TOUR GUIDE DATABASE

Database Management System

Muhammad Rizwan

Muhammad Faheem 42346

Muhammad Bilal

Naseem

JUNE 27, 2025

IQRA UNIVERSITY H-9

ISLAMABAD, PAKISTAN

Abstract

The **Tourism Guide Database** project is a comprehensive relational database system designed to facilitate the management of tourism-related data such as users, guides, tours, bookings, and reviews. It supports multilingual guides, tour planning across various global locations, and a secure feedback and booking system. The project follows normalization standards and applies indexes for performance efficiency. Structured Query Language (SQL) is used for all data definition and manipulation operations. The database also includes views, constraints, and role-based user access to simulate a real-world, secure, and scalable tourism management platform.

Introduction

Tourism is one of the fastest-growing industries worldwide. With the increased use of technology in travel planning, there is a need for digital systems that can manage tourism data efficiently. This project introduces a structured and normalized database named **Tourism Guide Database**, developed using SQL. The database manages user registrations, multilingual guides, available tours, city/location information, tour bookings, and customer reviews. It also considers data integrity, query performance, and security measures to allow role-based access to sensitive information.

Objectives

- To design and implement a relational database for managing tourism services.
- To maintain structured data about users, guides, languages, tours, and bookings.
- To establish relationships and enforce referential integrity using primary and foreign keys.
- To implement access controls for users and administrators using MySQL roles and privileges.
- To ensure optimized query performance through indexing.
- To provide views for frontend or reporting use without exposing sensitive backend information.

Methodology

The methodology followed during the project includes:

a. Requirement Gathering:
Understanding the essential components of a tourism system such as users, guides, tours, locations, bookings, and reviews.

b. Relational Schema Design:
Dividing the system into logical tables like Users, Guides, Languages, Tours, Bookings, etc., and applying normalization to reduce redundancy.

c. SQL-Based Implementation:

- Use of SQL DDL for creating tables with constraints.
- Use of SQL DML for inserting sample data.
- Use of joins and indexes to test relationships and optimize queries.
- Creation of views and indexes for performance.

d. Security & Access Control:
MySQL users and roles were created for admin, customers, and guides. Access to operations such as SELECT, INSERT, UPDATE, etc., was controlled based on roles.

e. Testing and Query Execution:
Testing included verifying foreign key relationships, executing join queries, and ensuring views return the expected data.

Database Design (Schema Overview)

a. Users

- user_id (PK), first_name, last_name, email (unique), password_hash, created_at, update_at

b. Guides

- guide_id (PK), bio, rating, created_at, updated_at, user_id (FK to Users)

c. Languages

- language_id (PK), lang_name, iso_code (unique)

d. Guide_Languages (Many-to-Many)

- Composite PK: (guide_id, language_id)
- FKs to Guides and Languages

e. Locations

- location_id (PK), city, country, latitude, longitude, description

f. Tours

- tour_id (PK), title, guide_id (FK), price, duration_days, created_at, updated_at

g. Tour_Locations (Many-to-Many)

- Composite PK: (tour_id, location_id)
- FKs to Tours and Locations

h. Bookings

- booking_id (PK), user_id (FK), tour_id (FK), booking_date, num_people, price, status

i. Reviews

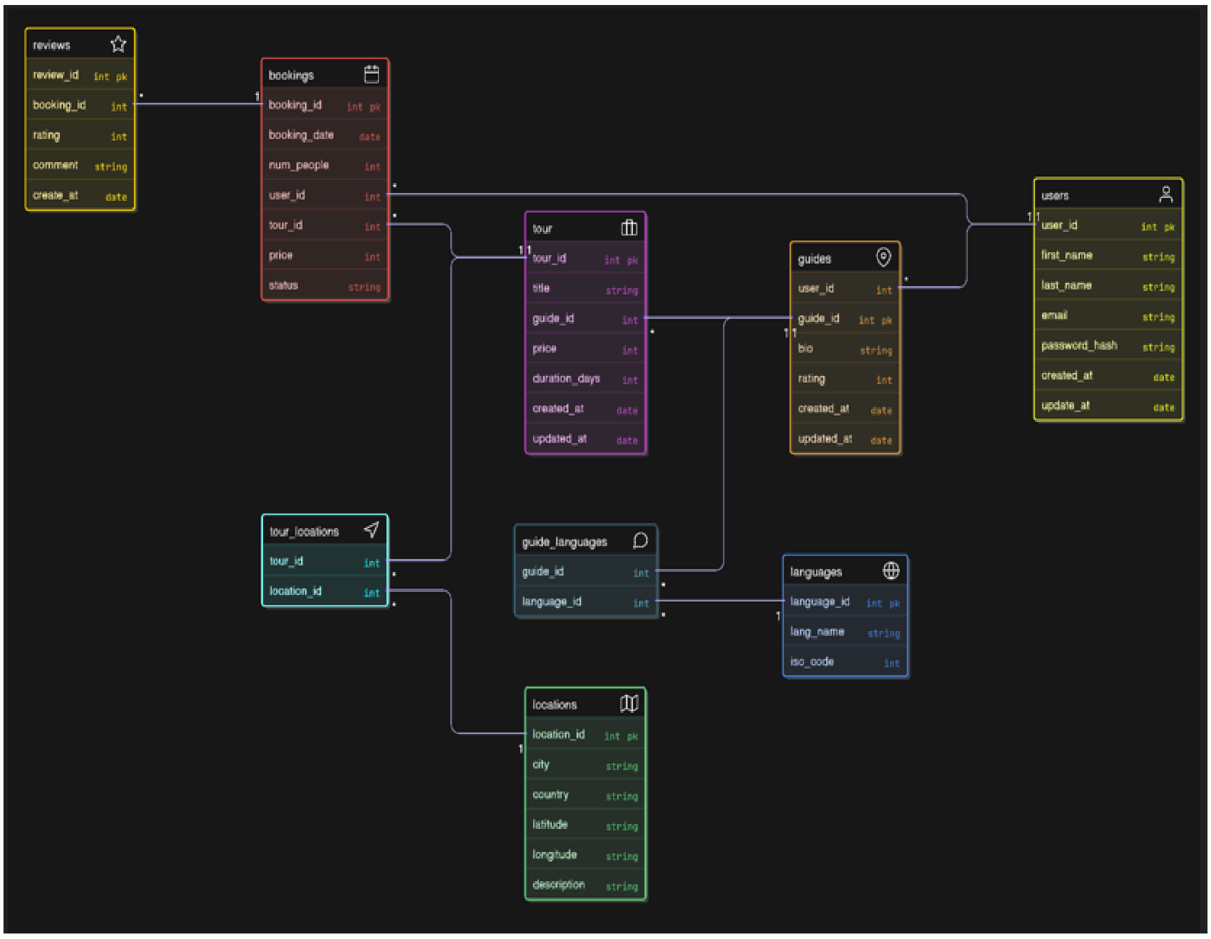
- review_id (PK), booking_id (FK), rating, comment, create_at

Indexes:

- Users.email, Bookings.user_id, Reviews.booking_id, Languages.iso_code
- Composite indexes were used where needed (e.g., Guide_Languages)

SCREENSHOTS AND IMPLEMENTATION:

ER DIAGRAM:

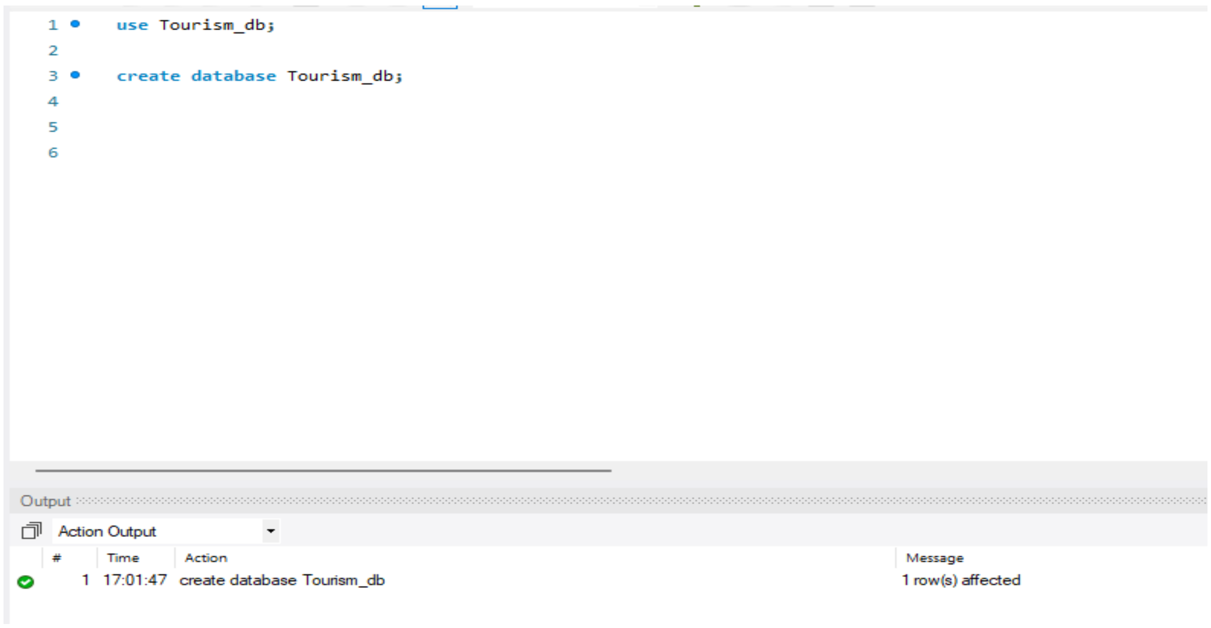


Queries:

1.Create Database:

use Tourism_db;

create database Tourism_db;



```
use Tourism_db;

create database Tourism_db;
```

```
#   Time       Action                        Message
1   17:01:47   create database Tourism_db    1 row(s) affected
```

2.Create Table And Insert Values:

```
create table Users(
user_id             int(11) PRIMARY KEY,
first_name          varchar(255),
last_name           varchar(255),
email               varchar(255) UNIQUE,
password_hash varchar(255),
created_at          date,
update_at           date

);
```

insert into Users(user_id, first_name, last_name, email, password_hash, created_at, update_at)

values(1, "Muhammad", "Faheem", "faheem@gmail.com", "H5G7E8S4G5", "2022-05-25", "2025-06-12"),

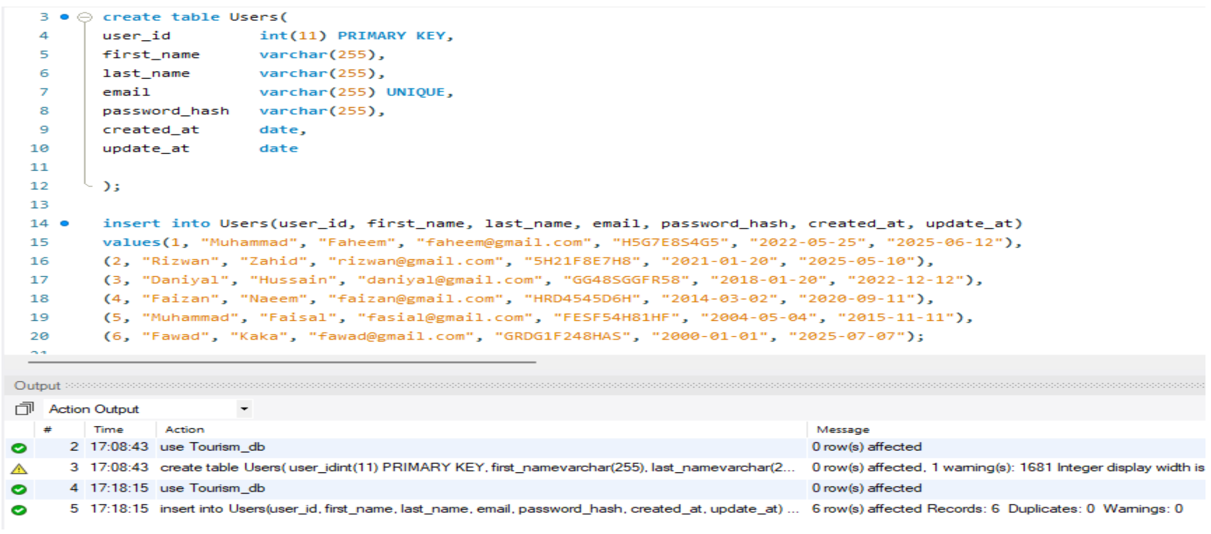
(2, "Rizwan", "Zahid", "rizwan@gmail.com", "5H21F8E7H8", "2021-01-20", "2025-05-10"),

(3, "Daniyal", "Hussain", "daniyal@gmail.com", "GG48SGGFR58", "2018-01-20", "2022-12-12"),

(4, "Faizan", "Naeem", "faizan@gmail.com", "HRD4545D6H", "2014-03-02", "2020-09-11"),

(5, "Muhammad", "Faisal", "fasial@gmail.com", "FESF54H81HF", "2004-05-04", "2015-11-11"),

(6, "Fawad", "Kaka", "fawad@gmail.com", "GRDG1F248HAS", "2000-01-01", "2025-07-07");



```
create table Users(
user_id          int(11) PRIMARY KEY,
first_name       varchar(255),
last_name        varchar(255),
email            varchar(255) UNIQUE,
password_hash    varchar(255),
created_at       date,
update_at        date

);

insert into Users(user_id, first_name, last_name, email, password_hash, created_at, update_at)
values(1, "Muhammad", "Faheem", "faheem@gmail.com", "H5G7E8S4G5", "2022-05-25", "2025-06-12"),
(2, "Rizwan", "Zahid", "rizwan@gmail.com", "5H21F8E7H8", "2021-01-20", "2025-05-10"),
(3, "Daniyal", "Hussain", "daniyal@gmail.com", "GG48SGGFR58", "2018-01-20", "2022-12-12"),
(4, "Faizan", "Naeem", "faizan@gmail.com", "HRD4545D6H", "2014-03-02", "2020-09-11"),
(5, "Muhammad", "Faisal", "fasial@gmail.com", "FESF54H81HF", "2004-05-04", "2015-11-11"),
(6, "Fawad", "Kaka", "fawad@gmail.com", "GRDG1F248HAS", "2000-01-01", "2025-07-07");
```

```
#  Time      Action                                                                                 Message
2  17:08:43  use Tourism_db                                                                         0 row(s) affected
3  17:08:43  create table Users( user_idint(11) PRIMARY KEY, first_namevarchar(255), last_namevarchar(2...   0 row(s) affected, 1 warning(s): 1681 Integer display width is
4  17:18:15  use Tourism_db                                                                         0 row(s) affected
5  17:18:15  insert into Users(user_id, first_name, last_name, email, password_hash, created_at, update_at) ...   6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
```

CREATE TABLE AND INSERT VALUES:

create table guides(

guide_id   int(11) PRIMARY KEY,

bio   varchar(255),

rating   int(11),

created_at   date,

updated_at   date,

user_id   int,

FOREIGN KEY(user_id) references Users(user_id)

);

INSERT INTO guides (guide_id, bio, rating, created_at, updated_at, user_id)

VALUES(1, "Experienced and passionate guide", 8, "2021-05-06", "2025-05-01", 3),

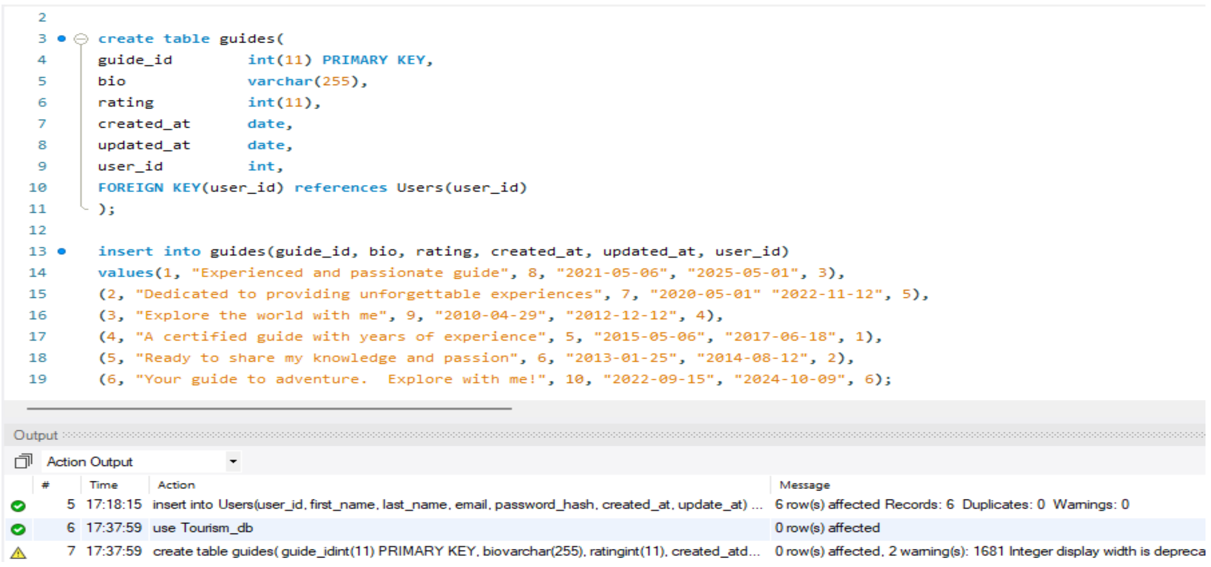
(2, "Dedicated to providing unforgettable", 7, "2020-05-01", "2022-11-12", 5),

(3, "Explore the world with me", 9, "2010-04-29", "2012-12-12", 4),

(4, "A certified guide with years of experience", 5, "2015-05-06", "2017-06-18", 1),

(5, "Ready to share my knowledge and passion", 6, "2013-01-25", "2014-08-12", 2),

(6, "Your guide to adventure. Explore with me!", 10, "2022-09-15", "2024-10-09", 6);



```
create table guides(
guide_id        int(11) PRIMARY KEY,
bio             varchar(255),
rating          int(11),
created_at      date,
updated_at      date,
user_id         int,
FOREIGN KEY(user_id) references Users(user_id)
);

insert into guides(guide_id, bio, rating, created_at, updated_at, user_id)
values(1, "Experienced and passionate guide", 8, "2021-05-06", "2025-05-01", 3),
(2, "Dedicated to providing unforgettable experiences", 7, "2020-05-01" "2022-11-12", 5),
(3, "Explore the world with me", 9, "2010-04-29", "2012-12-12", 4),
(4, "A certified guide with years of experience", 5, "2015-05-06", "2017-06-18", 1),
(5, "Ready to share my knowledge and passion", 6, "2013-01-25", "2014-08-12", 2),
(6, "Your guide to adventure.  Explore with me!", 10, "2022-09-15", "2024-10-09", 6);
```

```
Output
Action Output
 #  Time      Action                                                                              Message
 5  17:18:15  insert into Users(user_id, first_name, last_name, email, password_hash, created_at, update_at) ...  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
 6  17:37:59  use Tourism_db                                                                      0 row(s) affected
 7  17:37:59  create table guides( guide_idint(11) PRIMARY KEY, biovarchar(255), ratingint(11), created_atd...  0 row(s) affected, 2 warning(s): 1681 Integer display width is depreca
```

CREATE TABLE AND INSERT VALUES:

create table Languages(

language_id	int(11) PRIMARY KEY,

lang_name		varchar(255),

iso_code	int(11) UNIQUE

);

insert into Languages(language_id, lang_name, iso_code)

values(1, "English", 1234),

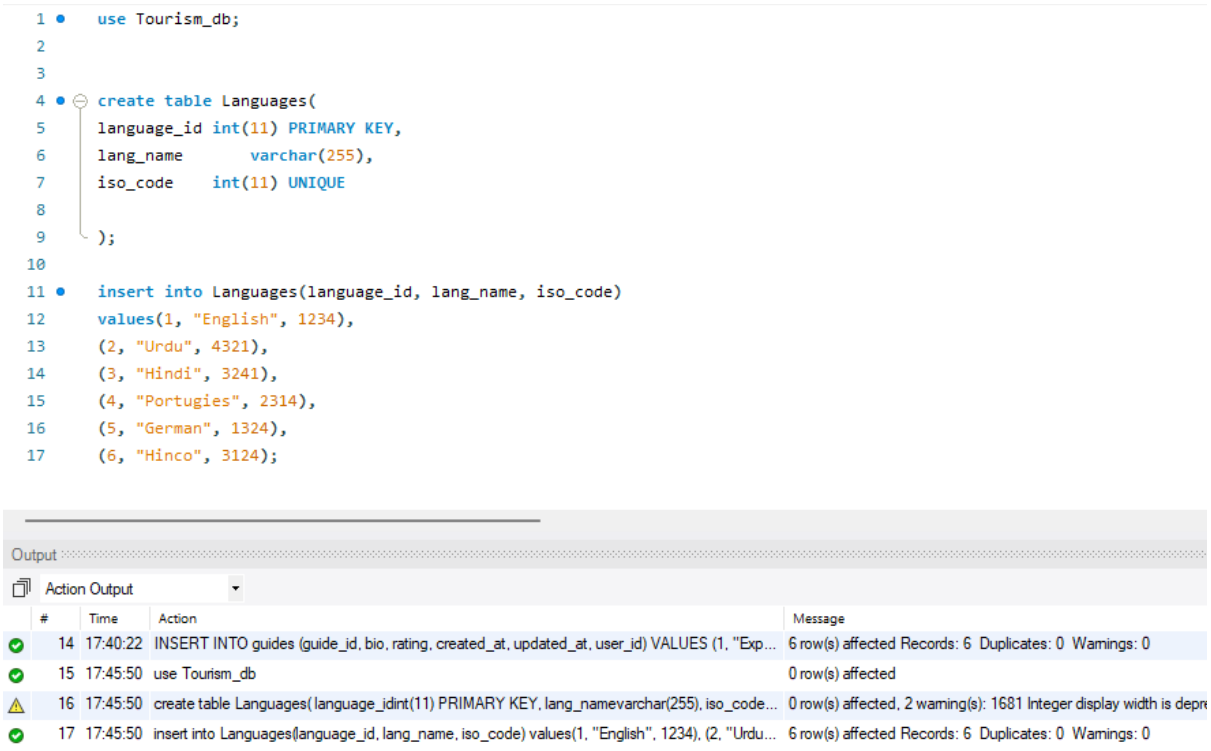
(2, "Urdu", 4321),

(3, "Hindi", 3241),

(4, "Portugies", 2314),

(5, "German", 1324),

(6, "Hinco", 3124);



```
use Tourism_db;


create table Languages(
language_id int(11) PRIMARY KEY,
lang_name       varchar(255),
iso_code    int(11) UNIQUE

);

insert into Languages(language_id, lang_name, iso_code)
values(1, "English", 1234),
(2, "Urdu", 4321),
(3, "Hindi", 3241),
(4, "Portugies", 2314),
(5, "German", 1324),
(6, "Hinco", 3124);
```

```
Output
Action Output
   #   Time      Action                                                                                      Message
✓  14  17:40:22  INSERT INTO guides (guide_id, bio, rating, created_at, updated_at, user_id) VALUES (1, "Exp...  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
✓  15  17:45:50  use Tourism_db                                                                              0 row(s) affected
⚠  16  17:45:50  create table Languages( language_idint(11) PRIMARY KEY, lang_namevarchar(255), iso_code...    0 row(s) affected, 2 warning(s): 1681 Integer display width is depre
✓  17  17:45:50  insert into Languages(language_id, lang_name, iso_code) values(1, "English", 1234), (2, "Urdu...  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
```

CREATE TABLE AND INSERT VALUES:

create table Guide_Languages(

guide_id int,

language_id int,

FOREIGN KEY(guide_id) references Guides(guide_id),

FOREIGN KEY(language_id) references Languages(language_id)

);

insert into Guide_Languages(guide_id, language_id)

values(1,3),

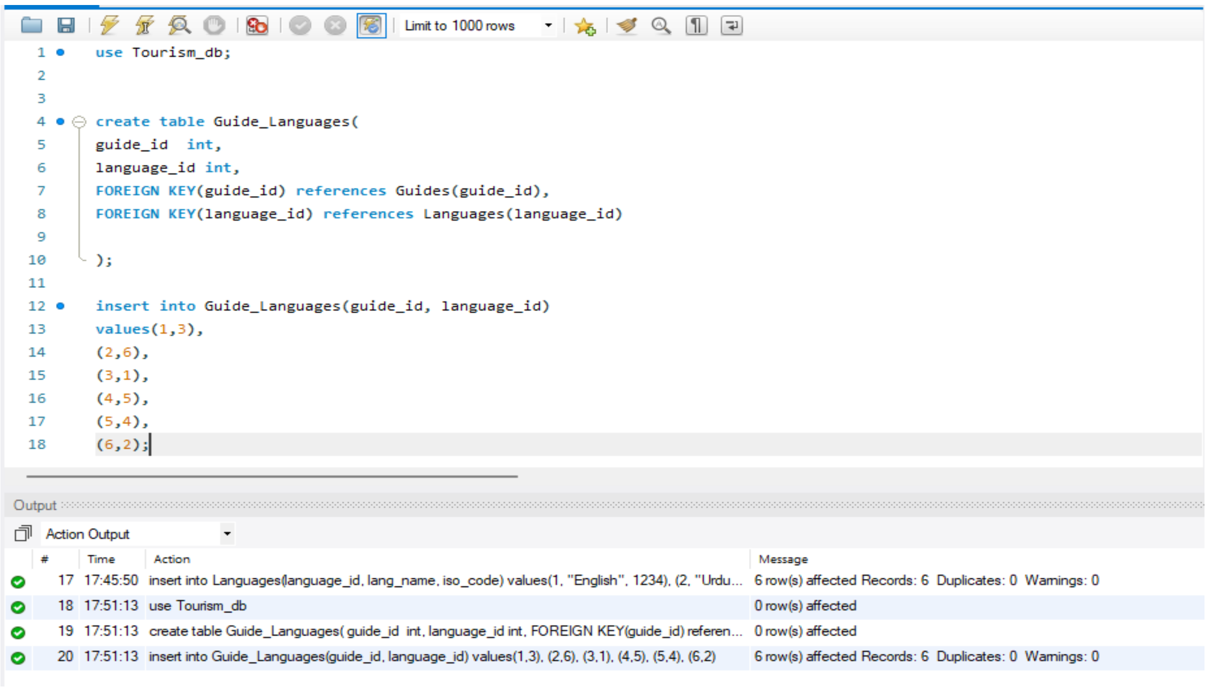
(2,6),

(3,1),

(4,5),

(5,4),

(6,2);



```
use Tourism_db;


create table Guide_Languages(
guide_id  int,
language_id int,
FOREIGN KEY(guide_id) references Guides(guide_id),
FOREIGN KEY(language_id) references Languages(language_id)

);

insert into Guide_Languages(guide_id, language_id)
values(1,3),
(2,6),
(3,1),
(4,5),
(5,4),
(6,2);
```

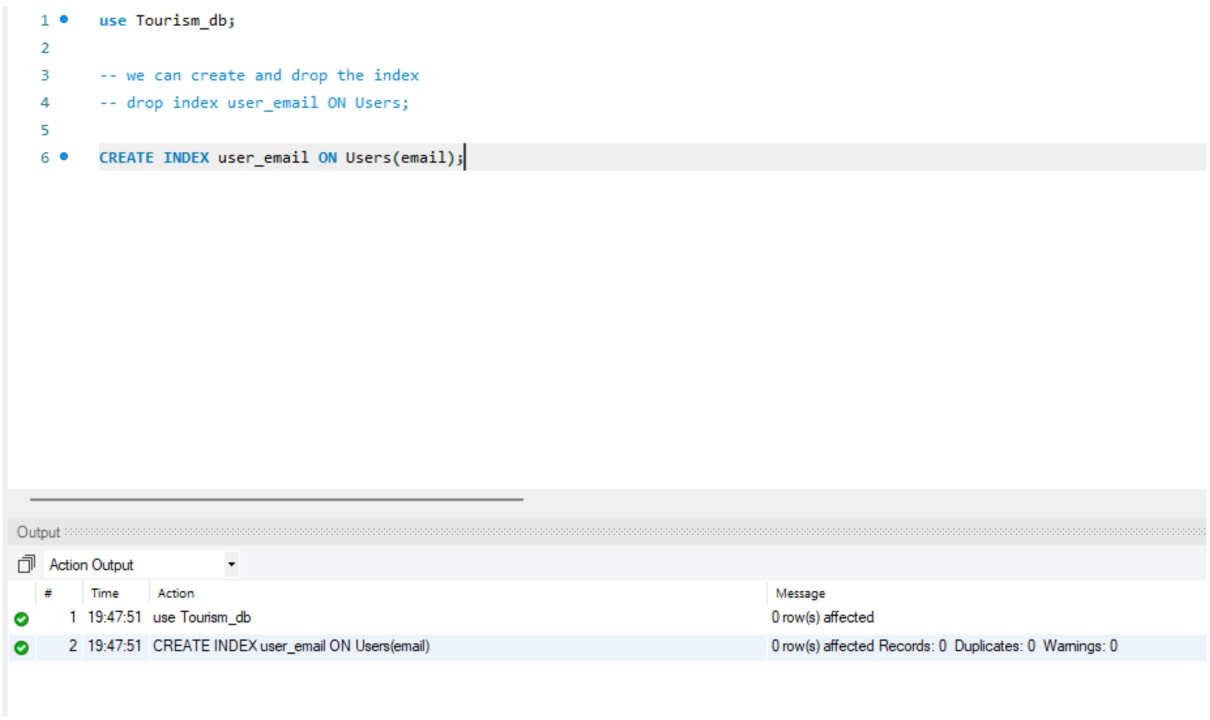
```
   #   Time      Action                                                                                     Message
   17  17:45:50  insert into Languages(language_id, lang_name, iso_code) values(1, "English", 1234), (2, "Urdu...  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
   18  17:51:13  use Tourism_db                                                                             0 row(s) affected
   19  17:51:13  create table Guide_Languages( guide_id  int, language_id int, FOREIGN KEY(guide_id) referen...  0 row(s) affected
   20  17:51:13  insert into Guide_Languages(guide_id, language_id) values(1,3), (2,6), (3,1), (4,5), (5,4), (6,2)  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
```

B. Index Strategy

Users.email IDX

Guides.email IDX (if Guides has its own email; otherwise index on the joined Users.email)

NOTE: Their is no email in Guides table



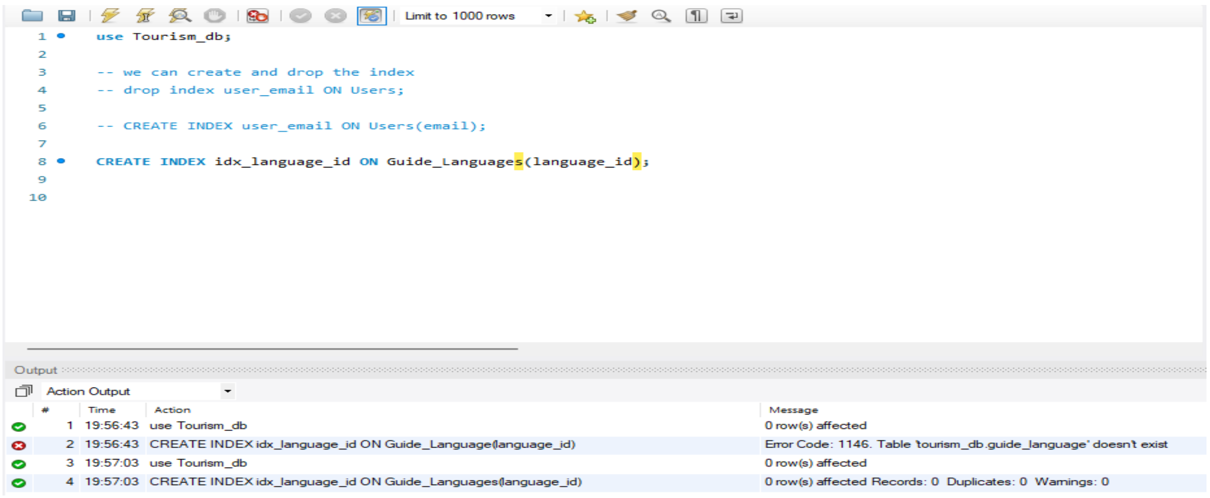
```
use Tourism_db;

-- we can create and drop the index
-- drop index user_email ON Users;

CREATE INDEX user_email ON Users(email);
```

```
Output
Action Output
#   Time      Action                                   Message
1   19:47:51  use Tourism_db                           0 row(s) affected
2   19:47:51  CREATE INDEX user_email ON Users(email)  0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0
```

Guide_Languages

Consider index on language_id for reverse lookup

QUERY: CREATE INDEX idx_language_id ON Guide_Languages(language_id);



```
use Tourism_db;

-- we can create and drop the index
-- drop index user_email ON Users;

-- CREATE INDEX user_email ON Users(email);

CREATE INDEX idx_language_id ON Guide_Languages(language_id);
```

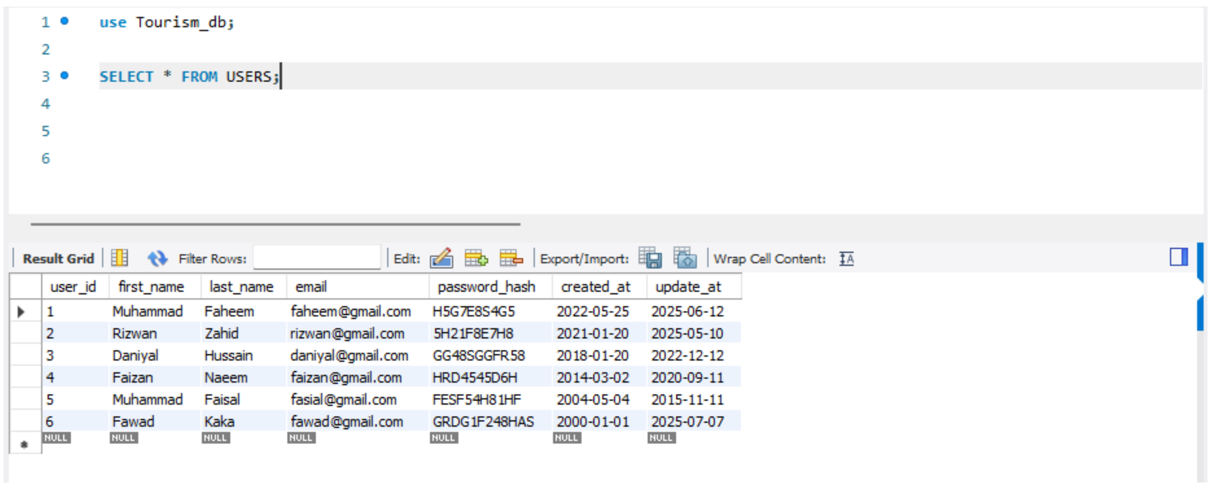
```
Output
Action Output
#   Time      Action                                                      Message
1   19:56:43  use Tourism_db                                              0 row(s) affected
2   19:56:43  CREATE INDEX idx_language_id ON Guide_Language(language_id)  Error Code: 1146. Table 'tourism_db.guide_language' doesn't exist
3   19:57:03  use Tourism_db                                              0 row(s) affected
4   19:57:03  CREATE INDEX idx_language_id ON Guide_Languages(language_id) 0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0
```

C. Sample Data & Verification

Prepare 5–10 example rows for each table

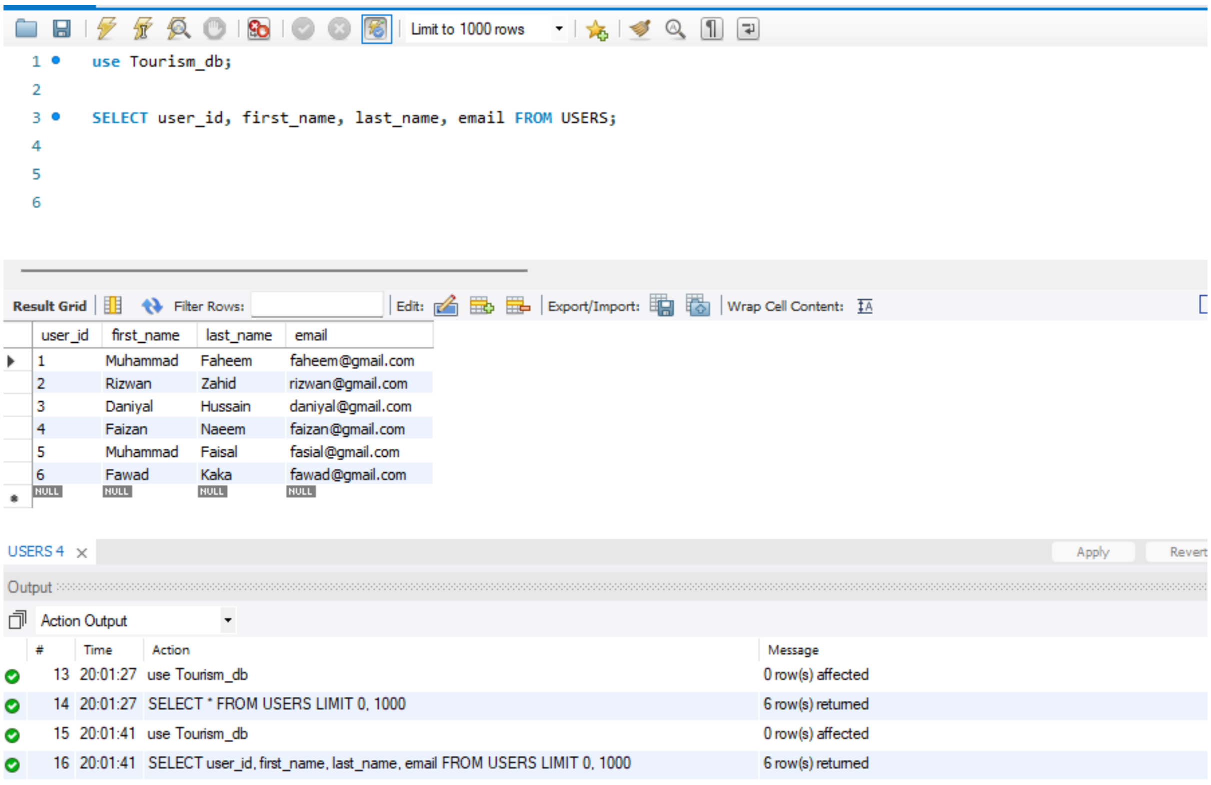
Test by writing SELECT queries to ensure joins work (e.g. fetch a guide's languages)

SAMPLE QUERIES:



```sql
use Tourism_db;

SELECT * FROM USERS;
```

user_id	first_name	last_name	email	password_hash	created_at	update_at
1	Muhammad	Faheem	faheem@gmail.com	H5G7E8S4G5	2022-05-25	2025-06-12
2	Rizwan	Zahid	rizwan@gmail.com	5H21F8E7H8	2021-01-20	2025-05-10
3	Daniyal	Hussain	daniyal@gmail.com	GG48SGGFR58	2018-01-20	2022-12-12
4	Faizan	Naeem	faizan@gmail.com	HRD4545D6H	2014-03-02	2020-09-11
5	Muhammad	Faisal	fasial@gmail.com	FESF54H81HF	2004-05-04	2015-11-11
6	Fawad	Kaka	fawad@gmail.com	GRDG1F248HAS	2000-01-01	2025-07-07
NULL	NULL	NULL	NULL	NULL	NULL	NULL



```sql
use Tourism_db;

SELECT user_id, first_name, last_name, email FROM USERS;
```

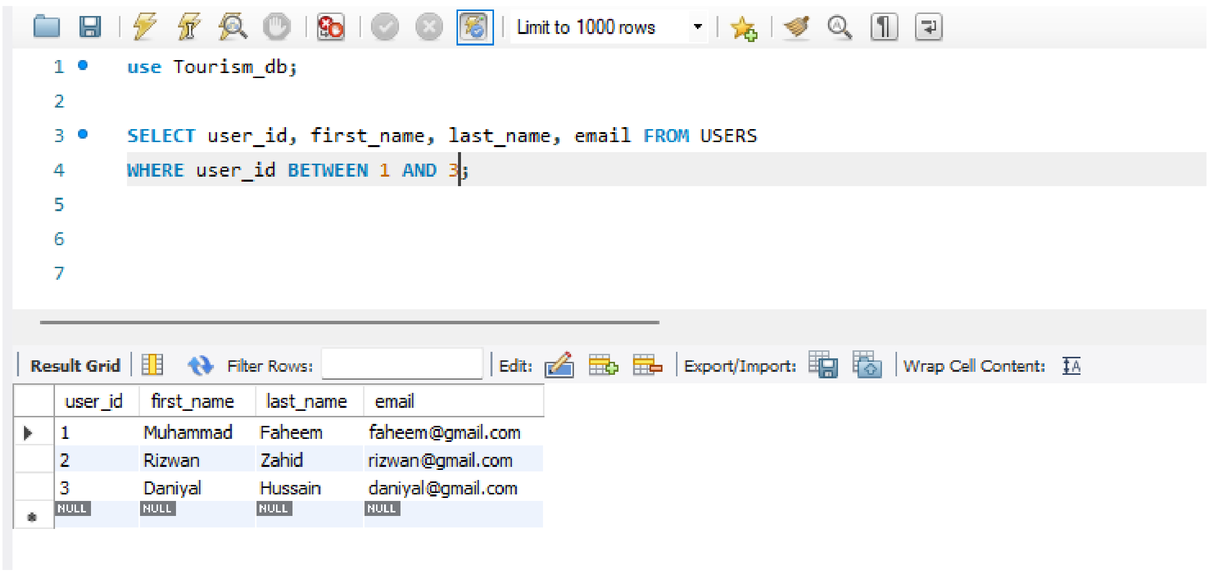
user_id	first_name	last_name	email
1	Muhammad	Faheem	faheem@gmail.com
2	Rizwan	Zahid	rizwan@gmail.com
3	Daniyal	Hussain	daniyal@gmail.com
4	Faizan	Naeem	faizan@gmail.com
5	Muhammad	Faisal	fasial@gmail.com
6	Fawad	Kaka	fawad@gmail.com
NULL	NULL	NULL	NULL

USERS 4

Output

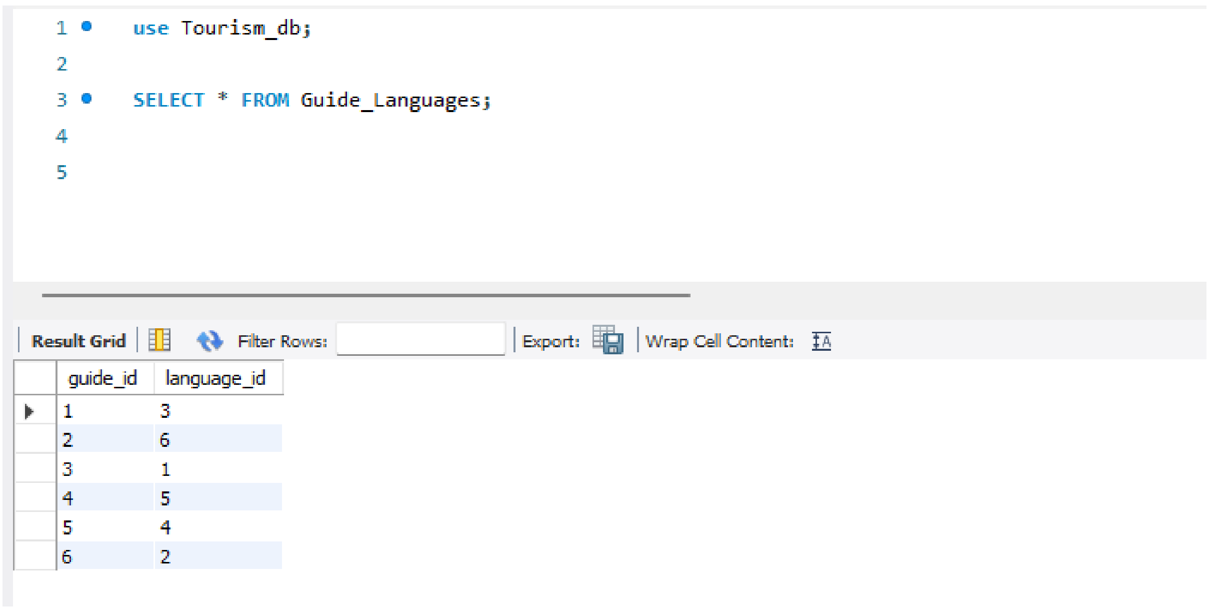
Action Output

#	Time	Action	Message
13	20:01:27	use Tourism_db	0 row(s) affected
14	20:01:27	SELECT * FROM USERS LIMIT 0, 1000	6 row(s) returned
15	20:01:41	use Tourism_db	0 row(s) affected
16	20:01:41	SELECT user_id, first_name, last_name, email FROM USERS LIMIT 0, 1000	6 row(s) returned



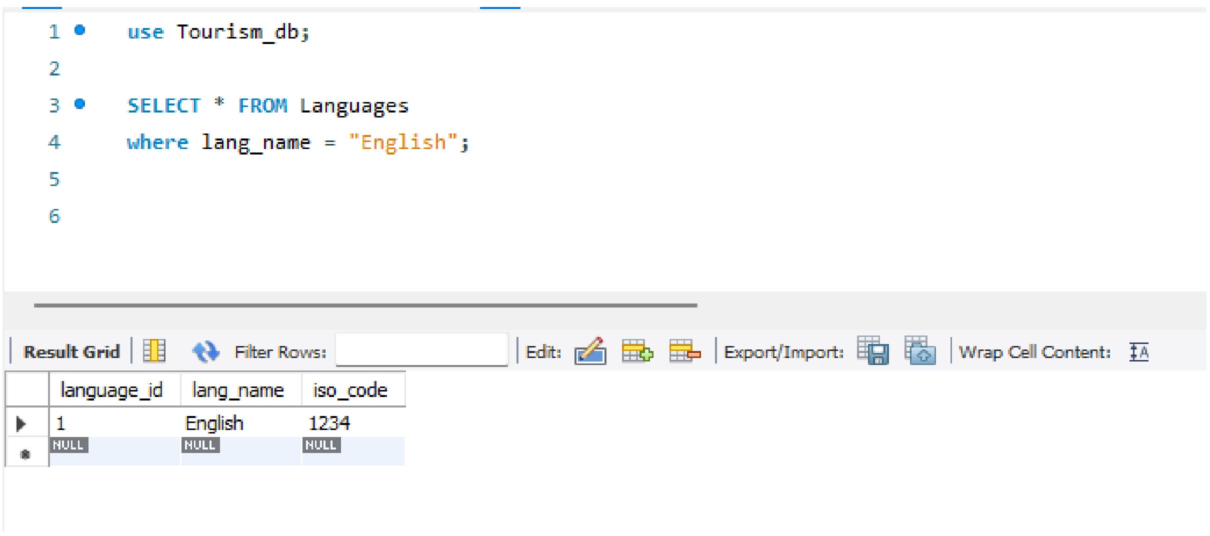
```sql
use Tourism_db;

SELECT user_id, first_name, last_name, email FROM USERS
WHERE user_id BETWEEN 1 AND 3;
```

user_id	first_name	last_name	email
1	Muhammad	Faheem	faheem@gmail.com
2	Rizwan	Zahid	rizwan@gmail.com
3	Daniyal	Hussain	daniyal@gmail.com
NULL	NULL	NULL	NULL



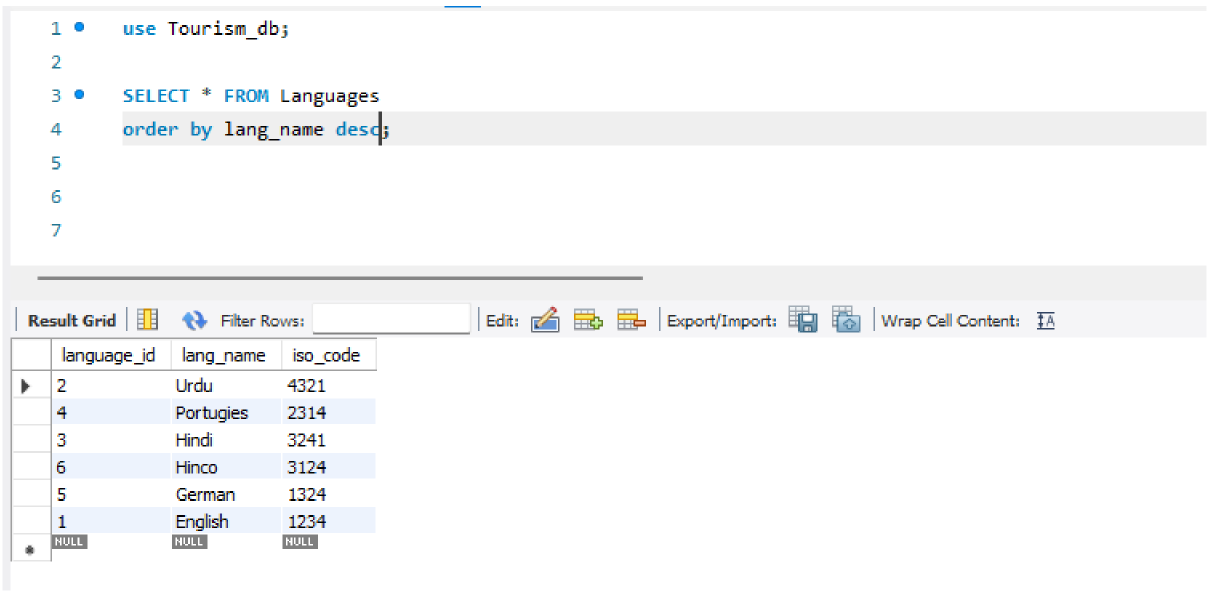
```sql
use Tourism_db;

SELECT * FROM Guide_Languages;
```

guide_id	language_id
1	3
2	6
3	1
4	5
5	4
6	2



```sql
use Tourism_db;

SELECT * FROM Languages
where lang_name = "English";
```

language_id	lang_name	iso_code
1	English	1234
NULL	NULL	NULL



```
use Tourism_db;

SELECT * FROM Languages
order by lang_name desc;
```

	language_id	lang_name	iso_code
▶	2	Urdu	4321
	4	Portugies	2314
	3	Hindi	3241
	6	Hinco	3124
	5	German	1324
	1	English	1234
*	NULL	NULL	NULL

CREATE TABLE AND INSERT VALUES:

create table Locations(

location_id int(11) PRIMARY KEY,

city varchar(255),

country varchar(255),

latitude varchar(255),

longitude varchar(255),

description varchar(10000)

);

insert into Locations(location_id, city, country, latitude, longitude, description)

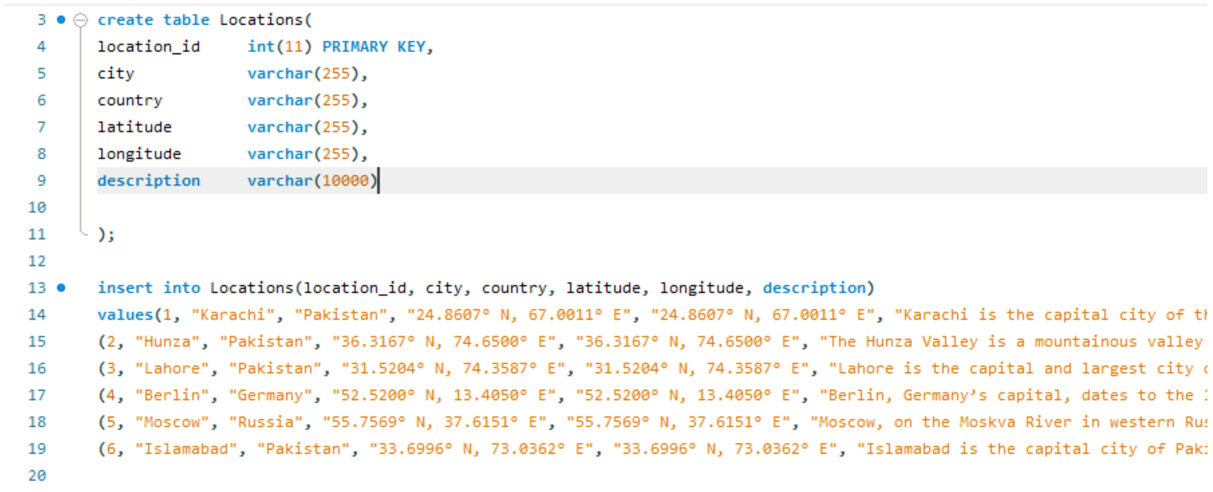
values(1, "Karachi", "Pakistan", "24.8607° N, 67.0011° E", "24.8607° N, 67.0011° E", "Karachi is the capital city of the province of Sindh, Pakistan. It is the largest city in Pakistan and 12th largest in the world"),

(2, "Hunza", "Pakistan", "36.3167° N, 74.6500° E", "36.3167° N, 74.6500° E", "The Hunza Valley is a mountainous valley located in the northern region of the Gilgit-Baltistan, Pakistan"),

(3, "Lahore", "Pakistan", "31.5204° N, 74.3587° E", "31.5204° N, 74.3587° E", "Lahore is the capital and largest city of the Pakistani province of Punjab. It is the second-largest city in Pakistan"), (4, "Berlin", "Germany", "52.5200° N, 13.4050° E", "52.5200° N, 13.4050° E", "Berlin, Germany’s capital, dates to the 13th century"),

(5, "Moscow", "Russia", "55.7569° N, 37.6151° E", "55.7569° N, 37.6151° E", "Moscow, on the Moskva River in western Russia, is the nation’s cosmopolitan capital"),

(6, "Islamabad", "Pakistan", "33.6996° N, 73.0362° E", "33.6996° N, 73.0362° E", "Islamabad is the capital city of Pakistan. It is the country's tenth-most populous city ");



```
create table Locations(
location_id     int(11) PRIMARY KEY,
city            varchar(255),
country         varchar(255),
latitude        varchar(255),
longitude       varchar(255),
description     varchar(10000)

);

insert into Locations(location_id, city, country, latitude, longitude, description)
values(1, "Karachi", "Pakistan", "24.8607° N, 67.0011° E", "24.8607° N, 67.0011° E", "Karachi is the capital city of th
(2, "Hunza", "Pakistan", "36.3167° N, 74.6500° E", "36.3167° N, 74.6500° E", "The Hunza Valley is a mountainous valley
(3, "Lahore", "Pakistan", "31.5204° N, 74.3587° E", "31.5204° N, 74.3587° E", "Lahore is the capital and largest city
(4, "Berlin", "Germany", "52.5200° N, 13.4050° E", "52.5200° N, 13.4050° E", "Berlin, Germany’s capital, dates to the
(5, "Moscow", "Russia", "55.7569° N, 37.6151° E", "55.7569° N, 37.6151° E", "Moscow, on the Moskva River in western Ru
(6, "Islamabad", "Pakistan", "33.6996° N, 73.0362° E", "33.6996° N, 73.0362° E", "Islamabad is the capital city of Pak
```

CREATE TABLE AND INSERT VALUES:

```
create table Tour(
tour_id             int(11) PRIMARY KEY,
title               varchar(255),
guide_id            int,
price               int(11),
duration_days   int(11),
created_at          date,
updated_at          date,
FOREIGN KEY(guide_id) references Guides(guide_id)
);
```

insert into Tour(tour_id, title, guide_id, price, duration_days, created_at, updated_at)

values(1, "Mardan", 6, 35000, 4, "2015-05-01", "2016-01-01"),

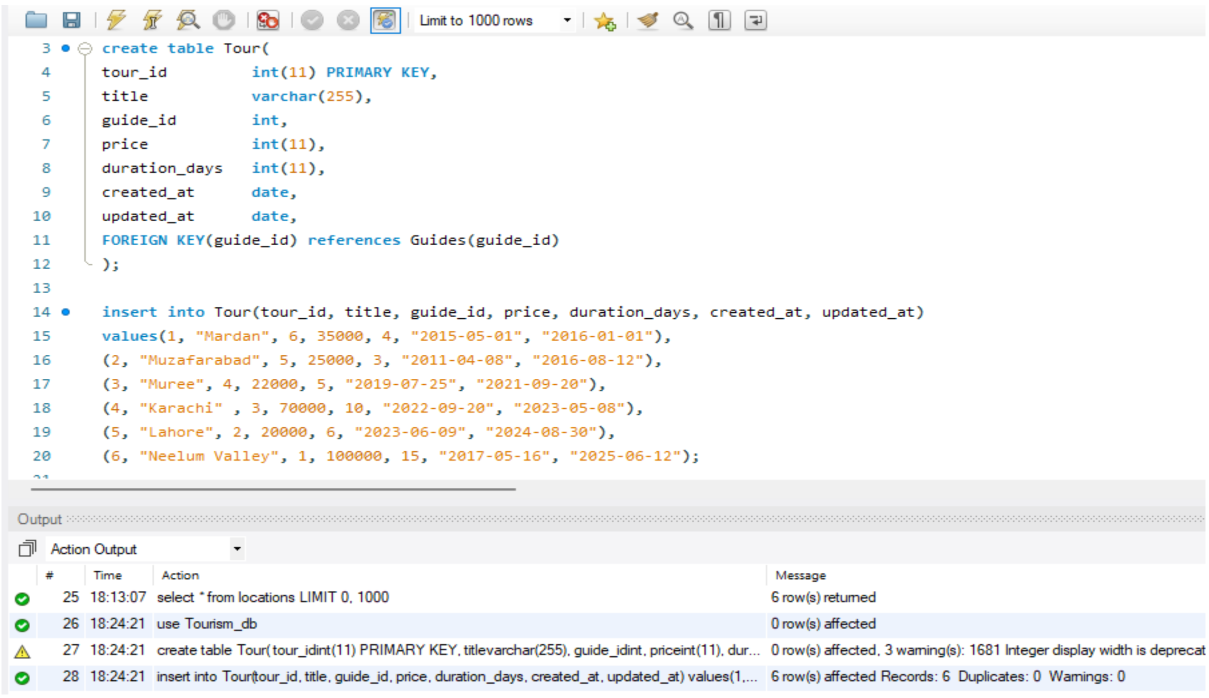
(2, "Muzafarabad", 5, 25000, 3, "2011-04-08", "2016-08-12"),

(3, "Muree", 4, 22000, 5, "2019-07-25", "2021-09-20"),

(4, "Karachi" , 3, 70000, 10, "2022-09-20", "2023-05-08"),

(5, "Lahore", 2, 20000, 6, "2023-06-09", "2024-08-30"),

(6, "Neelum Valley", 1, 100000, 15, "2017-05-16", "2025-06-12");



Limit to 1000 rows

```
create table Tour(
tour_id         int(11) PRIMARY KEY,
title           varchar(255),
guide_id        int,
price           int(11),
duration_days   int(11),
created_at      date,
updated_at      date,
FOREIGN KEY(guide_id) references Guides(guide_id)
);

insert into Tour(tour_id, title, guide_id, price, duration_days, created_at, updated_at)
values(1, "Mardan", 6, 35000, 4, "2015-05-01", "2016-01-01"),
(2, "Muzafarabad", 5, 25000, 3, "2011-04-08", "2016-08-12"),
(3, "Muree", 4, 22000, 5, "2019-07-25", "2021-09-20"),
(4, "Karachi" , 3, 70000, 10, "2022-09-20", "2023-05-08"),
(5, "Lahore", 2, 20000, 6, "2023-06-09", "2024-08-30"),
(6, "Neelum Valley", 1, 100000, 15, "2017-05-16", "2025-06-12");
```

Output

Action Output

	#	Time	Action	Message
✓	25	18:13:07	select * from locations LIMIT 0, 1000	6 row(s) returned
✓	26	18:24:21	use Tourism_db	0 row(s) affected
⚠	27	18:24:21	create table Tour(tour_idint(11) PRIMARY KEY, titlevarchar(255), guide_idint, priceint(11), dur...	0 row(s) affected, 3 warning(s): 1681 Integer display width is deprecat
✓	28	18:24:21	insert into Tour(tour_id, title, guide_id, price, duration_days, created_at, updated_at) values(1,...	6 row(s) affected Records: 6 Duplicates: 0 Warnings: 0

CREATE TABLE AND INSERT VALUES:

create table Tour_Locations(

tour_id   int,

location_id   int,

FOREIGN KEY(tour_id) references Tour(tour_id),

FOREIGN KEY(location_id) references Locations(location_id)

);

insert into Tour_Locations(tour_id, location_id)

values(1,5),

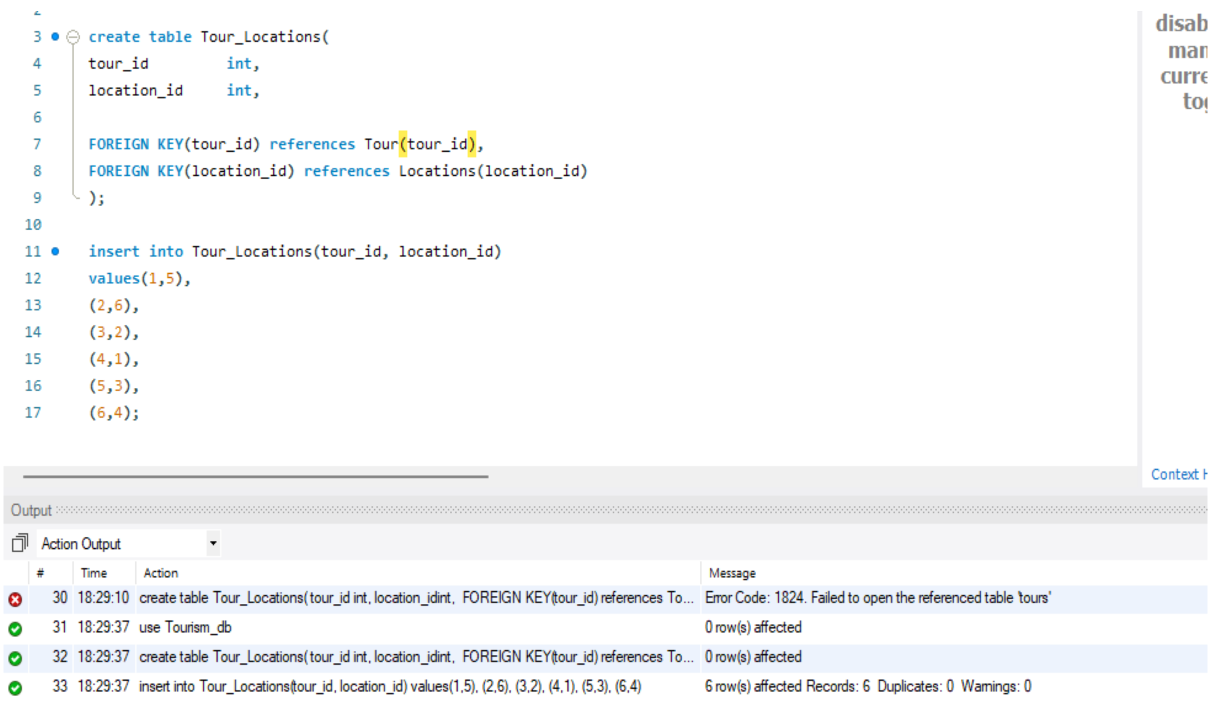
(2,6),

(3,2),

(4,1),

(5,3),

(6,4);



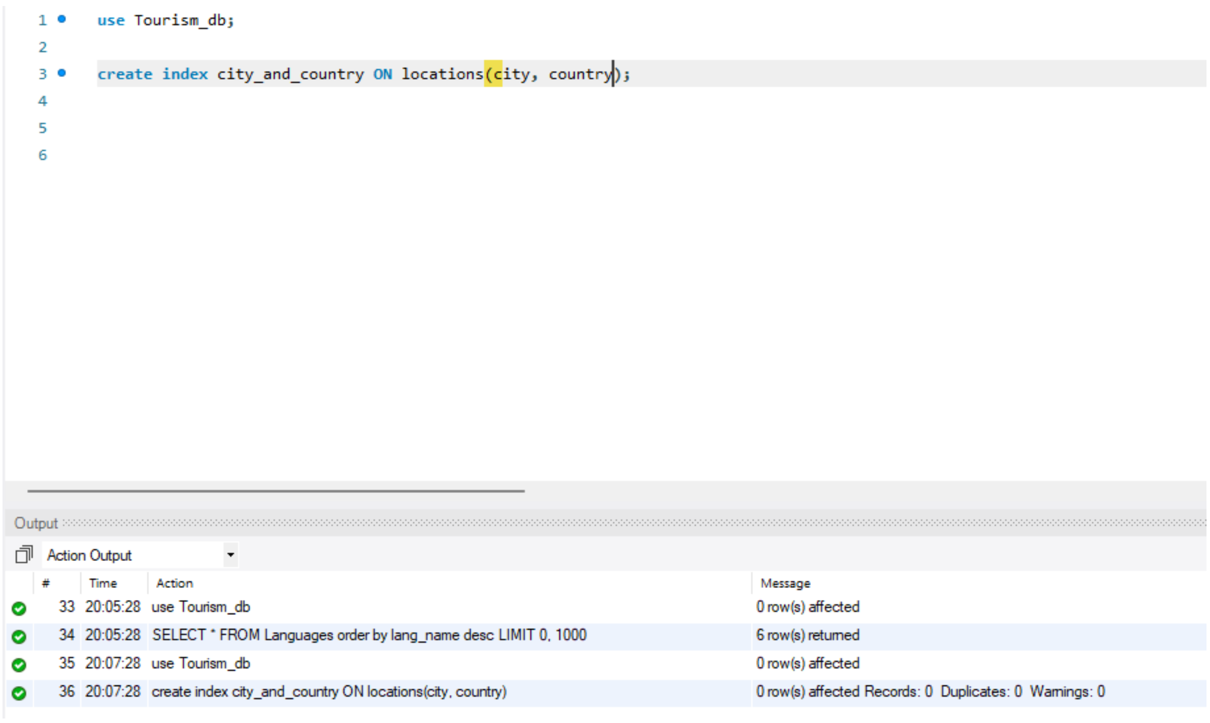
```
create table Tour_Locations(
tour_id         int,
location_id     int,

FOREIGN KEY(tour_id) references Tour(tour_id),
FOREIGN KEY(location_id) references Locations(location_id)
);

insert into Tour_Locations(tour_id, location_id)
values(1,5),
(2,6),
(3,2),
(4,1),
(5,3),
(6,4);
```

```
Output
Action Output
#   Time      Action                                                                              Message
30  18:29:10  create table Tour_Locations( tour_id int, location_idint,  FOREIGN KEY(tour_id) references To...  Error Code: 1824. Failed to open the referenced table 'tours'
31  18:29:37  use Tourism_db                                                                      0 row(s) affected
32  18:29:37  create table Tour_Locations( tour_id int, location_idint,  FOREIGN KEY(tour_id) references To...  0 row(s) affected
33  18:29:37  insert into Tour_Locations(tour_id, location_id) values(1,5), (2,6), (3,2), (4,1), (5,3), (6,4)  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
```

B. Index Strategy

Locations.city IDX, Locations.country IDX

Tours.guide_id IDX



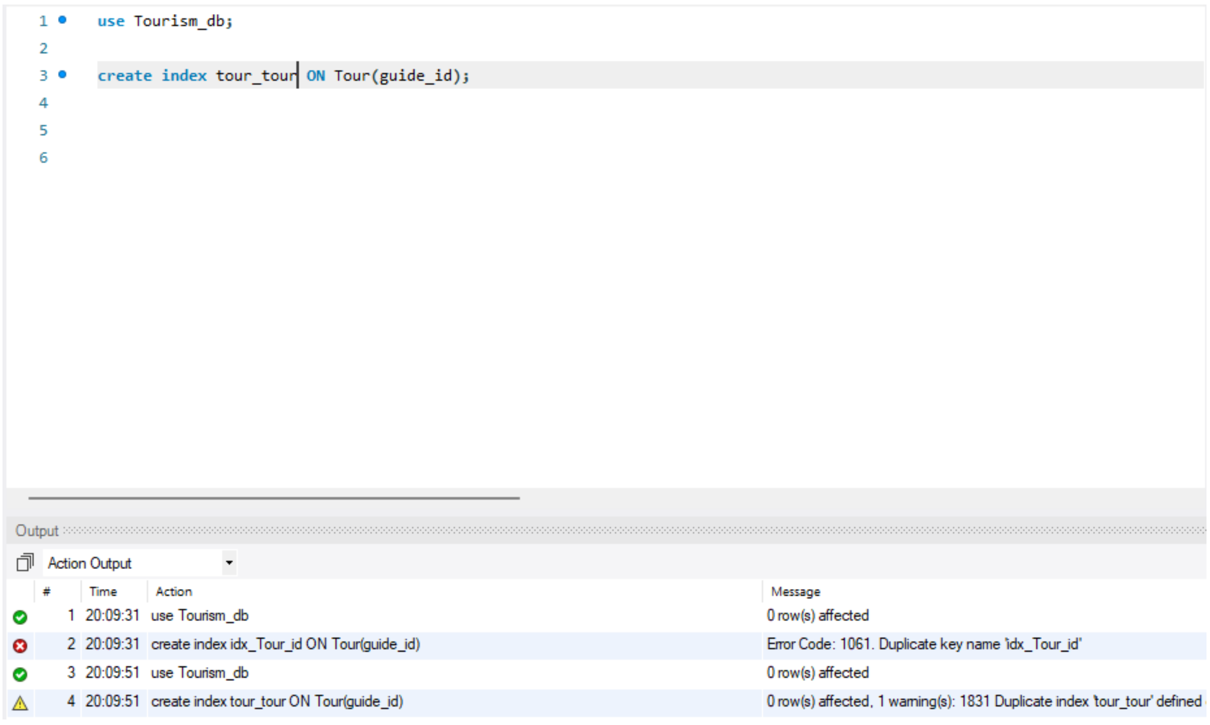
```
use Tourism_db;

create index city_and_country ON locations(city, country);
```

Output

Action Output

```
#   Time      Action                                                     Message
33  20:05:28  use Tourism_db                                             0 row(s) affected
34  20:05:28  SELECT * FROM Languages order by lang_name desc LIMIT 0, 1000   6 row(s) returned
35  20:07:28  use Tourism_db                                             0 row(s) affected
36  20:07:28  create index city_and_country ON locations(city, country)  0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0
```

NOTE: THIS IS NOT ERROR I HAVE PRESS BUTTON TWO TIMES SO THAT'S WHY HE IS SHOWING DUPLICATE ERROR



```
use Tourism_db;

create index tour_tour ON Tour(guide_id);
```

Output

Action Output

```
#  Time      Action                                        Message
1  20:09:31  use Tourism_db                                0 row(s) affected
2  20:09:31  create index idx_Tour_id ON Tour(guide_id)    Error Code: 1061. Duplicate key name 'idx_Tour_id'
3  20:09:51  use Tourism_db                                0 row(s) affected
4  20:09:51  create index tour_tour ON Tour(guide_id)      0 row(s) affected, 1 warning(s): 1831 Duplicate index 'tour_tour' defined
```

C. Views for Front-End

v_PublicTourDetails

Join Tours + Guides + Tour_Locations + Locations

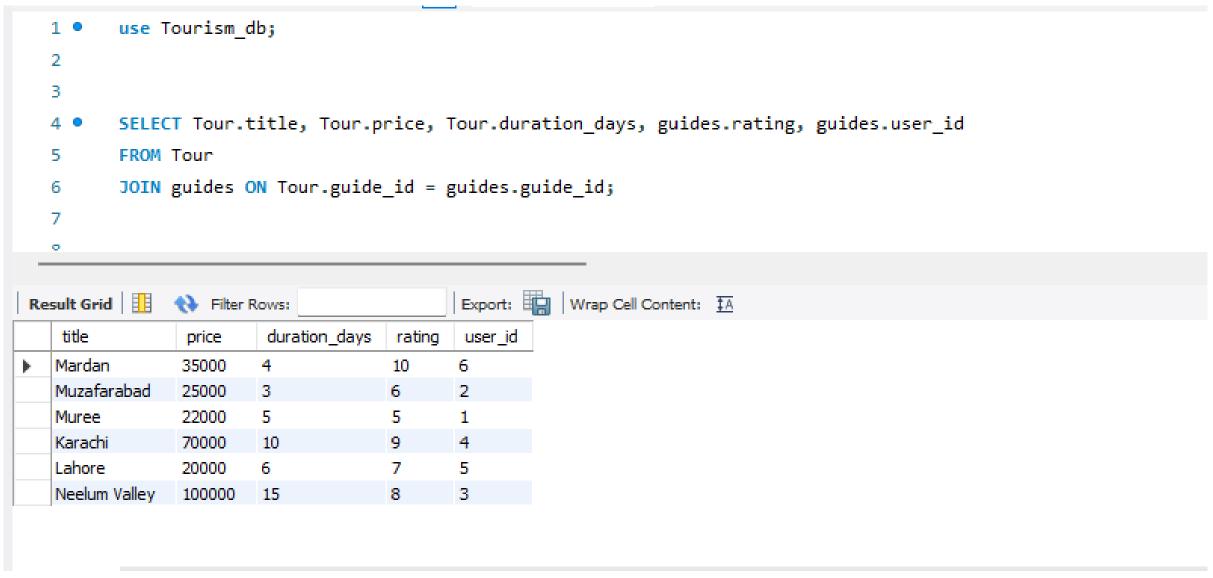
Select only the columns the app needs (hides internal IDs, timestamps)

QUERY:

SELECT Tour.title, Tour.price, Tour.duration_days, guides.rating, guides.user_id

FROM Tour

JOIN guides ON Tour.guide_id = guides.guide_id;



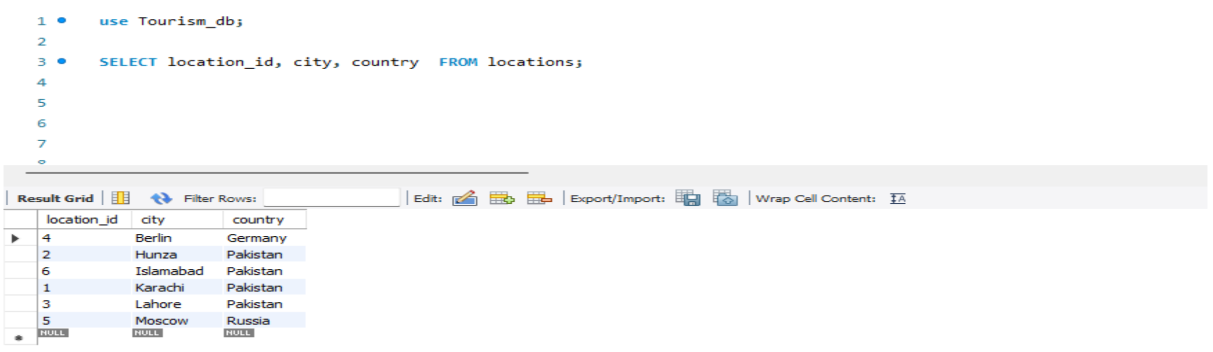
```
use Tourism_db;


SELECT Tour.title, Tour.price, Tour.duration_days, guides.rating, guides.user_id
FROM Tour
JOIN guides ON Tour.guide_id = guides.guide_id;
```

title	price	duration_days	rating	user_id
Mardan	35000	4	10	6
Muzafarabad	25000	3	6	2
Muree	22000	5	5	1
Karachi	70000	10	9	4
Lahore	20000	6	7	5
Neelum Valley	100000	15	8	3

D. Sample Data & Verification

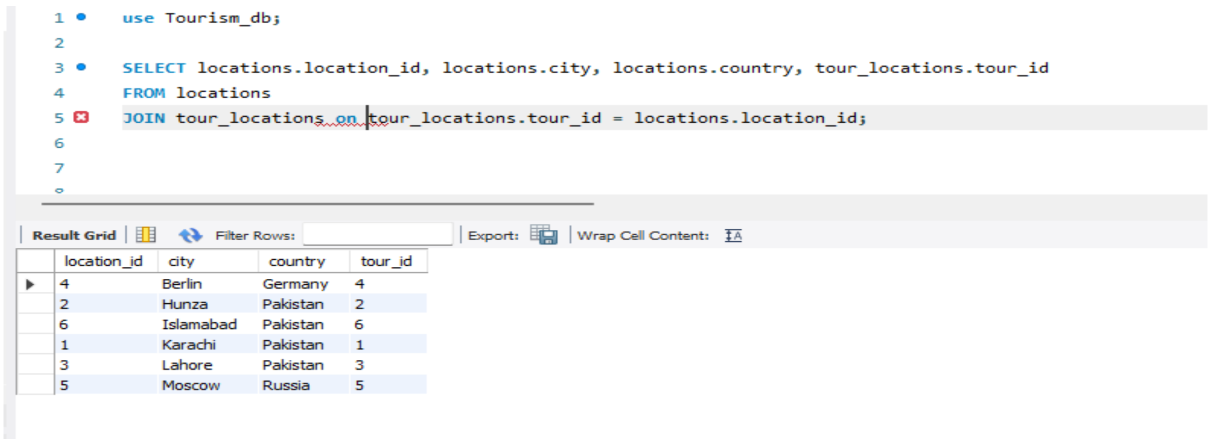
Populate each table with sample tours and locations

Validate the view by running SELECT (e.g. list all tours in “PAKISTAN”)



```
use Tourism_db;

SELECT location_id, city, country  FROM locations;
```

location_id	city	country
4	Berlin	Germany
2	Hunza	Pakistan
6	Islamabad	Pakistan
1	Karachi	Pakistan
3	Lahore	Pakistan
5	Moscow	Russia
NULL	NULL	NULL



```
use Tourism_db;

SELECT locations.location_id, locations.city, locations.country, tour_locations.tour_id
FROM locations
JOIN tour_locations on tour_locations.tour_id = locations.location_id;
```

location_id	city	country	tour_id
4	Berlin	Germany	4
2	Hunza	Pakistan	2
6	Islamabad	Pakistan	6
1	Karachi	Pakistan	1
3	Lahore	Pakistan	3
5	Moscow	Russia	5

@Faheem : Booking & Feedback System

A. Table & Column Design

Bookings

Columns: booking_id, user_id FK → Users, tour_id FK → Tours, booking_date, num_people, total_price, status

PK: booking_id

CREATE TABLE AND INSERT VALUES:

create table Bookings(

booking_id	int(11) PRIMARY KEY,
user_id	int,
tour_id	int,
booking_date	date,
num_people	int(11),
price	int(11),
status	varchar(255),

FOREIGN KEY(user_id) references Users(user_id),

FOREIGN KEY(tour_id) references Tour(tour_id)

);

insert into Bookings(booking_id, user_id, tour_id, booking_date, num_people, price, status)

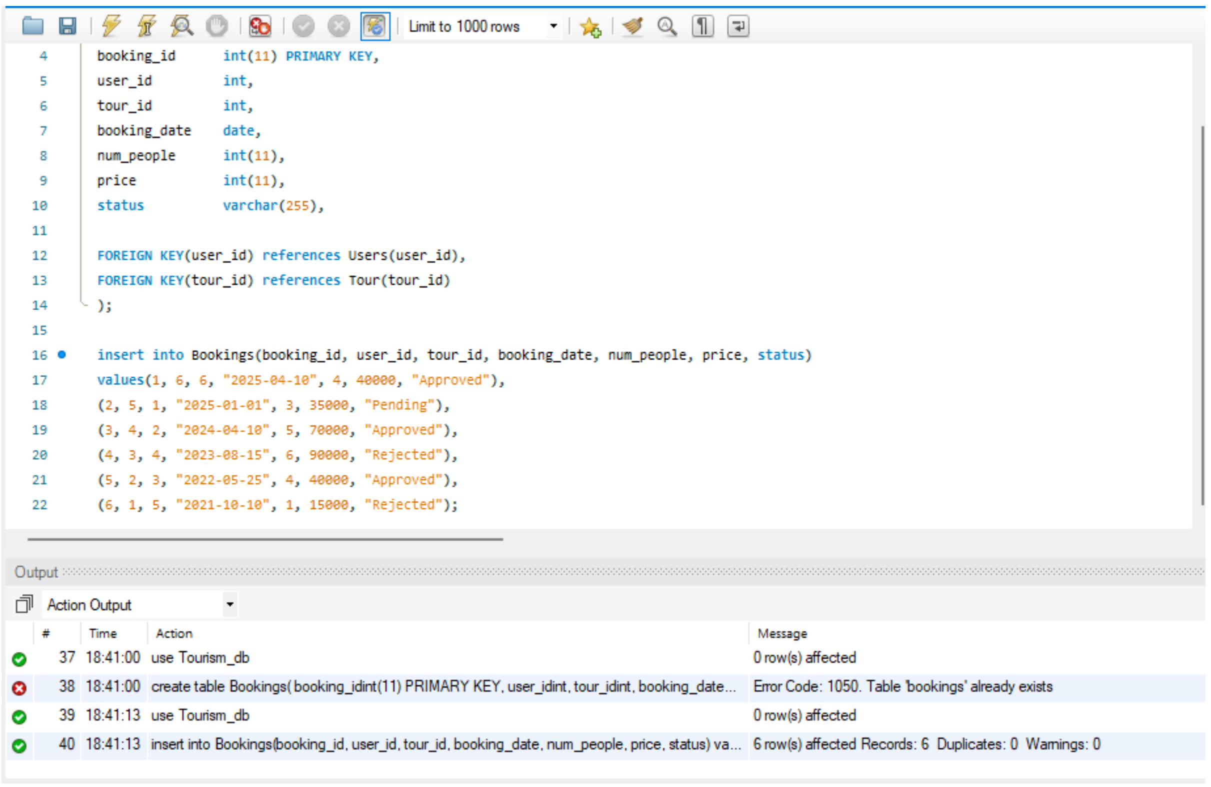
values(1, 6, 6, "2025-04-10", 4, 40000, "Approved"), (2, 5, 1, "2025-01-01", 3, 35000, "Pending"),

(3, 4, 2, "2024-04-10", 5, 70000, "Approved"),

(4, 3, 4, "2023-08-15", 6, 90000, "Rejected"),

(5, 2, 3, "2022-05-25", 4, 40000, "Approved"),

(6, 1, 5, "2021-10-10", 1, 15000, "Rejected");



```
booking_id      int(11) PRIMARY KEY,
user_id         int,
tour_id         int,
booking_date    date,
num_people      int(11),
price           int(11),
status          varchar(255),

FOREIGN KEY(user_id) references Users(user_id),
FOREIGN KEY(tour_id) references Tour(tour_id)
);

insert into Bookings(booking_id, user_id, tour_id, booking_date, num_people, price, status)
values(1, 6, 6, "2025-04-10", 4, 40000, "Approved"),
(2, 5, 1, "2025-01-01", 3, 35000, "Pending"),
(3, 4, 2, "2024-04-10", 5, 70000, "Approved"),
(4, 3, 4, "2023-08-15", 6, 90000, "Rejected"),
(5, 2, 3, "2022-05-25", 4, 40000, "Approved"),
(6, 1, 5, "2021-10-10", 1, 15000, "Rejected");
```

```
Output
Action Output
    #   Time      Action                                                                              Message
    37  18:41:00  use Tourism_db                                                                      0 row(s) affected
    38  18:41:00  create table Bookings( booking_idint(11) PRIMARY KEY, user_idint, tour_idint, booking_date...   Error Code: 1050. Table 'bookings' already exists
    39  18:41:13  use Tourism_db                                                                      0 row(s) affected
    40  18:41:13  insert into Bookings(booking_id, user_id, tour_id, booking_date, num_people, price, status) va...   6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0
```

CREATE TABLE AND INSERT VALUES:

create table Reviews(

review_id		int(11) PRIMARY KEY,

booking_id		int,

rating			int(11),

comment			varchar(255),

create_at		date,

FOREIGN KEY(booking_id) references Bookings(booking_id)

);

insert into Reviews(review_id, booking_id, rating, comment, create_at)

values(1,6, 8, "Nice Servies", "2025-01-15"),

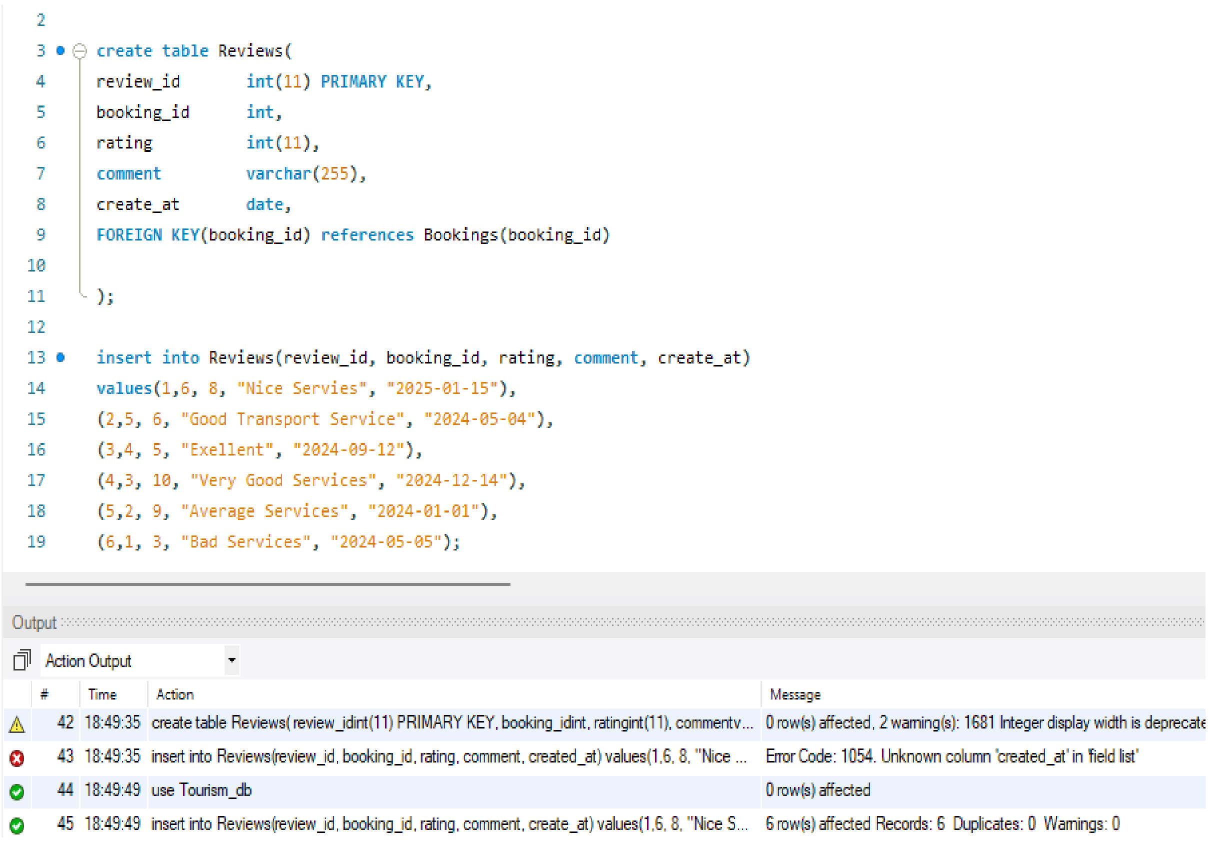
(2,5, 6, "Good Transport Service", "2024-05-04"),

(3,4, 5, "Exellent", "2024-09-12"),

(4,3, 10, "Very Good Services", "2024-12-14"),

(5,2, 9, "Average Services", "2024-01-01"),

(6,1, 3, "Bad Services", "2024-05-05");



```sql

create table Reviews(
review_id       int(11) PRIMARY KEY,
booking_id      int,
rating          int(11),
comment         varchar(255),
create_at       date,
FOREIGN KEY(booking_id) references Bookings(booking_id)

);

insert into Reviews(review_id, booking_id, rating, comment, create_at)
values(1,6, 8, "Nice Servies", "2025-01-15"),
(2,5, 6, "Good Transport Service", "2024-05-04"),
(3,4, 5, "Exellent", "2024-09-12"),
(4,3, 10, "Very Good Services", "2024-12-14"),
(5,2, 9, "Average Services", "2024-01-01"),
(6,1, 3, "Bad Services", "2024-05-05");
```

Output

Action Output

	#	Time	Action	Message
⚠	42	18:49:35	create table Reviews(review_idint(11) PRIMARY KEY, booking_idint, ratingint(11), commentv...	0 row(s) affected, 2 warning(s): 1681 Integer display width is deprecate
✗	43	18:49:35	insert into Reviews(review_id, booking_id, rating, comment, created_at) values(1,6, 8, "Nice ...	Error Code: 1054. Unknown column 'created_at' in 'field list'
✓	44	18:49:49	use Tourism_db	0 row(s) affected
✓	45	18:49:49	insert into Reviews(review_id, booking_id, rating, comment, create_at) values(1,6, 8, "Nice S...	6 row(s) affected Records: 6 Duplicates: 0 Warnings: 0

B. Index Strategy

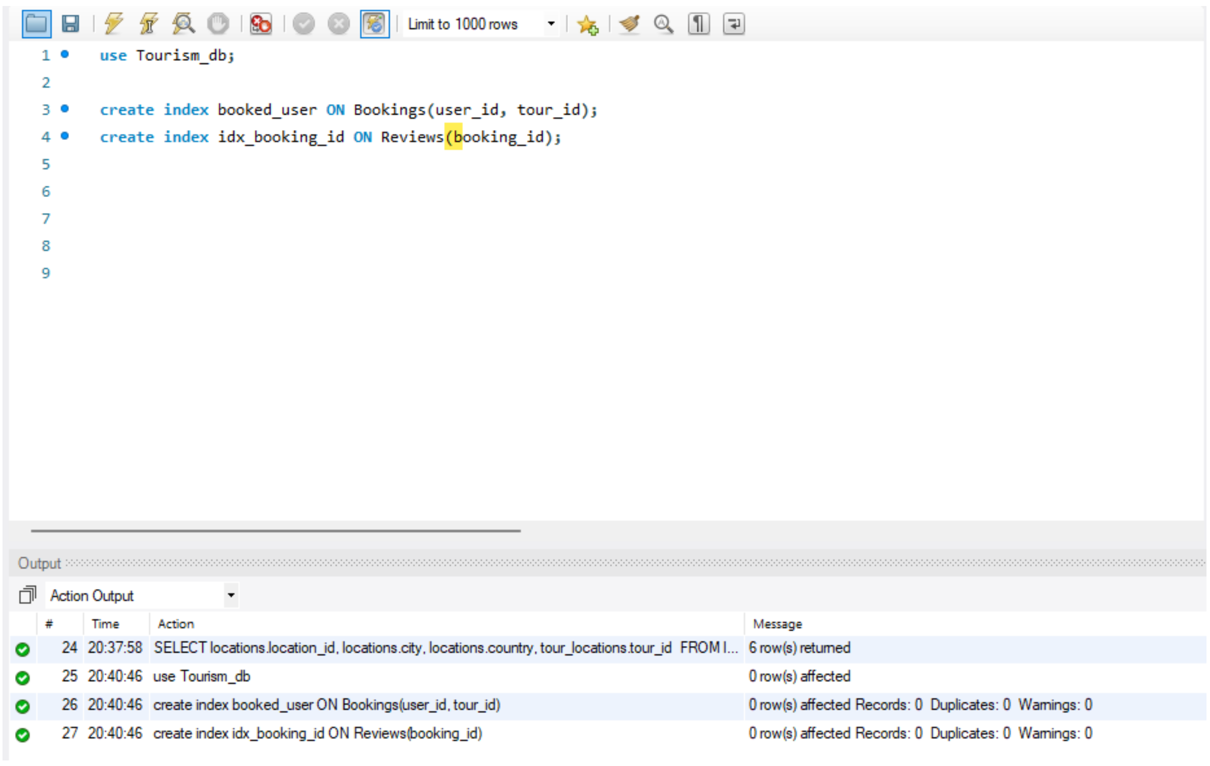
Bookings.user_id IDX, Bookings.tour_id IDX

Reviews.booking_id IDX

QUERY:

create index booked_user ON Bookings(user_id, tour_id);

create index idx_booking_id ON Reviews(booking_id);



```
use Tourism_db;

create index booked_user ON Bookings(user_id, tour_id);
create index idx_booking_id ON Reviews(booking_id);
```

```
Output
Action Output
    #   Time      Action                                                                              Message
    24  20:37:58  SELECT locations.location_id, locations.city, locations.country, tour_locations.tour_id  FROM l...  6 row(s) returned
    25  20:40:46  use Tourism_db                                                                      0 row(s) affected
    26  20:40:46  create index booked_user ON Bookings(user_id, tour_id)                              0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0
    27  20:40:46  create index idx_booking_id ON Reviews(booking_id)                                  0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0
```

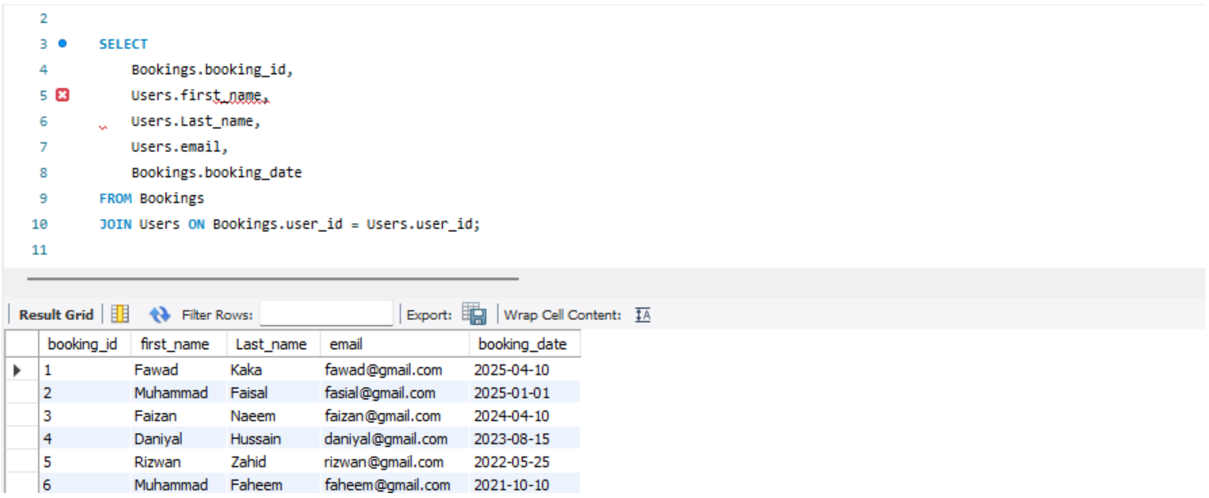
C. Sample Data & Verification

Load a handful of bookings and reviews

Run queries to ensure you can join “Booking → User” and “Booking → Tour”

QUERY:

```
SELECT
   Bookings.booking_id,
   Users.first_name,
   Users.Last_name,
   Users.email,
   Bookings.booking_date
FROM Bookings
JOIN Users ON Bookings.user_id = Users.user_id;
```



```
SELECT
    Bookings.booking_id,
    Users.first_name,
    Users.Last_name,
    Users.email,
    Bookings.booking_date
FROM Bookings
JOIN Users ON Bookings.user_id = Users.user_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

booking_id	first_name	Last_name	email	booking_date
1	Fawad	Kaka	fawad@gmail.com	2025-04-10
2	Muhammad	Faisal	fasial@gmail.com	2025-01-01
3	Faizan	Naeem	faizan@gmail.com	2024-04-10
4	Daniyal	Hussain	daniyal@gmail.com	2023-08-15
5	Rizwan	Zahid	rizwan@gmail.com	2022-05-25
6	Muhammad	Faheem	faheem@gmail.com	2021-10-10

B. Roles & Privileges

Define Roles

read_only_role

app_user_role

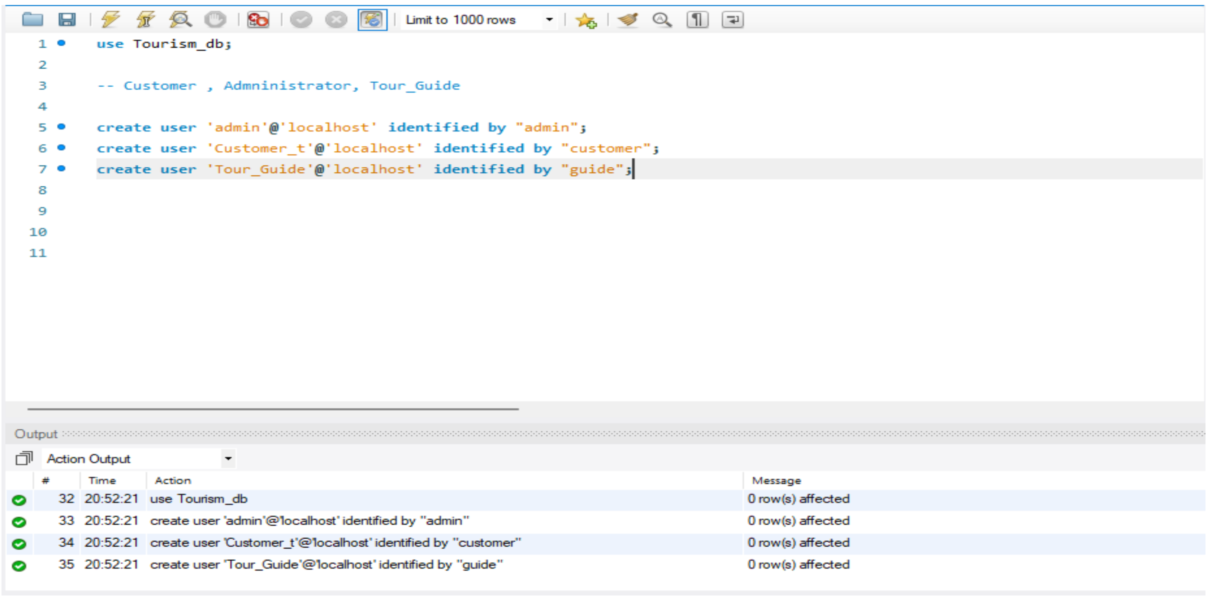
content_manager_role

QUERIES:

create user 'admin'@'localhost' identified by "admin";

create user 'Customer_t'@'localhost' identified by "customer";

create user 'Tour_Guide'@'localhost' identified by "guide";



```
use Tourism_db;

-- Customer , Administrator, Tour_Guide

create user 'admin'@'localhost' identified by "admin";
create user 'Customer_t'@'localhost' identified by "customer";
create user 'Tour_Guide'@'localhost' identified by "guide";
```

```
#   Time      Action                                                        Message
32  20:52:21  use Tourism_db                                                0 row(s) affected
33  20:52:21  create user 'admin'@'localhost' identified by "admin"         0 row(s) affected
34  20:52:21  create user 'Customer_t'@'localhost' identified by "customer" 0 row(s) affected
35  20:52:21  create user 'Tour_Guide'@'localhost' identified by "guide"    0 row(s) affected
```

Grant Permissions

read_only_role: SELECT on views (e.g. v_PublicTourDetails)

user_role:

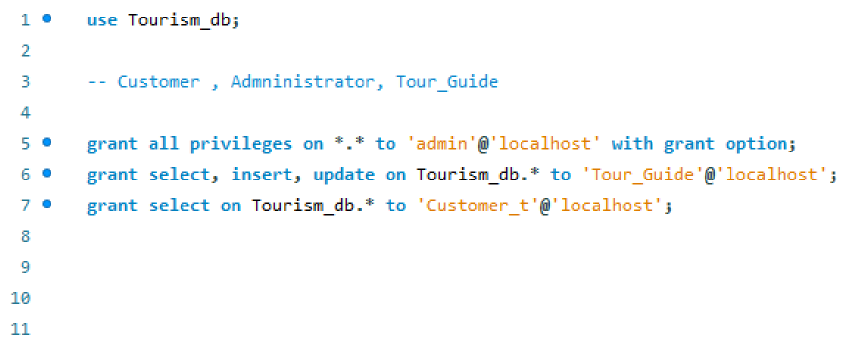
admin role:

QUERIES:

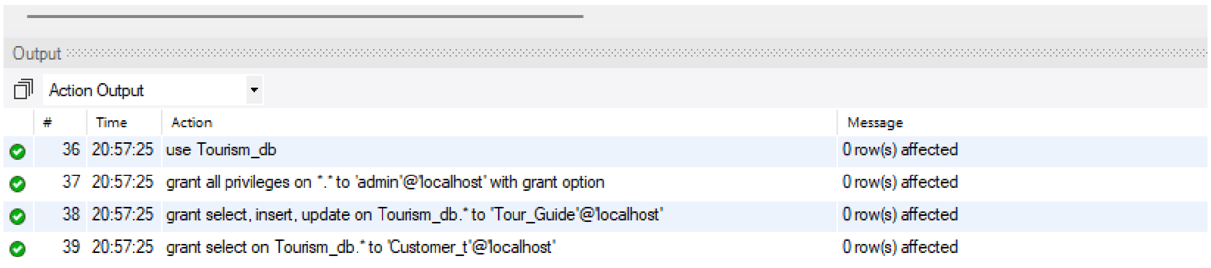
grant all privileges on *.* to 'admin'@'localhost' with grant option;

grant select, insert, update on Tourism_db.* to 'Tour_Guide'@'localhost';

grant select on Tourism_db.* to 'Customer_t'@'localhost';



```
use Tourism_db;

-- Customer , Admninistrator, Tour_Guide

grant all privileges on *.* to 'admin'@'localhost' with grant option;
grant select, insert, update on Tourism_db.* to 'Tour_Guide'@'localhost';
grant select on Tourism_db.* to 'Customer_t'@'localhost';
```



```
Output
Action Output
  #   Time      Action                                                              Message
  36  20:57:25  use Tourism_db                                                      0 row(s) affected
  37  20:57:25  grant all privileges on *.* to 'admin'@'localhost' with grant option  0 row(s) affected
  38  20:57:25  grant select, insert, update on Tourism_db.* to 'Tour_Guide'@'localhost'  0 row(s) affected
  39  20:57:25  grant select on Tourism_db.* to 'Customer_t'@'localhost'           0 row(s) affected
```

Conclusion

The Tourism Guide Database successfully fulfills the requirements of managing a tour-based application system. It captures user, guide, tour, location, and booking information efficiently. It ensures referential integrity, supports multilingual operations, and is secure through MySQL user roles. The project provides a robust backend that could be integrated with a web or mobile frontend for real-world use. The use of views, indexing, and normalization makes it scalable and performance-efficient.

Future Enhancements

- **Frontend Integration:** Develop a web-based UI or mobile app using this backend.
- **Payment Integration:** Add online payment features in the bookings table.
- **Advanced Filters:** Create advanced views for filtering tours by price, location, and guide rating.
- **Notification System:** Integrate email/SMS notifications for bookings and reviews.
- **Data Analytics:** Add reports on most visited places, guide performance, and customer feedback analysis.

References

- MySQL Documentation – https://dev.mysql.com/doc/
- W3Schools SQL Tutorial – https://www.w3schools.com/sql/
- Course Notes on DBMS