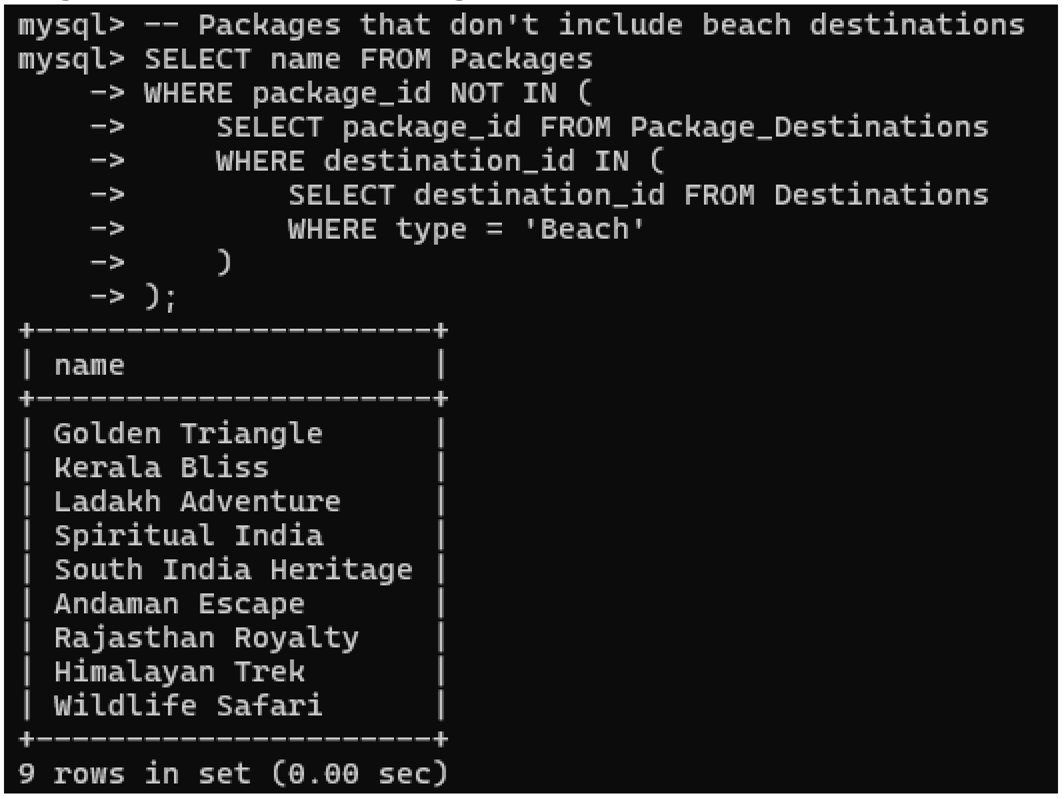
**Find packages to specific destinations (IN keyword):**



```
mysql> -- Find packages to specific Indian destinations
mysql> SELECT * FROM Packages
    -> WHERE package_id IN (1, 3, 5);  -- Golden Triangle, Ladakh, Spiritual India
+------------+------------------+-----------------------------------+----------+----------+
| package_id | name             | description                       | price    | duration |
+------------+------------------+-----------------------------------+----------+----------+
|          1 | Golden Triangle  | Delhi-Agra-Jaipur tour            | 27500.00 |        7 |
|          3 | Ladakh Adventure | High altitude trekking (Premium)  | 49500.00 |       10 |
|          5 | Spiritual India  | Varanasi-Rishikesh tour           | 24200.00 |        5 |
+------------+------------------+-----------------------------------+----------+----------+
3 rows in set (0.00 sec)
```

54.

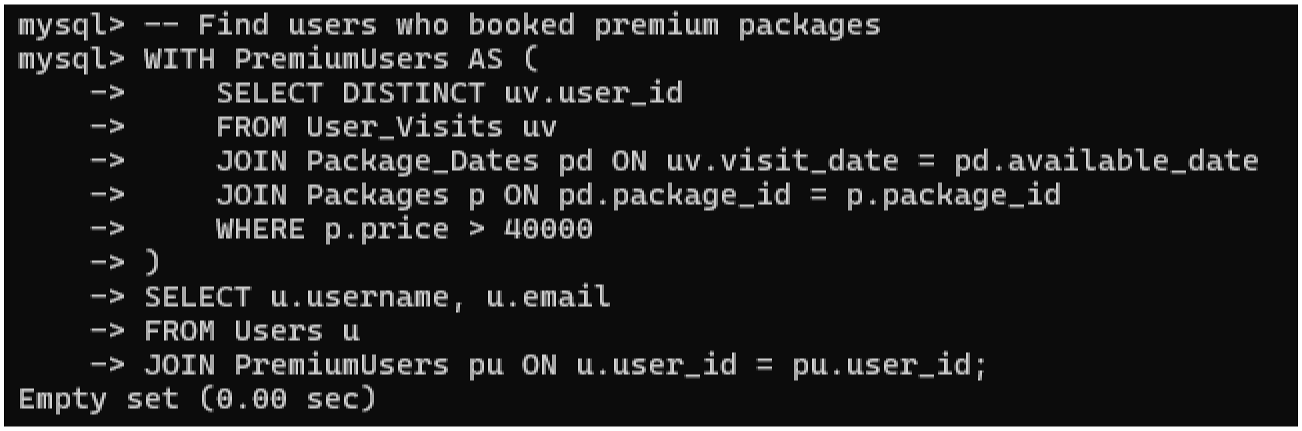
**Find packages with destination excluding beach (NOT IN):**



```
mysql> -- Packages that don't include beach destinations
mysql> SELECT name FROM Packages
    -> WHERE package_id NOT IN (
    ->     SELECT package_id FROM Package_Destinations
    ->     WHERE destination_id IN (
    ->         SELECT destination_id FROM Destinations
    ->         WHERE type = 'Beach'
    ->     )
    -> );
+----------------------+
| name                 |
+----------------------+
| Golden Triangle      |
| Kerala Bliss         |
| Ladakh Adventure     |
| Spiritual India      |
| South India Heritage |
| Andaman Escape       |
| Rajasthan Royalty    |
| Himalayan Trek       |
| Wildlife Safari      |
+----------------------+
9 rows in set (0.00 sec)
```

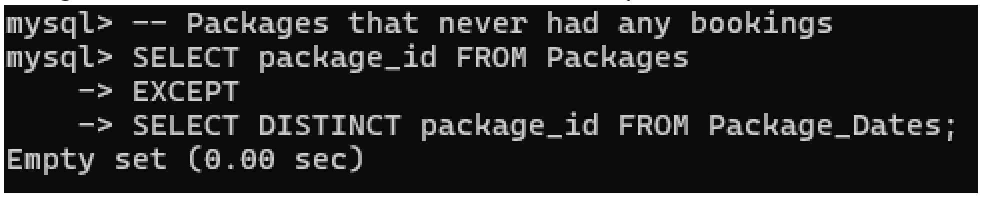
55.

**Find users who booked premium packages (WITH keyword):**

56. 

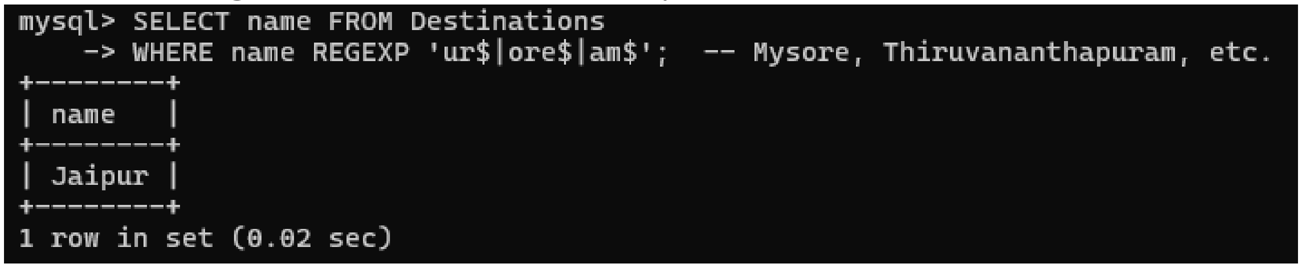
```
mysql> -- Find users who booked premium packages
mysql> WITH PremiumUsers AS (
    ->     SELECT DISTINCT uv.user_id
    ->     FROM User_Visits uv
    ->     JOIN Package_Dates pd ON uv.visit_date = pd.available_date
    ->     JOIN Packages p ON pd.package_id = p.package_id
    ->     WHERE p.price > 40000
    -> )
    -> SELECT u.username, u.email
    -> FROM Users u
    -> JOIN PremiumUsers pu ON u.user_id = pu.user_id;
Empty set (0.00 sec)
```

**Find packages that have no destination (EXCEPT keyword):**

57. 

```
mysql> -- Packages that never had any bookings
mysql> SELECT package_id FROM Packages
    -> EXCEPT
    -> SELECT DISTINCT package_id FROM Package_Dates;
Empty set (0.00 sec)
```

**Find Destinations ending with ur, ore, am (REGEXP keyword):**

58. 

```
mysql> SELECT name FROM Destinations
    -> WHERE name REGEXP 'ur$|ore$|am$';  -- Mysore, Thiruvananthapuram, etc.
+--------+
| name   |
+--------+
| Jaipur |
+--------+
1 row in set (0.02 sec)
```

# VI. Project demonstration

**Tools/Software/Libraries Used**

1. Database Management: MySQL (for SQL operations and database design)
2. GUI Tool: MySQL Workbench (for visualizing the database schema and executing queries)
3. Version Control: GitHub (for project documentation and collaboration)
4. SmartDraw (for ER Diagram)

**GUI:**

***(Since no GUI was explicitly developed, the demonstration focused on SQL operations and database interactions.)***

## VII. Self -Learning beyond classroom

**New Concepts Learned Independently**

1. User Management in SQL:
   - **Created users** (CREATE USER) and **granted specific permissions** (GRANT SELECT/INSERT).
   - Example:
   - sql
   - Copy
   - CREATE USER 'delhi_admin'@'localhost' IDENTIFIED BY 'Secure@123';
   - GRANT ALL PRIVILEGES ON Travel.* TO 'delhi_admin'@'localhost';

2. Advanced SQL Keywords
   - **WITH**: Simplified complex queries (e.g., analyzing premium package bookings).
   - **EXCEPT**: Found packages with no bookings.
   - **REGEXP**: Filtered Indian destinations (e.g., WHERE name REGEXP 'ur$|ore$' for cities like Jaipur).

## VIII. Learning from the Project

- Practical Database Design
  - Learned to normalize data (1NF to 3NF) and handle relationships (e.g., User_Visits as a resolver table).
- Real-World SQL Skills
  - Executed 50+ queries covering joins, aggregations, and transactions.
- Security Awareness
  - Implemented password hashing (bcrypt) and role-based access control (e.g., CREATE ROLE 'indian_management').

## IX. Challenges Faced

**Complex Joins**

- Initially struggled with multi-table joins (e.g., linking User_Visits with Packages and Destinations).
- Solution: Broke queries into subqueries using the WITH keyword.

## X. Conclusion

This project gave us hands-on experience in building a functional travel itinerary database system using SQL. Through designing the database structure and implementing various queries, we developed practical skills in data modeling and management. Working with **real-world scenarios**—such as tracking visits to popular Indian destinations like the Taj Mahal and the beaches of Goa—helped us understand how to structure data effectively.

We expanded our **SQL** knowledge beyond the basics by learning to use advanced features like Common Table Expressions (**WITH**) for complex queries and **REGEXP** for pattern matching in text data. The project also introduced us to important database administration concepts, including user privilege management using **GRANT** and **REVOKE** commands.

While working on this project, we encountered and solved several **technical challenges**, including query optimization and maintaining data **integrity**. These experiences not only improved our problem-solving abilities but also gave us confidence in handling real-world database systems.

Looking ahead, we are excited to enhance this project by developing a user interface and adding **intelligent recommendation features**. Overall, this practical exercise significantly **strengthened** our SQL expertise and database management capabilities, preparing us for more complex data-driven projects in the future.