Travel Advisor Database Management System

Group 12

Student1 : Anjali Ingle
Student2: Ameya Deshmukh

(617)-372-0345 (Tel of Student 1)
857-396-7408 (Tel of Student 2)

ingle.a@northeastern.edu
deshmukh.amey@notheastern.edu

Percentage of Effort Contributed by Student1:____50____
Percentage of Effort Contributed by Student2:____50____

Signature of Student 1:_____Anjali_____
Signature of Student 2:_____Ameya_____

Submission Date:_____ 12/10/23 _____

Travel Booking Management System Report

Executive Summary:

The Travel Advisory Management System is a one-stop platform for travel bookings. It's designed to make trip planning easier by storing details about cities, hotels,and transportation options, including prices. This system helps users find destinations, compare travel choices like flights or trains, and pick hotels. For administrators, it offers insights like popular destinations, favored travel methods, and average customer spending.

We meticulously designed our travel advisory system by initiating the creation of a comprehensive data model. This involved crafting Entity-Relationship (EER) and Unified Modeling Language (UML) diagrams, meticulously outlining the structure and relationships within our data.

Subsequently, we translated our conceptual designs into a practical implementation by creating a robust relational data model. This model served as the backbone for our MySQL database, facilitating the organization and management of our travel-related information.

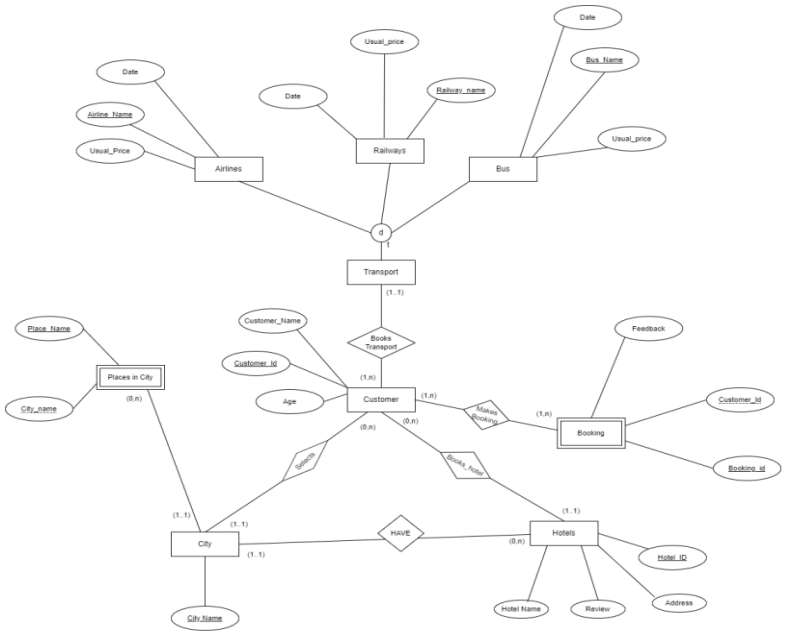
Utilizing this structured foundation, we seamlessly established connectivity between our MySQL data and MongoDB using Python scripts. Leveraging MongoDB Compass, we effortlessly translated our relational model into MongoDB collections.

Furthermore, our system was elevated through the integration of a user-friendly front-end developed in Streamlit. Incorporating insightful

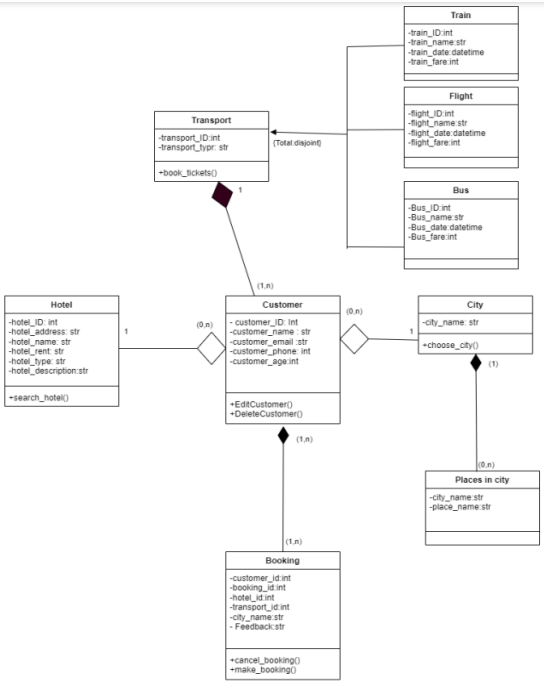
visualizations such as graphs, the interface showcased business analytics derived from our meticulously linked SQL and NoSQL databases.

In essence, our travel advisory platform embodies meticulous planning, conceptualization through EER and UML diagrams, structured data management via relational models, seamless integration with MongoDB, and a user-centric interface through Streamlit, culminating in an insightful and user-friendly travel advisory system.

Conceptual Data Modelling :



EER DIAGRAM



UML DIAGRAM

Relational Model:

Customer(Customer_ID(PK),Customer_Name,Age,City_NAME(FK),Hotel_ID,Transport_ID)

Booking(Customer_ID(PK),Booking_id(PK),Feedback)

Hotel(Hotel_ID,Address,Review,Hotel_Name,City_Name(FK))

City(City_Name)

Places in city(Place_Name(PK),City_name(PK))

Transport (Trasport_ID(PK),Transport_Type(PK))

Railways (Transport_ID(FK),Railway_Name (PK),Date,Usual_price)

Airlines (Transport_ID(FK),Airline_Name (PK),Date,Usual_price)

Bus (Transport_ID(FK),Bus_Name (PK),Date,Usual_price)

Hotels (Hotel_ID,Address,Review,Hotel_Name)

Makes_Booking(Customer_ID(FK),Booking_id(FK))

MYSQL Implementation :

1) Total bookings made by each customer

select c.customer_name,

t.count as 'Booked_using platform'

from customers c

join (select customer_id,count(*) as count

from booking_data group by Customer_ID)ton c.customer_id=t.customer_id

customer_name	Booked_using platform
John Doe	1
Jane Smith	3
Alice Johnson	3
Bob Williams	3
Eva Brown	1
Michael Clark	1
Sophia Lee	5
David Taylor	3
Olivia White	2
William Harris	1
Ava Martin	3
James Anderson	5
Daniel Martinez	2
Mia Lopez	5
Alexander King	1
Grace Hall	2
Liam Young	3
Isabella Lewis	3
Noah Adams	3

2) Find Hotels With a High Number of Bookings for every City

select SELECT h.City_Name, h.Hotel_ID, h.Hotel_Name,

COUNT(b.Hotel_ID) AS num_bookingsFROM hotels h JOIN

booking_data b ON h.Hotel_ID = b.Hotel_ID

GROUP BY h.City_Name, h.Hotel_ID, h.Hotel_Name

ORDER BY h.City_Name, num_bookings DESC;

City_Name	Hotel_ID	Hotel_Name	num_bookings
Ahmedabad	31	Ahmedabad Hotel 1	6
Bengaluru	11	Taj West End	7
Chennai	16	Taj Coromandel	5
Hyderabad	26	Hyderabad Hotel 1	7
Kolkata	21	The Oberoi Grand	11
Mumbai	1	Taj Mahal Palace	6
New Delhi	6	The Imperial	8

3) Feedback given by customers for hotels

SELECT C.Customer_ID, B.Feedback, B.Booking_ID,

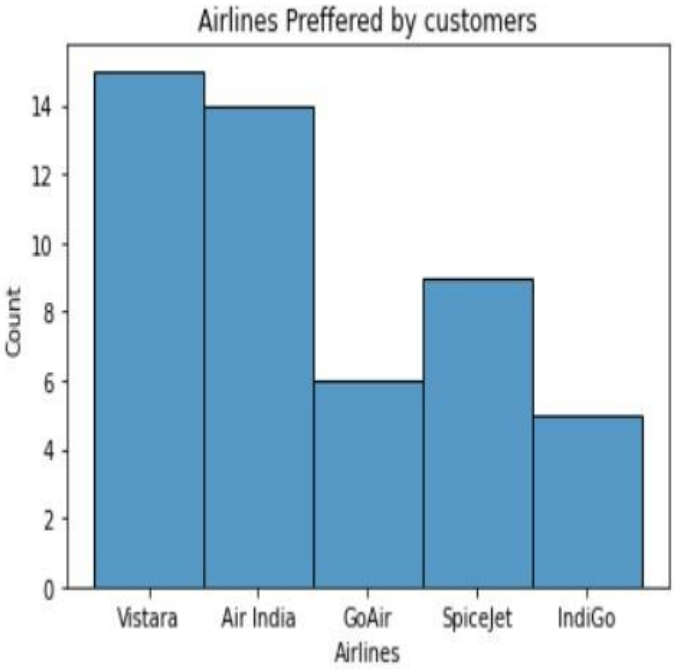
B.Hotel_ID, H.Hotel_NameFROM Customers C

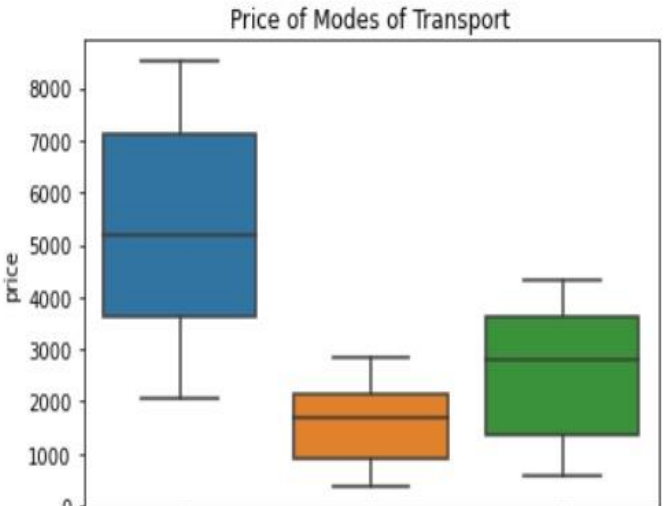
JOIN booking_data B ON C.Customer_ID = B.Customer_ID

JOIN hotels H ON B.Hotel_ID = H.Hotel_IDWHERE B.Feedback IS NOT NULL;

Customer_ID	Feedback	Booking_ID	Hotel_ID	Hotel_Name
3	Good	42	1	Taj Mahal Palace
12	Excellent	40	1	Taj Mahal Palace
2	Excellent	36	1	Taj Mahal Palace
4	Excellent	35	1	Taj Mahal Palace
8	Good	14	1	Taj Mahal Palace
2	Average	10	1	Taj Mahal Palace
17	Average	49	6	The Imperial
18	Good	45	6	The Imperial
19	Good	27	6	The Imperial
10	Good	23	6	The Imperial
11	Excellent	17	6	The Imperial
18	Good	15	6	The Imperial
5	Good	8	6	The Imperial
11	Excellent	4	6	The Imperial
19	Good	34	11	Taj West End
15	Excellent	33	11	Taj West End
3	Good	32	11	Taj West End
16	Good	24	11	Taj West End
3	Excellent	20	11	Taj West End
15	Good	12	11	Taj West End
12	Good	6	11	Taj West End
4	Excellent	50	16	Taj Coromandel
4	Average	44	16	Taj Coromandel
12	Average	39	16	Taj Coromandel

<u>Streamlite Application Implementation:</u>





NOSQL Implementation :

We connected our MYSQL database with MONGODB using Python.

jupyter mysql_nosql_conn Last Checkpoint: a day ago (autosaved)

Logout

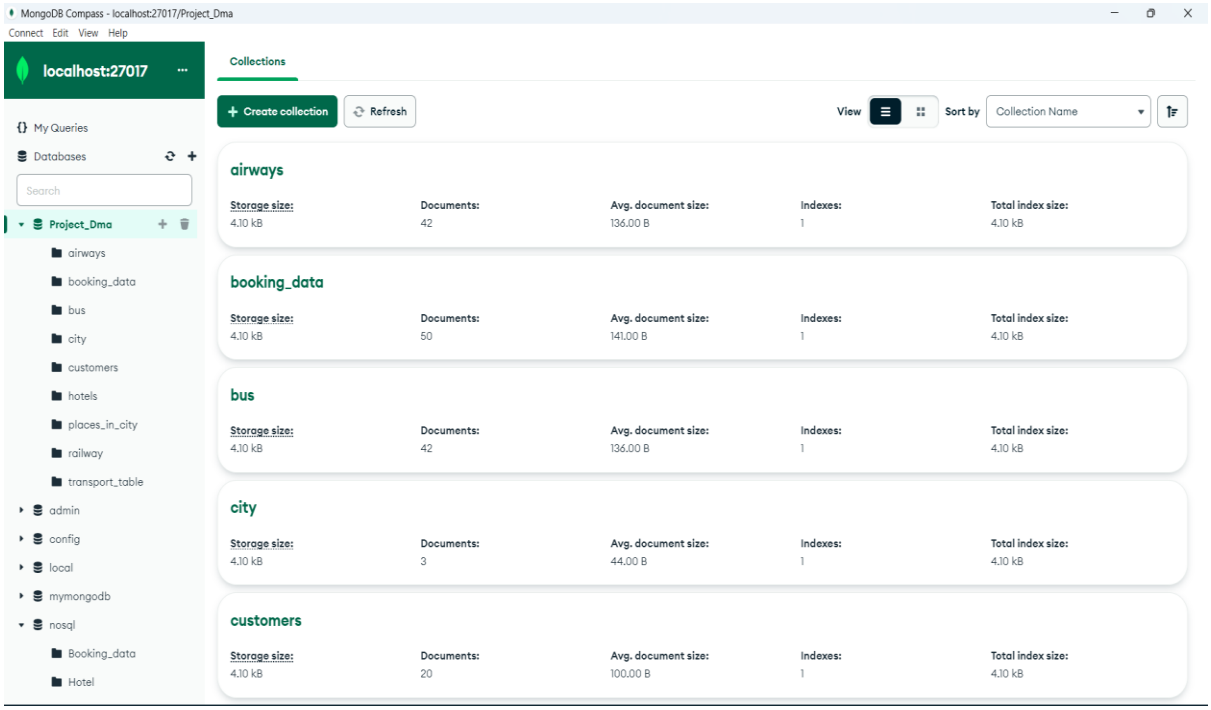
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

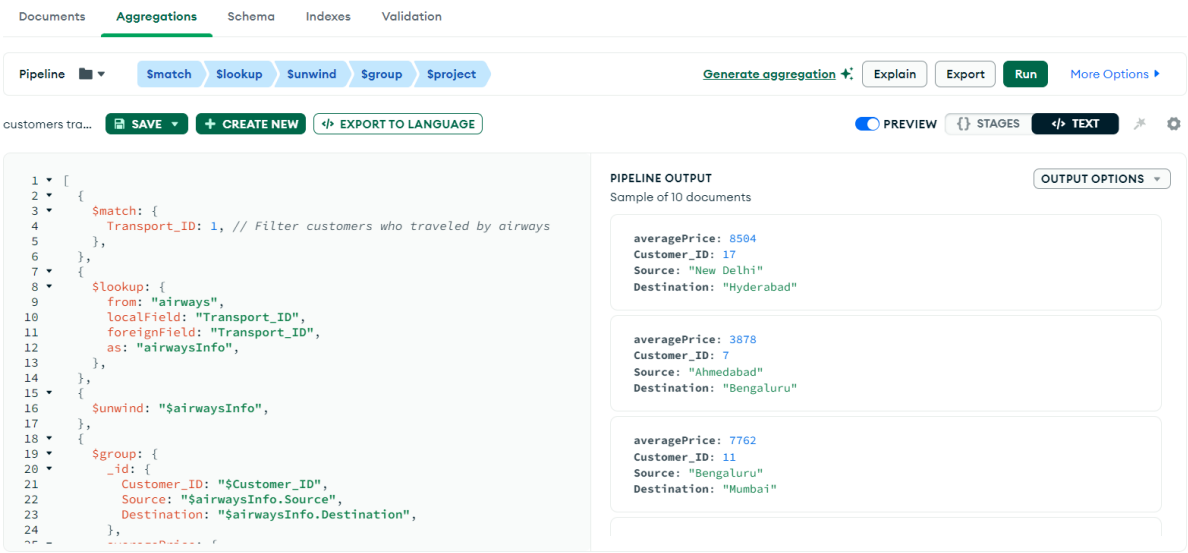
In [13]:

```python
import pymysql
from pymongo import MongoClient

# MySQL database configuration
mysql_config = {
    'host': 'localhost',
    'user': 'root',
    'password': '1234',
    'db': 'newschema',
}

# MongoDB Compass configuration
mongodb_host = "mongodb://localhost:27017/"
mongodb_dbname = "Project_Dma"  # Set your MongoDB database name here

def connect_to_mysql():
    return pymysql.connect(**mysql_config)

def connect_to_mongodb():
    myclient = MongoClient(mongodb_host)
    return myclient[mongodb_dbname]

def create_collection_if_not_exists(mongo_db, collection_name):
    if collection_name not in mongo_db.list_collection_names():
        mongo_db.create_collection(collection_name)

def fetch_tables_from_mysql(connection):
    tables = []
    try:
        with connection.cursor() as cursor:
            # Get the list of tables in the "newschema" database
            cursor.execute("SHOW TABLES FROM newschema")
            tables = [table[0] for table in cursor.fetchall()]
    except Exception as e:
        print(f"Error fetching tables from MySQL: {e}")
    return tables
```

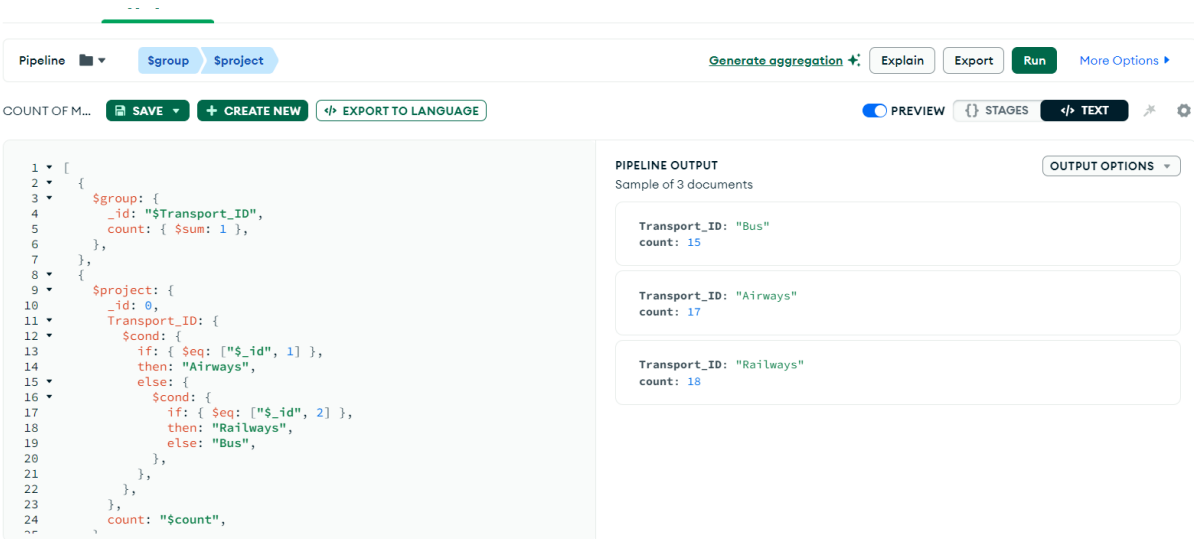
We created collections in MongoDB which stored all our data.



Customer travelling by airways.



Count of modes of transportation preferred by customers.



Summary & Recommendation :

"Our Travel Advisory Management System underwent extensive development, actively enhancing the transition to MongoDB with clear visual representations. Notably, we prioritized database cleanliness, refining MongoDB collections for streamlined, efficient NoSQL implementation.

In tandem, we proactively refined EER and UML diagrams, offering comprehensive insights into data structures. Our active efforts concentrated on improving data consistency between MySQL and MongoDB, bolstering reliability.

To further elevate the system, a focus on continuous improvements in clean database practices and optimal NoSQL implementations remains a primary area for ongoing enhancements. These active refinements ensure a more efficient and user-centric platform for travelers and administrators alike."