Schema of database

▼ tables

Flight Table

Column Name	Data Type (MySQL)	Description
`travelCode`	INT (Primary Key)	Unique identifier for the trip (linked with flights/hotels if applicable).
`User_ID`	INT (Foreign Key)	Unique user identifier (linked to users.csv).
`Departure`	VARCHAR(255)	Departure location of the flight.
`Arrival`	VARCHAR(255)	Destination location of the flight.
`flightType`	ENUM('Domestic', 'International')	Type of flight.
`Flight_price`	DECIMAL(10,2)	Cost of the flight.
`Flight_duration`	DECIMAL(5,2)	Duration of the flight in hours.
`Flight_Distance`	DECIMAL(10,2)	Distance covered by the flight in kilometers.
`Flight_agency`	VARCHAR(255)	Airline company providing the flight.
`Departure_date`	DATETIME	Date and time of departure.

Hotel Table

Column Name	Data Type (MySQL)	Description
`User_ID`	INT (Foreign Key)	Unique user identifier (linked to users.csv).
`travelCode`	INT (Primary Key)	Unique identifier for the trip (linked with flights/hotels if applicable).
`Hotel_Name`	VARCHAR(255)	Name of the hotel.
`Arrival_place`	VARCHAR(255)	City or place where the hotel is located.
`Hotel_stay`	INT	Number of nights stayed.
`Hotel_per_day_price`	DECIMAL(10,2)	Cost per night of stay.
`Check-in`	DATETIME	Date and time of check-in.
`Hotel_TotalPrice`	DECIMAL(10,2)	Total cost of the stay.

Car Rental Table

Column Name	Data Type (MySQL)	Description
User_ID	INT (Foreign Key)	Unique user identifier (linked to users.csv).
travelCode	INT (Primary Key)	Unique identifier for the trip (linked with flights/hotels if applicable).
Check-in	DATETIME	Date and time when the car rental starts.
pickupLocation	VARCHAR(255)	Location where the car is picked up.
dropoffLocation	VARCHAR(255)	Location where the car is returned.
carType	ENUM('Sedan', 'SUV', 'Hatchback', 'Luxury')	Type of car rented.
rentalAgency	VARCHAR(255)	Name of the car rental provider.
rentalDuration	INT	Total rental duration in days or hours.
Car_total_distance	INT	Total distance allowed/traveled.
fuelPolicy	ENUM('Full-to-Full', 'Prepaid')	Fuel policy for the rental.
Car_bookingStatus	ENUM('Confirmed', 'Cancelled', 'Pending')	Status of the booking.
total_rent_price	DECIMAL(10,2)	Cost of renting the car.

Passenger Table

Column Name	Data Type (MySQL)	Description
User_ID	INT (Foreign Key)	Unique user identifier (linked to users.csv).
company	VARCHAR(255)	Company associated with the passenger.
Name	VARCHAR(255)	Passenger's full name.
gender_x	ENUM('Male', 'Female', 'Other')	Passenger's gender.

Customer Call Table

Column Name	Data Type (MySQL)	Description
User_ID	INT (Foreign Key)	Unique user identifier (linked to users.csv).
Arrival_date	DATETIME	Date of arrival.

`issueType`	VARCHAR(255)	Type of issue reported.
`resolutionStatus`	ENUM('Resolved', 'Pending', 'Escalated')	Status of the issue resolution.
`supervisorID`	INT	ID of the supervisor handling the issue.
`Call_Date`	DATETIME	Date and time of the call.

Guest Table

Column Name	Data Type (MySQL)	Description
`Guest_ID`	INT (Primary Key)	Unique identifier for the guest.
`travelCode`	INT (Foreign Key)	Unique identifier for the trip (linked with flights/hotels if applicable).
`Guest_name`	VARCHAR(255)	Guest's full name.
`Guest_Gender`	ENUM('Male', 'Female', 'Other')	Guest's gender.
`Age`	INT	Guest's age.
`Guest_PhoneNo`	VARCHAR(20)	Guest's phone number.
`Guest_email`	VARCHAR(255)	Guest's email address.
`idProof`	VARCHAR(255)	Identification details.

Review Table

Column Name	Data Type (MySQL)	Description
`travelCode`	INT (Foreign Key)	Unique identifier for the trip (linked with flights/hotels if applicable).
`User_ID`	INT (Foreign Key)	Unique user identifier (linked to users.csv).
`Car_rented`	VARCHAR(255)	Car rental status.
`review_car`	TEXT	Review for the car rental service.
`review_hotel`	TEXT	Review for the hotel.
`review_flights`	TEXT	Review for the flight.
`flight_rating`	INT	Rating for the flight (1-5).
`hotel_rating`	INT	Rating for the hotel (1-5).
`car_rating`	INT	Rating for the car rental (1-5).

`overall_rating`	DECIMAL(3,2)	Overall rating of the trip.

User Table

Column Name	Data Type (MySQL)	Description
`user_id`	INT (Primary Key)	Unique user identifier (linked to users.csv).
`company`	VARCHAR(255)	Company name associated with the user.
`name`	VARCHAR(255)	User's full name.
`gender`	ENUM('Male', 'Female', 'Other')	User's gender.
`age`	INT	User's age.
`HotelBookingReviews`	TEXT	User's reviews on hotel bookings.
`FlightBookingReviews`	TEXT	User's reviews on flight bookings.
`CarRentalReviews`	TEXT	User's reviews on car rentals.
`Rating`	INT	User's overall rating.
`travelCode`	INT	Unique identifier for the trip (linked with flights/hotels if applicable).

▼ MySQL Schema

1. User Table

```
CREATE TABLE User (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    company VARCHAR(255),
    name VARCHAR(255),
    gender ENUM('Male', 'Female', 'Other'),
    age INT,
    -- HotelBookingReviews TEXT,  -- Consider a separate re
    -- FlightBookingReviews TEXT, -- Consider a separate re
    -- CarRentalReviews TEXT,    -- Consider a separate revi
    -- Rating INT,               -- Consider a separate rev
    travelCode INT,  -- Consider moving this to a separate
    FOREIGN KEY (travelCode) REFERENCES Trips(travelCode) -
);
```

2. Trips Table (Recommended)

It's generally better practice to separate trip information into its own table, as one user might have multiple trips.

```
CREATE TABLE Trips (
    travelCode INT PRIMARY KEY AUTO_INCREMENT,  -- Or use a
    user_id INT,
    FOREIGN KEY (user_id) REFERENCES User(user_id),
    -- Add other trip-related details here if needed (e.g.,
);
```

3. Flight Table

```
CREATE TABLE Flight (
    travelCode INT,  -- Foreign key to Trips
    User_ID INT,     -- Foreign key to User (consider remov
    Departure VARCHAR(255),
    Arrival VARCHAR(255),
    flightType ENUM('Domestic', 'International'),
    Flight_price DECIMAL(10,2),
    Flight_duration DECIMAL(5,2),
    Flight_Distance DECIMAL(10,2),
    Flight_agency VARCHAR(255),
    Departure_date DATETIME,
    PRIMARY KEY (travelCode, User_ID), -- Composite key for
    FOREIGN KEY (travelCode) REFERENCES Trips(travelCode),
    FOREIGN KEY (User_ID) REFERENCES User(user_id)
);
```

4. Hotel Table

```
CREATE TABLE Hotel (
    travelCode INT,  -- Foreign key to Trips
    User_ID INT,     -- Foreign key to User (consider remov
    Hotel_Name VARCHAR(255),
    Arrival_place VARCHAR(255),
    Hotel_stay INT,
    Hotel_per_day_price DECIMAL(10,2),
```

```
    Check_in DATETIME,
    Hotel_TotalPrice DECIMAL(10,2),
    PRIMARY KEY (travelCode, User_ID), -- Composite key for
    FOREIGN KEY (travelCode) REFERENCES Trips(travelCode),
    FOREIGN KEY (User_ID) REFERENCES User(user_id)
);
```

5. Car Rental Table

```
CREATE TABLE CarRental (
    travelCode INT,  -- Foreign key to Trips
    User_ID INT,     -- Foreign key to User (consider remov
    Check_in DATETIME,
    pickupLocation VARCHAR(255),
    dropoffLocation VARCHAR(255),
    carType ENUM('Sedan', 'SUV', 'Hatchback', 'Luxury'),
    rentalAgency VARCHAR(255),
    rentalDuration INT,
    Car_total_distance INT,
    fuelPolicy ENUM('Full-to-Full', 'Prepaid'),
    Car_bookingStatus ENUM('Confirmed', 'Cancelled', 'Pendi
    total_rent_price DECIMAL(10,2),
    PRIMARY KEY (travelCode, User_ID), -- Composite key for
    FOREIGN KEY (travelCode) REFERENCES Trips(travelCode),
    FOREIGN KEY (User_ID) REFERENCES User(user_id)
);
```

6. Passenger Table

```
CREATE TABLE Passenger (
    User_ID INT,     -- Foreign key to User
    company VARCHAR(255),
    Name VARCHAR(255),
    gender_x ENUM('Male', 'Female', 'Other'),
    PRIMARY KEY (User_ID, Name), -- Composite key so a user
    FOREIGN KEY (User_ID) REFERENCES User(user_id)

);
```

7. Customer Call Table

```
CREATE TABLE CustomerCall (
    User_ID INT,        -- Foreign key to User
    Arrival_date DATETIME,
    issueType VARCHAR(255),
    resolutionStatus ENUM('Resolved', 'Pending', 'Escalatec
    supervisorID INT,  -- Consider a foreign key to a Super
    Call_Date DATETIME,
    PRIMARY KEY (User_ID, Call_Date), -- Composite key so a
    FOREIGN KEY (User_ID) REFERENCES User(user_id)
);
```

8. Guest Table

```
CREATE TABLE Guest (
    Guest_ID INT PRIMARY KEY AUTO_INCREMENT,
    travelCode INT,  -- Foreign key to Trips
    Guest_name VARCHAR(255),
    Guest_Gender ENUM('Male', 'Female', 'Other'),
    Age INT,
    Guest_PhoneNo VARCHAR(20),
    Guest_email VARCHAR(255),
    idProof VARCHAR(255),
    FOREIGN KEY (travelCode) REFERENCES Trips(travelCode)
);
```

9. Review Table

```
CREATE TABLE Review (
    review_id INT PRIMARY KEY AUTO_INCREMENT,  -- Add a pri
    travelCode INT,  -- Foreign key to Trips
    User_ID INT,        -- Foreign key to User
    Car_rented VARCHAR(255), -- Consider making this a bool
    review_car TEXT,
    review_hotel TEXT,
    review_flights TEXT,
    flight_rating INT,
```

```
      hotel_rating INT,
      car_rating INT,
      overall_rating DECIMAL(3,2),
      FOREIGN KEY (travelCode) REFERENCES Trips(travelCode),
      FOREIGN KEY (User_ID) REFERENCES User(user_id)
 );
```

Key Improvements and Considerations:

- **Trips Table:** Created a `Trips` table to better manage trip information and allow users to have multiple trips. This also simplifies the foreign key relationships in other tables.

- **Composite Keys:** Used composite keys in `Flight`, `Hotel`, `CarRental` to enforce that a user can only have one entry per trip for each of these categories.

- **Review Table:** Created a separate `Review` table with a primary key (`review_id`) for better management of reviews. This allows for multiple reviews per trip if needed in the future and makes querying reviews more efficient.

- **Foreign Keys:** Added foreign key constraints to enforce referential integrity between tables.

- **Data Types:** Used appropriate data types for each column.

- **Indexing:** Consider adding indexes to frequently queried columns (e.g., `user_id`, `travelCode`, `Departure_date`, `Arrival_place`, `Hotel_Name`, etc.) after you've populated the tables and analyzed your query patterns. Indexes significantly speed up data retrieval.

- **Normalization:** The schema is now more normalized, reducing data redundancy and improving data integrity.