**GROUP - 2**

**FLIGHT BOOKING SYSTEM DATABASE PROJECT**

Jahnavi Chintakindi (Team Co-ordinator)

SambamurthiRaju Dantuluri

Padmapadanand Bhaskara

Nara Swapna Raaga

## **INFO 5707 Section 022 - Data Modelling for Information Professionals (Fall 2023 1)**

**Prof.** [**Navya Reddy Velverthi**](https://unt.instructure.com/courses/95992/users/212929)

**PROJECT DESCRIPTION:**

In the new society, the process of reserving flights is changing to become a complicated and effective system, demanding increasingly advanced ways for managing reservations made by a growing client base. Air travel demand continues to rise, and airlines must find effective ways to meet it in a smooth manner. The way airlines handle aircraft reservations have been transformed over time by technological advancements, particularly the introduction of computers, which has resulted in a considerable shift away from manual techniques and paper processes.

Previously, airlines used manual techniques to manage flight reservations. From handwritten booking records to tangible ticketing systems, there was a lot of paperwork. This manual method was not only time-consuming and labour-intensive, but it was also prone to mistakes and inefficiencies. As the number of air passengers grows, so does the need for a more structured, precise, and effective system.

The emergence of computers revolutionized the aviation business. Airlines have had the option to transition from a manual, time-consuming manner of managing flight reservations to streamlined, automated systems. Through computerized reservation systems, airlines have been able to store, access, and manage booking records in a digitized fashion. As a result, the booking process is now faster and more precise, and the overall client experience has improved.

Today, the airline employs sophisticated computerized flight reservation systems that integrate many aspects of passenger information, aircraft availability, seat assignment, payment processing, and schedule management. These systems are linked, allowing for real-time updates and coordination among the airline's numerous divisions. Furthermore, passengers can use online reservation systems to make reservations, pick preferences, and simply acquire electronic tickets.

Overall, the transition from paper-based reservation procedures to highly efficient and automated computerized systems constitute a significant shift in the aviation industry. This transition not only streamlined the airline reservation process, but it also ensured a high level of consumer satisfaction and underlined the essential role that technology has played in modern aviation.

**OBJECTIVES:**

1. Efficiently manage and control all database data for flights, passengers, airports, flight services, seat assignments, flight costs, bookings, payment processing, and calendar scheduling.
2. Maintain detailed passenger flight records, including reservation and boarding information, with regular updates to maintain correctness.
3. Handle flight amenities, such as meal choices and special accommodations, to improve the passenger experience.
4. Allow passengers to purchase flights based on the flight services they desire.
5. Provide passengers with flight information based on airport locations and flight availability.
6. Keep records updated as transactions are executed to ensure real-time data correctness.
7. Using the comprehensive database system, provides easily retrieve and recover flight reservation records.
8. Once a flight ticket is booked, securely save transaction details, whether entered via credit or debit card.

**SCOPE:**

1. The primary goal is to protect the privacy of flight-related data and improve the effectiveness of managing airline operations.
2. Every airline ticket booking system user will be able to search for flight availability based on airport source and destination, type of flight, flight service, and calendar date and time.
3. The system automates the process of generating flight reports, providing useful information about flight availability and performance.
4. The system incorporates techniques to decrease data redundancy, ensuring that flight-related information is organized and maintained efficiently.
5. Passengers can select and reserve their preferred flight, as well as cancel or alter the ticket details.
6. This database is capable of handling numerous ticket bookings as well as data storage recovery.
7. Allows for the efficient handling of data redundancy.
8. Overall, the flight booking system database provides practicality, easy flight scheduling with passenger customisation, and flight reservation data storage and retrieval.
9. Data backups can be offered by the system as needed.

**PROJECT REQUIREMENTS:**

**Operating System**: WINDOWS

**Database**: SQL Server

**Applications:** MS Word, MS Power Point

**DATABASE REQUIREMENTS:**

The following is a list that provides the different data tables that will be used.

1. Airport Table
2. Passenger Table
3. Flight\_Details Table
4. Service\_Offering Table
5. Seat\_Details Table
6. Flight\_Cost Table
7. Reservation Table
8. Travel\_Class Table
9. Calendar Table
10. Payment\_Status Table

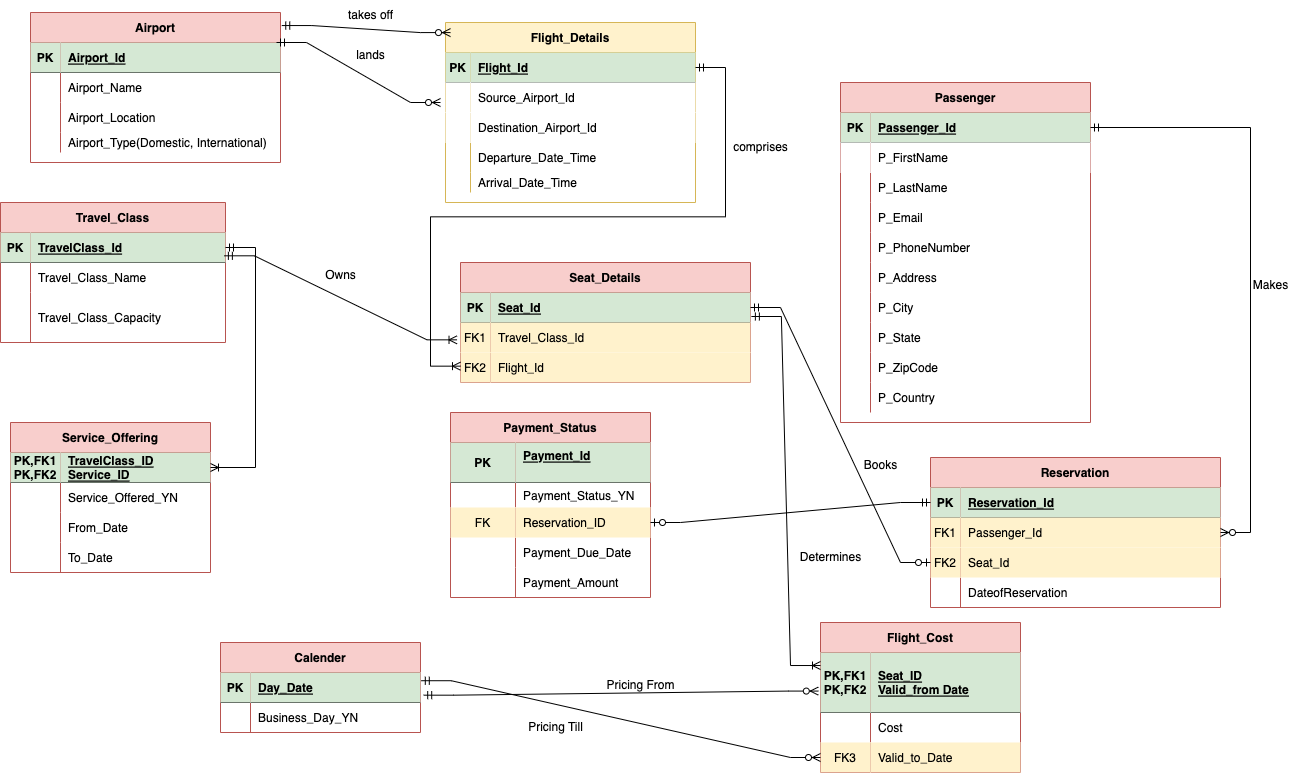
**USER REQUIREMENTS:**

1. Users may search for flight reservations and maintain track of passenger bookings. This comprises information such as the Reservation ID, Flight ID, Passenger ID, flight date and time, ticket pricing, and any revisions to this information.
2. The system shall generate the list of flights available at that calendar date each day. Details of the passenger include their first name, last name, reservation\_id, passenger ’s\_age, gender, address, email address, and phone number.
3. Based on the passenger’s choice of flight the ideal flight ticket is selected to confirm the flight ticket which consists of travel\_id, passenger\_id, flight\_details, and reservation details.
4. Once the ticket is booked, the passenger can retrieve the ticket details from the system to review and modify the ticket to cancel, update, and save the changes.
5. Finally, once the ticket is confirmed the passenger is requested to make a payment and the payment details are shown based on the calendar date chosen and cost, and payment is made.
6. Payment details are maintained in the database that consists of information having payment ID, bill ID, payment amount, and payment date.
7. Data is systematically organized and stored in designated files. To achieve this, data records are created, which store information such as Record ID, Log ID, Passenger ID, login and logout times, and the status of the flight.

**BUSINESS RULES:**

1. Each airport can have multiple flight details associated with their respective source and destination.
2. Each flight may offer various flight services such as Food Service, Entertainment Service, and Wifi Service. These services are offered to many flights.
3. Service\_Offering is an associative entity that stores Service\_ID and Flight\_ID, linking specific services to specific flights.
4. Each flight must have one or more seats, and each seat is associated with a particular Travel\_Class\_ID. Seats can belong to classes like Business Class or Economy Class.
5. Passengers can make reservations for seats. When a passenger reserves a seat, it will be assigned based on their choice. Each reservation is linked to a specific Seat\_Details, Travel\_Class, and Flight\_Details.
6. For each seat reserved, a passenger may or may not pay to confirm the seat. The Payment\_Status entity tracks whether the payment has been made or not.
7. A passenger can reserve one seat per Reservation\_ID.
8. Payment\_Status is an associative entity that holds information related to a particular Reservation\_ID and Payment\_ID.
9. Payment information, including due date and amount, is stored in the Payment entity.
10. Flight\_Cost can be determined based on information from Seat\_Details and the Calendar. This includes the seat's travel class and the date when the seat is booked.

**ER DIAGRAM FOR FLIGHT BOOKING SYSTEM DATABASE PROJECT**



**DATA DICTIONARY:**

A Data Dictionary is a catalogue – a repository – of the elements in a system. As the name suggests, these elements center on the data and the way they are structured to meet user requirements and organization needs. In a Data Dictionary you will find a list of all the elements are data flow data stores and processes. The Data Dictionary stores details and description of these elements.

**DATA DICTIONARY IMPORTANCE:**

* Analysts use Data Dictionaries for a variety of reasons.
* To keep track of the details in a large system.
* To communicate a unified meaning for all system components.
* To document the system's features.
* To make it easier to analyze details in order to evaluate characteristics and decide where system changes should be made.
* To locate system errors and omissions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Table Name** | **Attribute Name** | **Description** | **Data Type** | **Data Format** | **Required** | **PK or FK** |
| **Airport** | Airport\_Id  Airport\_Name  Airport\_Location  Airport\_Type | Unique number for the Airport  Name of the Airport  Locality of Airport  Type of Airport | INTEGER VARCHAR VARCHAR VARCHAR | 99999  XXXX  XXXX  XXXX | Y  Y  Y  Y | PK |
| **Flight Details** | Flight\_Id Airport\_Id  Destination\_Airport\_ID  Departure\_Date\_Time  Arrival\_Date\_Time | Unique number of flight Unique number of Airport source Unique number of Airport destination  Departure date and time of flight  Arrival date and time of flight | INTEGER  INTEGER  INTEGER DATETIME DATETIME | 99999  99999  99999 YYYY-MM DD YYYY-MM-DD | Y  Y  Y  Y  Y | PK  FK1  FK2 |
| **Passenger** | Passenger\_Id  P\_FirstName  P\_LastName  P\_Email  P\_PhoneNumber  P\_Address  P\_City  P\_State  P\_Zipcode  P\_Country | Unique Id of the Passenger  Firstname of the Passenger  Lastname of the Passenger  Email Id of the Passenger  Phonenumber of the passenger  Address of the Passenger  City of the Passenger  State of the Passenger  Zipcode of the Passenger  Country of the Passenger | INTEGER  VARCHAR  VARCHAR  VARCHAR  INTEGER  VARCHAR  VARCHAR  VARCHAR  INTEGER  VARCHAR | 99999  XXXX  XXXX  XXXX  99999  XXXX  XXXX  XXXX  99999  XXXX | Y  Y  Y  Y  Y  Y  Y  Y  Y  Y | PK |
| **Travel\_Class** | TravelClass\_Id  Travel\_Class\_Name  Travel\_Class\_Capacity | Identification number of TravelClass  Name of the Travel Class  Capacity of the Travel class | INTEGER  VARCHAR  VARCHAR | 99999  XXXX  XXXX | Y  Y  Y | PK |
| **Payment\_Status** | Payment\_Id  Payment\_DueDate  Payment\_Amount  Reservation\_ID  Payment\_Status\_YN | Identification number of the Payment  Last Date of the payment  Amount to be paid  Identification number of the Reservation  Status of the payment | INTEGER  DATETIME  INTEGER  INTEGER  VARCHAR | 99999  YYYY-MM-DD  99999  99999  xxxxxx | Y  Y  Y  Y  Y | PK    FK |
| **Reservation** | Reservation\_Id  Passenger\_Id  Seat\_Id  DateofReservation | Identification number of the Reservation  Identification number of the Passenger  Identification number of the Seat  Reservation date | INTEGER  INTEGER  INTEGER  DATETIME | 99999  99999  99999  YYYY-MM-DD | Y  Y  Y  Y | PK  FK1  FK2 |
| **Service\_Offering** | Service\_ID  Flight\_ID  Service\_Offered\_YN  From\_Date  To\_Date | Idenification number of the service  Identification number of flight  If Service is offered or not  Service start date  Service end date | INTEGER  INTEGER  VARCHAR  DATETIME  DATETIME | 99999  99999  XXXX  YYYY-MM-DD  YYYY-MM-DD | Y  Y  Y  Y  Y | PK,FK1  PK,FK2 |
| **Service\_Offering** | Service\_ID  Service\_Name | Idenification number of the service  Type of the Service | INTEGER  VARCHAR | 99999  XXXXX | Y  Y | PK |
| **Calender** | Day\_Date  Business\_Day\_YN | The date chosen on the calender  the business day choice | DATETIME  DATETIME | YYYY-MM-DD  YYYY-MM-DD | Y  Y | PK |
| **Seat\_Details** | Seat\_ID  Travel\_Class\_ID  Flight\_ID | Identification number of the Seat  Identification number of TravelClass  Identification number of flight | INTEGER  INTEGER  INTEGER | 99999  99999  99999 | Y  Y  Y | PK  FK1  FK2 |

**Queries and Operations**

**Section 1: Entity Generation and Data Entry: Create Database, Create Tables, and Insert Values to the Tables**

**Entity Generation and Data Entry for Table Airport:**

**Statements Explanation:**

* Firstly, the "Table\_Airport" is established through the CREATE TABLE query, outlining key columns: AIRPORT\_ID, AIRPORTNAME, AIRPORTCITY, and AIRPORTCOUNTRY.
* Secondly, column definitions specify the role of each attribute, designating AIRPORT\_ID as a unique identifier, AIRPORTNAME for the airport's name, AIRPORTCITY indicating its location, and AIRPORTCOUNTRY specifying the country.
* Next, the primary key constraint is introduced with CONSTRAINT AIRPORT\_PK PRIMARY KEY (AIRPORT\_ID), ensuring the uniqueness of each airport record and preventing duplicates by utilizing AIRPORT\_ID as the primary key.
* Finally, a series of INSERT INTO statements populate "Table\_Airport" with data for 25 distinct airports, providing comprehensive details such as names, cities, and countries, thereby completing the initial dataset for the database.

**CREATE TABLE** **QUERY:**

CREATE TABLE Table\_Airport

(

AIRPORT\_ID INT NOT NULL,

AIRPORTNAME VARCHAR(100),

AIRPORTCITY VARCHAR(100),

AIRPORTCOUNTRY VARCHAR(100),

CONSTRAINT AIRPORT\_PK PRIMARY KEY (AIRPORT\_ID)

);

**INSERT STATEMENTS:**

INSERT INTO Airport (Airport\_ID, AirportName, AirportCity, AirportCountry)

VALUES

(1, 'JFK International Airport', 'New York City', 'United States'),

(2, 'Heathrow Airport', 'London', 'United Kingdom'),

(3, 'Changi Airport', 'Singapore', 'Singapore'),

(4, 'Los Angeles International Airport', 'Los Angeles', 'United States'),

(5, 'Beijing Capital International Airport', 'Beijing', 'China'),

(6, 'Dubai International Airport', 'Dubai', 'United Arab Emirates'),

(7, 'Tokyo Haneda Airport', 'Tokyo', 'Japan'),

(8, 'O''Hare International Airport', 'Chicago', 'United States'), -- Corrected apostrophe

(9, 'Indira Gandhi International Airport', 'Delhi', 'India'),

(10, 'Soekarno-Hatta International Airport', 'Jakarta', 'Indonesia'), -- Fixed value

(11, 'Shanghai Pudong International Airport', 'Shanghai', 'China'),

(12, 'Charles de Gaulle Airport', 'Paris', 'France'),

(13, 'Sydney Airport', 'Sydney', 'Australia'),

(14, 'Incheon International Airport', 'Seoul', 'South Korea'),

(15, 'Frankfurt Airport', 'Frankfurt', 'Germany'),

(16, 'Madrid-Barajas Adolfo Suárez Airport', 'Madrid', 'Spain'),

(17, 'Denver International Airport', 'Denver', 'United States'),

(18, 'Suvarnabhumi Airport', 'Bangkok', 'Thailand'),

(19, 'Toronto Pearson International Airport', 'Toronto', 'Canada'),

(20, 'Hong Kong International Airport', 'Hong Kong', 'China'),

(21, 'Munich Airport', 'Munich', 'Germany'),

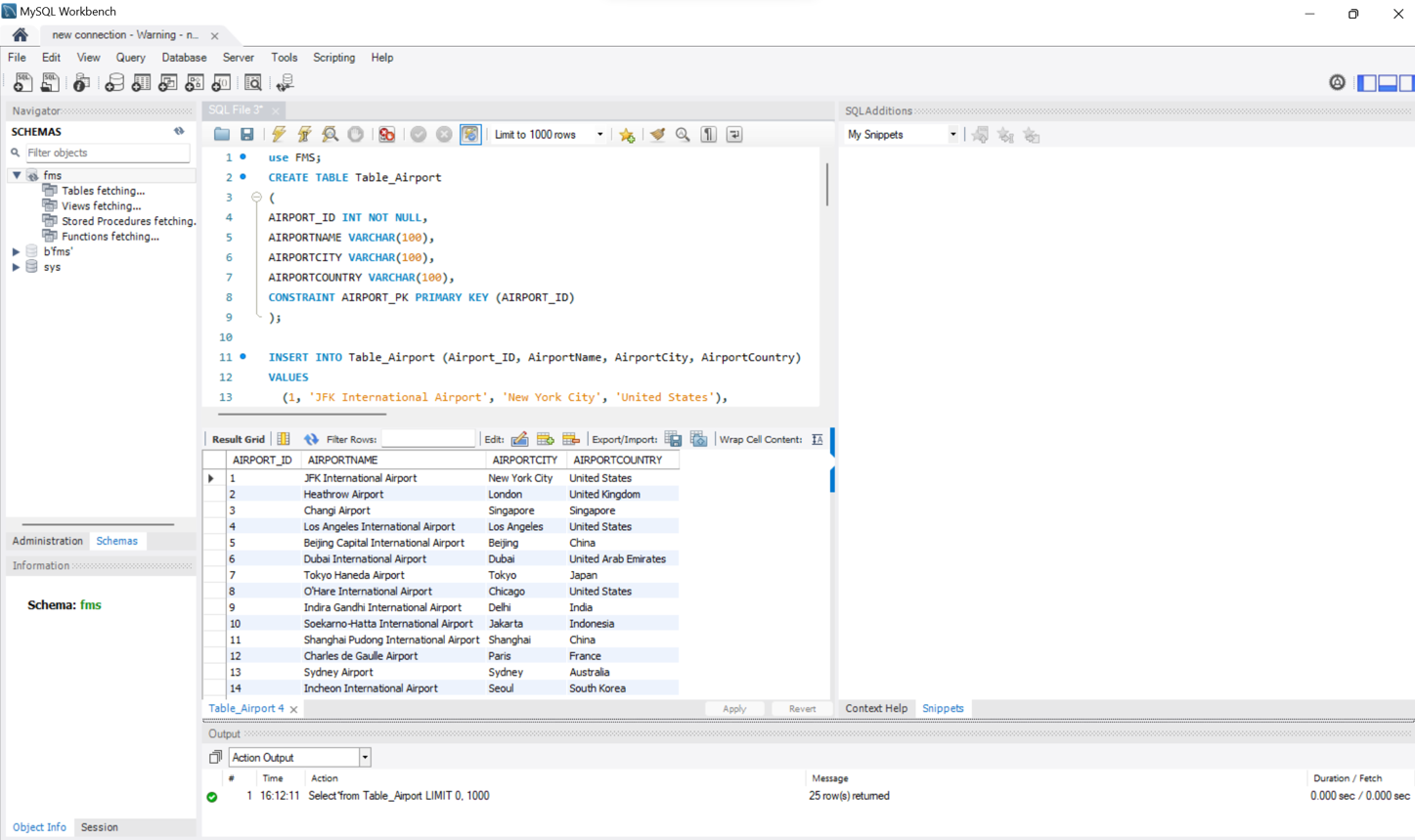
(22, 'Dallas/Fort Worth International Airport', 'Dallas', 'United States'),

(23, 'Amsterdam Airport Schiphol', 'Amsterdam', 'Netherlands'),

(24, 'Guangzhou Baiyun International Airport', 'Guangzhou', 'China'),

(25, 'Singapore Changi Airport', 'Singapore', 'Singapore');

**RESULT:**



**Entity Generation and Data Entry for Table Passenger:**

**Statements Explanation:**

* The SQL statements define and populate the "PASSENGER" table. Firstly, the CREATE TABLE query establishes the table structure, including columns such as PASSENGER\_ID (integer, not null), P\_FIRSTNAME, P\_LASTNAME, P\_EMAIL (unique, not null), P\_PHONENUMBER (unique, not null), P\_ADDRESS, P\_CITY, P\_STATE, P\_ZIPCODE, and P\_COUNTRY. The primary key constraint is set on the PASSENGER\_ID column.
* Secondly, a series of INSERT INTO statements populate the "PASSENGER" table with data for 25 passengers. Each INSERT INTO statement includes values for the specified columns, providing details such as first name, last name, email, phone number, address, city, state, zip code, and country. The dataset represents a diverse set of passenger information.

**QUERY:** Table Passenger

CREATE TABLE PASSENGER

(

PASSENGER\_ID INT NOT NULL,

P\_FIRSTNAME VARCHAR(100),

P\_LASTNAME VARCHAR(100),

P\_EMAIL VARCHAR(100) NOT NULL UNIQUE ,

P\_PHONENUMBER VARCHAR(10) NOT NULL UNIQUE,

P\_ADDRESS VARCHAR(100),

P\_CITY VARCHAR(100),

P\_STATE VARCHAR(100),

P\_ZIPCODE VARCHAR(5) ,

P\_COUNTRY VARCHAR(100),

PRIMARY KEY (PASSENGER\_ID)

);

**INSERT STATEMENTS:**

INSERT INTO PASSENGER (PASSENGER\_ID, P\_FIRSTNAME, P\_LASTNAME, P\_EMAIL, P\_PHONENUMBER, P\_ADDRESS, P\_CITY, P\_STATE, P\_ZIPCODE, P\_COUNTRY)

VALUES

(1, 'John', 'Doe', 'john@example.com', 1234567890, '123 Main St', 'City', 'State', '12345', 'Country'),

(2, 'Jane', 'Smith', 'jane@example.com', 9876543210, '456 Elm St', 'Town', 'State', '21456', 'Country'),

(3, 'Alice', 'Johnson', 'alice@example.com', 5555558555, '789 Oak St', 'Village', 'State', '34567', 'Country'),

(4, 'Bob', 'Williams', 'bob@example.com', 1112223333, '321 Maple St', 'Hamlet', 'State', '45678', 'Country'),

(5, 'Emily', 'Brown', 'emily@example.com', 4444444444, '654 Birch St', 'County', 'State', '56789', 'Country'),

(6, 'Sarah', 'Davis', 'sarah@example.com', 7778889999, '987 Pine St', 'Suburb', 'State', '67890', 'Country'),

(7, 'Michael', 'Wilson', 'michael@example.com', 6667778888, '135 Cedar St', 'District', 'State', '78901', 'Country'),

(8, 'Rachel', 'Miller', 'rachel@example.com', 3333332333, '246 Spruce St', 'Borough', 'State', '89012', 'Country'),

(9, 'David', 'Taylor', 'david@example.com', 2221110000, '579 Walnut St', 'State', 'State', '90123', 'Country'),

(10, 'Emma', 'Garcia', 'emma@example.com', 9999990999, '753 Cherry St', 'Province', 'State', '01234', 'Country'),

(11, 'Olivia', 'Martinez', 'olivia@example.com', 8885888888, '864 Vine St', 'Territory', 'State', '12365', 'Country'),

(12, 'William', 'Johnson', 'william@example.com', 7773777777, '921 Magnolia St', 'Region', 'State', '23456', 'Country'),

(13, 'Sophia', 'Hernandez', 'sophia@example.com', 6666666666, '246 Rose St', 'Sector', 'State', '34567', 'Country'),

(14, 'James', 'Gonzalez', 'james@example.com', 5555505555, '753 Daisy St', 'Division', 'State', '45678', 'Country'),

(15, 'Ava', 'Perez', 'ava@example.com', 4444844444, '864 Tulip St', 'Area', 'State', '56789', 'Country'),

(16, 'Logan', 'Torres', 'logan@example.com', 3333733333, '921 Lily St', 'Zone', 'State', '67890', 'Country'),

(17, 'Ethan', 'Ramirez', 'ethan@example.com', 2222221222, '135 Orchid St', 'Precinct', 'State', '78901', 'Country'),

(18, 'Mia', 'Flores', 'mia@example.com', 1111121111, '579 Sunflower St', 'Parish', 'State', '89012', 'Country'),

(19, 'Harper', 'Cruz', 'harper@example.com', 4010020500, '753 Poppy St', 'Municipality', 'State', '90123', 'Country'),

(20, 'Alexander', 'Scott', 'alexander@example.com', 9999099699, '864 Iris St', 'Canton', 'State', '01234', 'Country'),

(21, 'Grace', 'Phillips', 'grace@example.com', 8888388858, '921 Lotus St', 'Township', 'State', '12645', 'Country'),

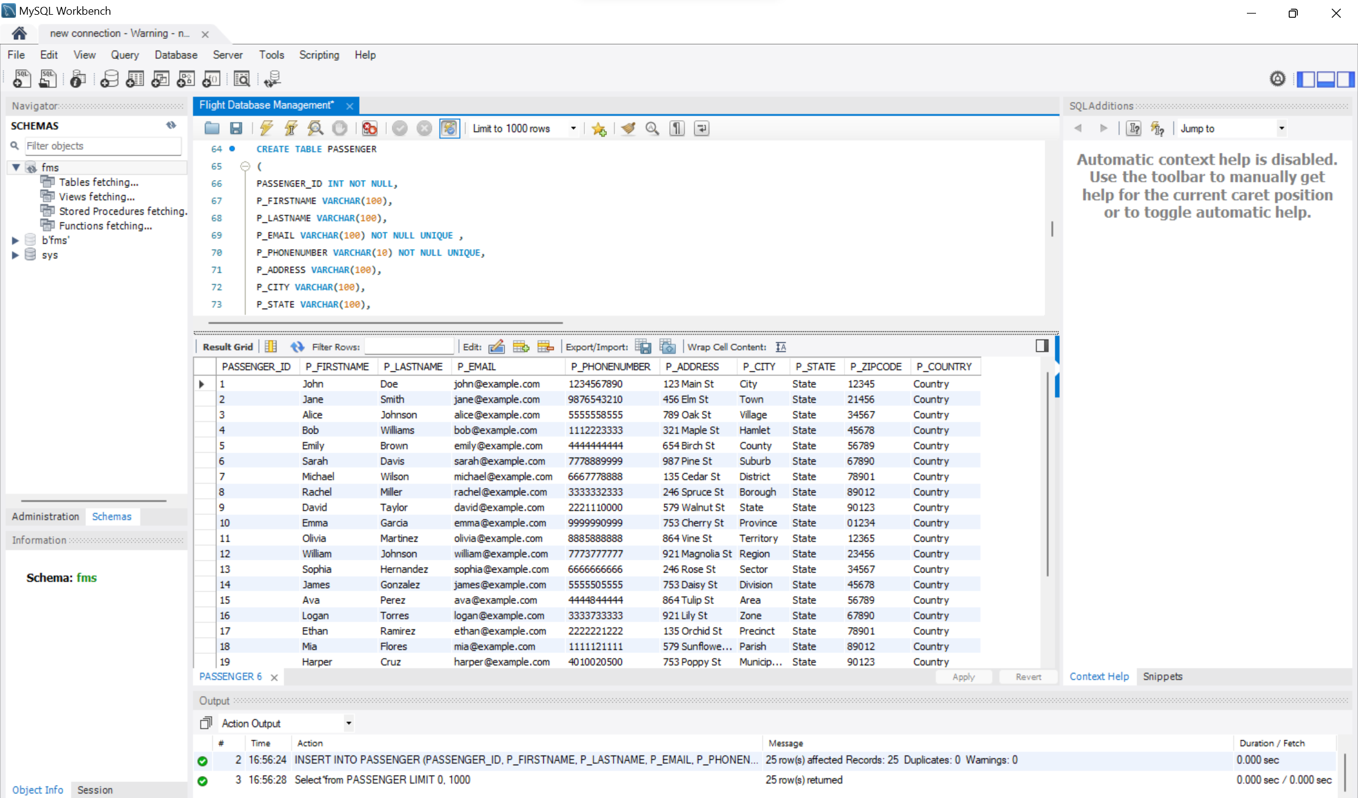
(22, 'Lucas', 'Evans', 'lucas@example.com', 7774777787, '246 Violet St', 'Territory', 'State', '22456', 'Country'),

(23, 'Liam', 'Stewart', 'liam@example.com', 6666466696, '753 Daisy St', 'Zone', 'State', '34567', 'Country'),

(24, 'Zoe', 'Turner', 'zoe@example.com', 5555545555, '864 Lily St', 'Sector', 'State', '45678', 'Country'),

(25, 'Evelyn', 'Ward', 'evelyn@example.com', 4444344444, '921 Orchid St', 'Division', 'State','35396','Country');

**RESULT:**



**Entity Generation and Data Entry for Table Travel\_Class:**

**Statements Explanation:**

* The SQL statements pertain to the creation and population of the "TRAVEL\_CLASS" table. Firstly, the **CREATE TABLE** query defines the table structure, incorporating columns such as **TRAVEL\_CLASS\_ID**, **TRAVEL\_CLASS\_NAME** with a restricted set of values, and **TRAVEL\_CLASS\_CAPACITY**. The primary key constraint ensures the uniqueness of each travel class.
* Secondly, a set of **INSERT INTO** statements populate the table with data for 25 distinct travel classes, specifying details such as class names and their corresponding capacities. The dataset showcases a range of travel classes, each adhering to the predefined class name constraints and providing diverse capacity values.

**QUERY:** Table Travel\_Class

CREATE TABLE TRAVEL\_CLASS  
(  
 TRAVEL\_CLASS\_ID INT NOT NULL,  
 TRAVEL\_CLASS\_NAME VARCHAR(100) CONSTRAINT NAME\_LIST\_CHK CHECK (TRAVEL\_CLASS\_NAME IN('FIRST CLASS','BUSINESS CLASS','PREMIUM ECONOMY','ECONOMY CLASS','BASIC ECONOMY')),  
 TRAVEL\_CLASS\_CAPACITY BIGINT,  
 CONSTRAINT TRAVEL\_CLASS\_PK PRIMARY KEY (TRAVEL\_CLASS\_ID)  
);

**INSERT STATEMENTS:**

INSERT INTO Travel\_Class (Travel\_Class\_ID, Travel\_Class\_Name, Travel\_Class\_Capacity)

VALUES

(1, 'First Class', 50),

(2, 'Business Class', 100),

(3, 'Premium Economy', 150),

(4, 'Economy Class', 200),

(5, 'Basic Economy', 250),

(6, 'First Class', 50),

(7, 'Business Class', 100),

(8, 'Premium Economy', 150),

(9, 'Economy Class', 200),

(10, 'Basic Economy', 250),

(11, 'First Class', 50),

(12, 'Business Class', 100),

(13, 'Premium Economy', 150),

(14, 'Economy Class', 200),

(15, 'Basic Economy', 250),

(16, 'First Class', 50),

(17, 'Business Class', 100),

(18, 'Premium Economy', 150),

(19, 'Economy Class', 200),

(20, 'Basic Economy', 250),

(21, 'First Class', 50),

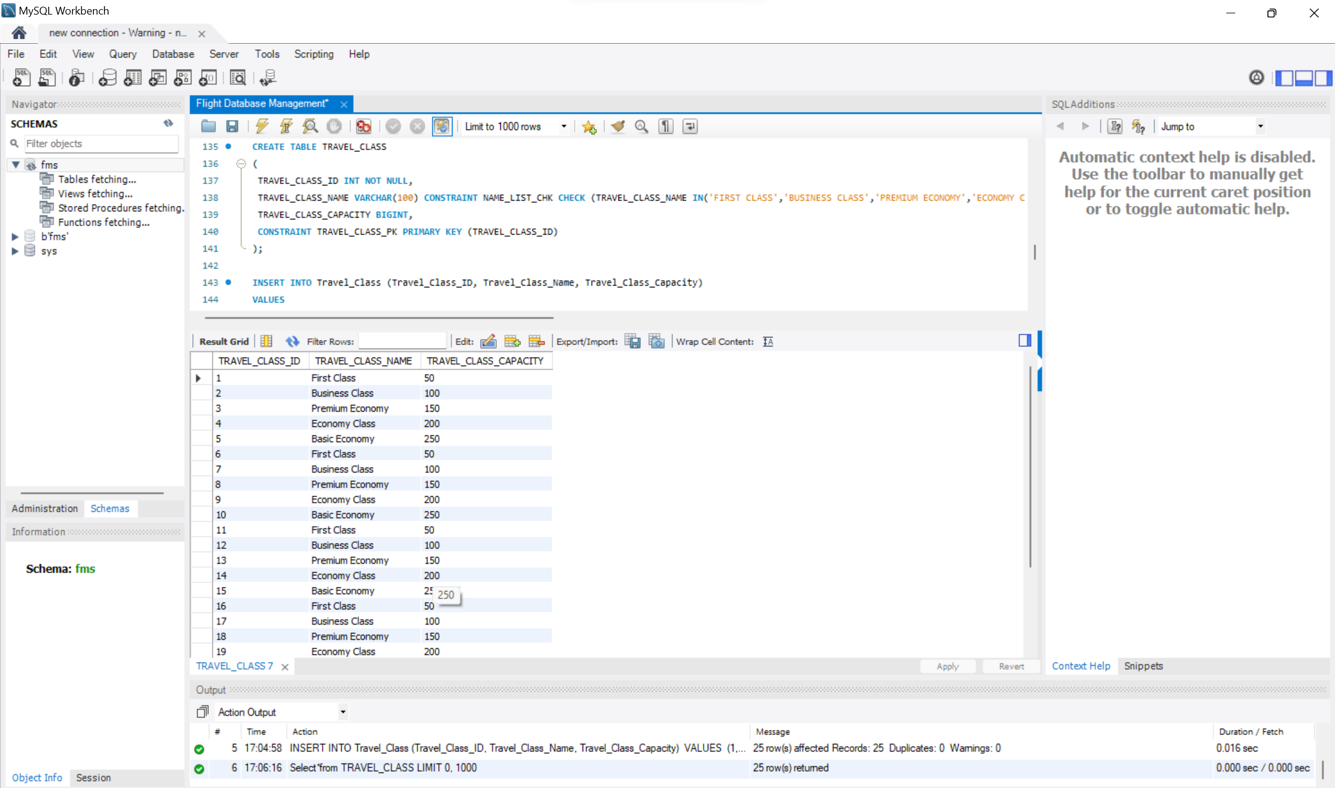
(22, 'Business Class', 100),

(23, 'Premium Economy', 150),

(24, 'Economy Class', 200),

(25, 'Basic Economy', 250);

**RESULT:**



**Entity Generation and Data Entry for Table Calendar:**

**Statements Explanation:**

* The SQL statements pertain to the creation and population of the "CALENDAR" table. Firstly, the **CREATE TABLE** query defines the table structure with columns such as **DAY\_DATE** (date, not null) and **BUSINESS\_DAY\_YN** (char, restricted to 'Y' or 'N'). A check constraint ensures the character values are limited to the specified set. The primary key constraint is applied to the **DAY\_DATE** column, ensuring unique date entries.
* Secondly, a set of **INSERT INTO** statements populate the table with data for 25 specific dates, marking each day as a business day ('Y') or not ('N'). The dataset represents a calendar with designated business days, providing a clear distinction for each date in the database.

**QUERY:** TABLE CALENDAR

CREATE TABLE CALENDAR  
(  
 DAY\_DATE DATE NOT NULL,  
 BUSINESS\_DAY\_YN CHAR(1) CONSTRAINT CHECK\_CHARACTER\_BUSINESS\_DAY\_YN CHECK(BUSINESS\_DAY\_YN IN ('Y','N')),  
 CONSTRAINT CALENDAR\_PK PRIMARY KEY (DAY\_DATE)  
);

**INSERT STATEMENTS:**

INSERT INTO CALENDAR (DAY\_DATE, BUSINESS\_DAY\_YN)

VALUES

('2023-01-01', 'N'),

('2023-01-02', 'Y'),

('2023-01-03', 'Y'),

('2023-01-04', 'Y'),

('2023-01-05', 'Y'),

('2023-01-06', 'Y'),

('2023-01-07', 'N'),

('2023-01-08', 'N'),

('2023-01-09', 'Y'),

('2023-01-10', 'Y'),

('2023-01-11', 'Y'),

('2023-01-12', 'Y'),

('2023-01-13', 'Y'),

('2023-01-14', 'N'),

('2023-01-15', 'N'),

('2023-01-16', 'Y'),

('2023-01-17', 'Y'),

('2023-01-18', 'Y'),

('2023-01-19', 'Y'),

('2023-01-20', 'Y'),

('2023-01-21', 'N'),

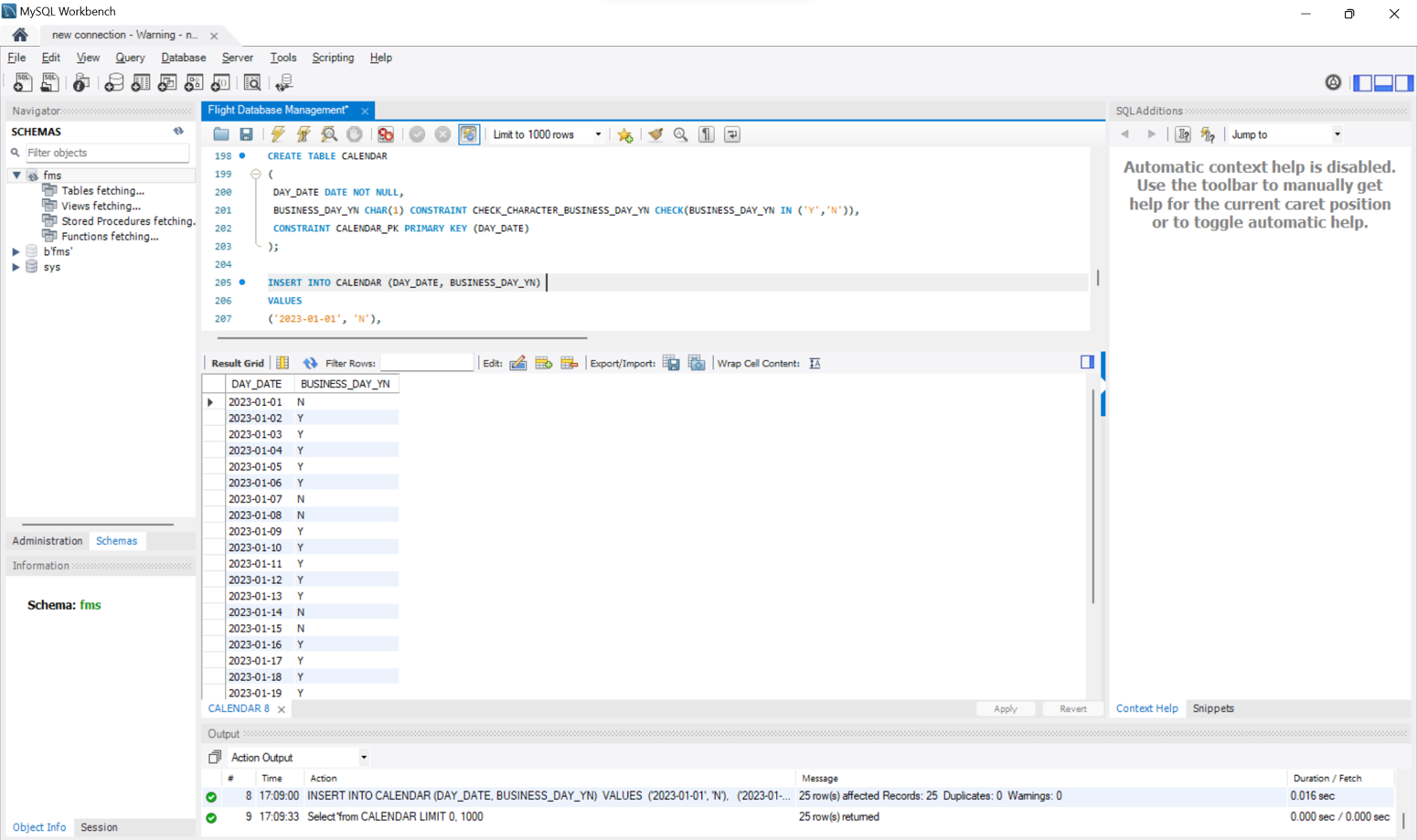
('2023-01-22', 'N'),

('2023-01-23', 'Y'),

('2023-01-24', 'Y'),

('2023-01-25', 'Y');

**RESULT:**



**Entity Generation and Data Entry for Table Flight Service:**

**Statements Explanation:**

* The SQL statements involve the creation and population of the "FLIGHT\_SERVICE" table. Firstly, the **CREATE TABLE** query defines the table structure with columns including **SERVICE\_ID** (integer, not null) and **SERVICE\_NAME** (varchar, constrained to predefined values). A check constraint ensures that **SERVICE\_NAME** values are limited to a specific set, including 'FOOD,' 'FRENCH WINE,' 'WIFI,' 'ENTERTAINMENT,' and 'LOUNGE.' The primary key constraint is applied to **SERVICE\_ID** to ensure unique service entries.
* Secondly, a set of **INSERT INTO** statements populate the table with data for 25 distinct flight services, specifying **SERVICE\_ID** and **SERVICE\_NAME** for each. The dataset reflects a variety of flight services, maintaining conformity with the predefined set of service names

**QUERY:** TABLE FLIGHT\_SERVICE

CREATE TABLE FLIGHT\_SERVICE

(

SERVICE\_ID INT NOT NULL,

SERVICE\_NAME VARCHAR(100) CONSTRAINT SERVICE\_CHK CHECK(SERVICE\_NAME IN ('FOOD','FRENCH WINE','WIFI','ENTERTAINMENT','LOUNGE')),

CONSTRAINT FLIGHT\_SERVICE\_PK PRIMARY KEY (SERVICE\_ID)

);

**INSERT STATEMENTS:**

INSERT INTO FLIGHT\_SERVICE (SERVICE\_ID, SERVICE\_NAME)

VALUES

(1, 'FOOD'),

(2, 'FRENCH WINE'),

(3, 'WIFI'),

(4, 'ENTERTAINMENT'),

(5, 'LOUNGE'),

(6, 'FOOD'),

(7, 'FRENCH WINE'),

(8, 'WIFI'),

(9, 'ENTERTAINMENT'),

(10, 'LOUNGE'),

(11, 'FOOD'),

(12, 'FRENCH WINE'),

(13, 'WIFI'),

(14, 'ENTERTAINMENT'),

(15, 'LOUNGE'),

(16, 'FOOD'),

(17, 'FRENCH WINE'),

(18, 'WIFI'),

(19, 'ENTERTAINMENT'),

(20, 'LOUNGE'),

(21, 'FOOD'),

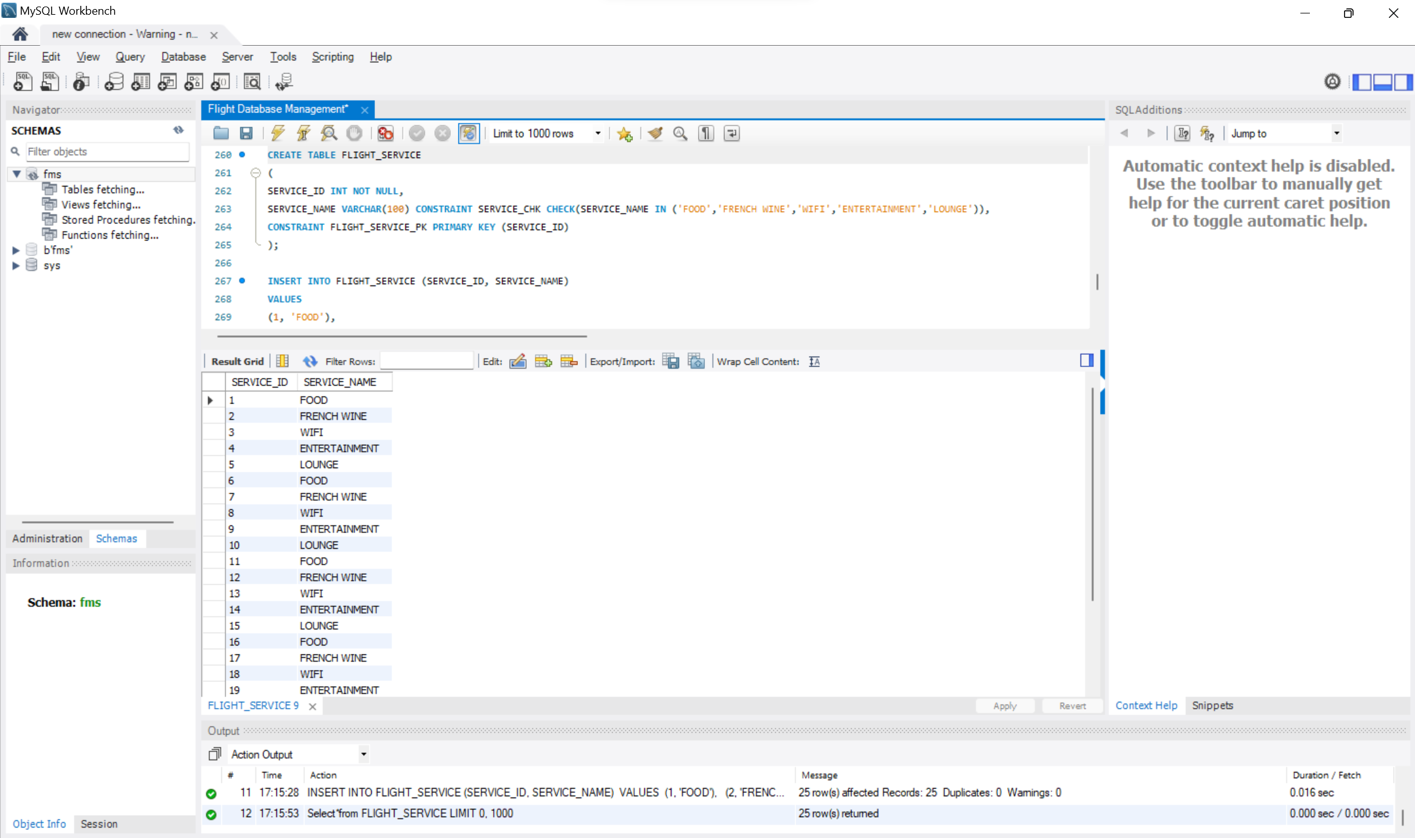
(22, 'FRENCH WINE'),

(23, 'WIFI'),

(24, 'ENTERTAINMENT'),

(25, 'LOUNGE');

**RESULT:**



**Entity Generation and Data Entry for Table Flight Service:**

**Statements Explanation:**

* The "FLIGHT\_DETAILS" table is created to store information about flights, including details such as **FLIGHT\_ID**, **AIRPORT\_ID** (source airport), **DESTINATION\_AIRPORT\_ID** (destination airport), **DEPARTURE\_DATE\_TIME**, **ARRIVAL\_DATE\_TIME**, and **AIRPLANE\_TYPE**.
* Primary key and foreign key constraints are applied to ensure data integrity. Additional constraints include a check for the relationship between departure and arrival date-time and an airport check to ensure the source and destination airports are distinct. The table is populated with 25 entries, representing diverse flights with specific details that comply with the defined constraints.

**QUERY:** Flight Details

CREATE TABLE FLIGHT\_DETAILS (

FLIGHT\_ID INT NOT NULL,

AIRPORT\_ID INT NOT NULL,

DESTINATION\_AIRPORT\_ID INT NOT NULL ,

DEPARTURE\_DATE\_TIME DATETIME,-- CONSTRAINT DATE\_CHECK CHECK (DEPARTURE\_DATE\_TIME< ARRIVAL\_DATE\_TIME),

ARRIVAL\_DATE\_TIME DATETIME,

AIRPLANE\_TYPE VARCHAR(100) CONSTRAINT AIRPLANE\_CHECK CHECK(AIRPLANE\_TYPE IN ('AIRBUS A380','BOEING 747')),

CONSTRAINT FLIGHT\_DETAILS\_PK PRIMARY KEY (FLIGHT\_ID),

CONSTRAINT FLIGHT\_DETAILS\_SOURCE\_FK1 FOREIGN KEY (AIRPORT\_ID) REFERENCES Table\_Airport(AIRPORT\_ID),

CONSTRAINT FLIGHT\_DETAILS\_DESTINATION\_FK2 FOREIGN KEY (DESTINATION\_AIRPORT\_ID) REFERENCES Table\_Airport(AIRPORT\_ID),

CONSTRAINT DATE\_CHECK\_FD CHECK (DEPARTURE\_DATE\_TIME< ARRIVAL\_DATE\_TIME),

CONSTRAINT AIRPORT\_CHK CHECK (AIRPORT\_ID != DESTINATION\_AIRPORT\_ID));

**INSERT STATEMENTS:**

INSERT INTO Flight\_Details (Flight\_ID, AIRPORT\_ID, Destination\_Airport\_ID, Departure\_Date\_Time, Arrival\_Date\_Time, Airplane\_Type)

VALUES

(1, 1, 2, '2023-01-01 08:00:00', '2023-01-01 10:00:00', 'Airbus A380'),

(2, 3, 4, '2023-01-02 08:00:00', '2023-01-02 10:00:00', 'Boeing 747'),

(3, 5, 6, '2023-01-03 08:00:00', '2023-01-03 10:00:00', 'Airbus A380'),

(4, 7, 8, '2023-01-04 08:00:00', '2023-01-04 10:00:00', 'Boeing 747'),

(5, 9, 10, '2023-01-05 08:00:00', '2023-01-05 10:00:00', 'Airbus A380'),

(6, 11, 12, '2023-01-06 08:00:00', '2023-01-06 10:00:00', 'Boeing 747'),

(7, 13, 14, '2023-01-07 08:00:00', '2023-01-07 10:00:00', 'Airbus A380'),

(8, 15, 16, '2023-01-08 08:00:00', '2023-01-08 10:00:00', 'Boeing 747'),

(9, 17, 18, '2023-01-09 08:00:00', '2023-01-09 10:00:00', 'Airbus A380'),

(10, 19, 20, '2023-01-10 08:00:00', '2023-01-10 10:00:00', 'Boeing 747'),

(11, 21, 22, '2023-01-11 08:00:00', '2023-01-11 10:00:00', 'Airbus A380'),

(12, 23, 24, '2023-01-12 08:00:00', '2023-01-12 10:00:00', 'Boeing 747'),

(13, 25, 1, '2023-01-13 08:00:00', '2023-01-13 10:00:00', 'Airbus A380'),

(14, 2, 3, '2023-01-14 08:00:00', '2023-01-14 10:00:00', 'Boeing 747'),

(15, 4, 5, '2023-01-15 08:00:00', '2023-01-15 10:00:00', 'Airbus A380'),

(16, 6, 7, '2023-01-16 08:00:00', '2023-01-16 10:00:00', 'Boeing 747'),

(17, 8, 9, '2023-01-17 08:00:00', '2023-01-17 10:00:00', 'Airbus A380'),

(18, 10, 11, '2023-01-18 08:00:00', '2023-01-18 10:00:00', 'Boeing 747'),

(19, 12, 13, '2023-01-19 08:00:00', '2023-01-19 10:00:00', 'Airbus A380'),

(20, 14, 15, '2023-01-20 08:00:00', '2023-01-20 10:00:00', 'Boeing 747'),

(21, 16, 17, '2023-01-21 08:00:00', '2023-01-21 10:00:00', 'Airbus A380'),

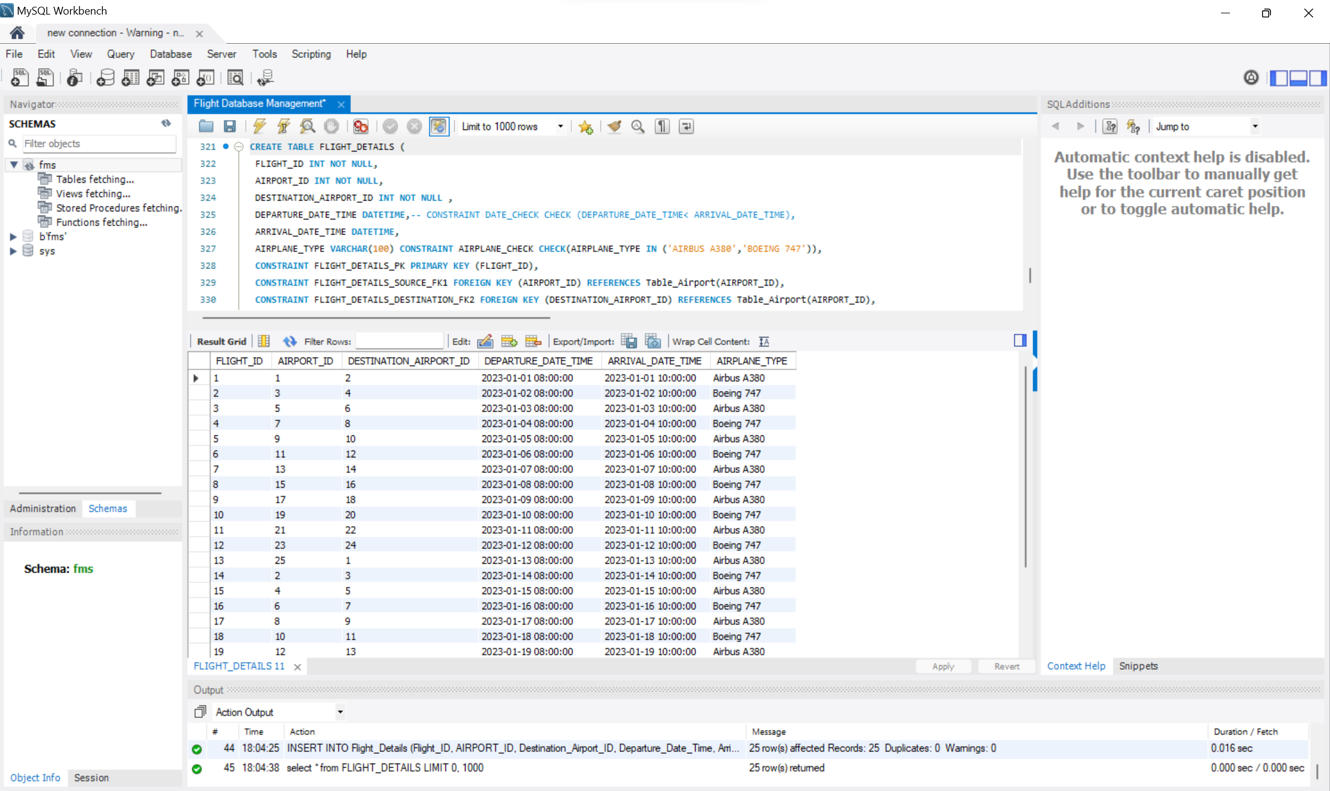
(22, 18, 19, '2023-01-22 08:00:00', '2023-01-22 10:00:00', 'Boeing 747'),

(23, 20, 21, '2023-01-23 08:00:00', '2023-01-23 10:00:00', 'Airbus A380'),

(24, 22, 23, '2023-01-24 08:00:00', '2023-01-24 10:00:00', 'Boeing 747'),

(25, 24, 25, '2023-01-25 08:00:00', '2023-01-25 10:00:00', 'Airbus A380');

**RESULT:**



**Entity Generation and Data Entry for Table Seat Details:**

**Statements Explanation:**

* Firstly, the "SEAT\_DETAILS" table has been created to effectively manage seat information for flights. It incorporates key columns such as **SEAT\_ID**, **TRAVEL\_CLASS\_ID**, and **FLIGHT\_ID** to comprehensively capture details about each seat.
* Secondly, to ensure data integrity, the table enforces a primary key constraint (**SEAT\_DETAILS\_PK**) on the **SEAT\_ID** column, and foreign key constraints (**SEAT\_DETAILS\_TRAVELCLASSID\_FK1** and **SEAT\_DETAILS\_FLIGHTID\_FK2**) establish relationships with other pertinent tables, namely "TRAVEL\_CLASS" and "FLIGHT\_DETAILS."
* Moving on, the column descriptions provide clarity on the purpose of each field, with **SEAT\_ID** serving as a unique identifier, **TRAVEL\_CLASS\_ID** denoting the associated travel class, and **FLIGHT\_ID** indicating the flight to which the seat is assigned. Lastly, the data insertion process involves populating the table with diverse seat assignments across various travel classes and flights, exemplified through the use of **INSERT INTO** statements.
* This approach ensures that the "SEAT\_DETAILS" table is well-structured and capable of accommodating a range of seating scenarios in the airline reservation system.

**QUERY:** Seat Details

CREATE TABLE SEAT\_DETAILS  
(  
 SEAT\_ID VARCHAR(100) NOT NULL,  
 TRAVEL\_CLASS\_ID INT NOT NULL,   
FLIGHT\_ID INT NOT NULL,  
 CONSTRAINT SEAT\_DETAILS\_PK PRIMARY KEY (SEAT\_ID),  
 CONSTRAINT SEAT\_DETAILS\_TRAVELCLASSID\_FK1 FOREIGN KEY (TRAVEL\_CLASS\_ID) REFERENCES TRAVEL\_CLASS(TRAVEL\_CLASS\_ID),  
 CONSTRAINT SEAT\_DETAILS\_FLIGHTID\_FK2 FOREIGN KEY (FLIGHT\_ID) REFERENCES FLIGHT\_DETAILS(FLIGHT\_ID)  
)

**INSERT STATEMENTS:**

INSERT INTO SEAT\_DETAILS (SEAT\_ID, TRAVEL\_CLASS\_ID, FLIGHT\_ID)

VALUES

('A1', 1, 1),

('A2', 2, 2),

('A3', 3, 3),

('B1', 4, 4),

('B2', 5, 5),

('B3', 1, 6),

('C1', 2, 7),

('C2', 3, 8),

('C3', 4, 9),

('D1', 5, 10),

('D2', 1, 11),

('D3', 2, 12),

('E1', 3, 13),

('E2', 4, 14),

('E3', 5, 15),

('F1', 1, 16),

('F2', 2, 17),

('F3', 3, 18),

('G1', 4, 19),

('G2', 5, 20),

('G3', 1, 21),

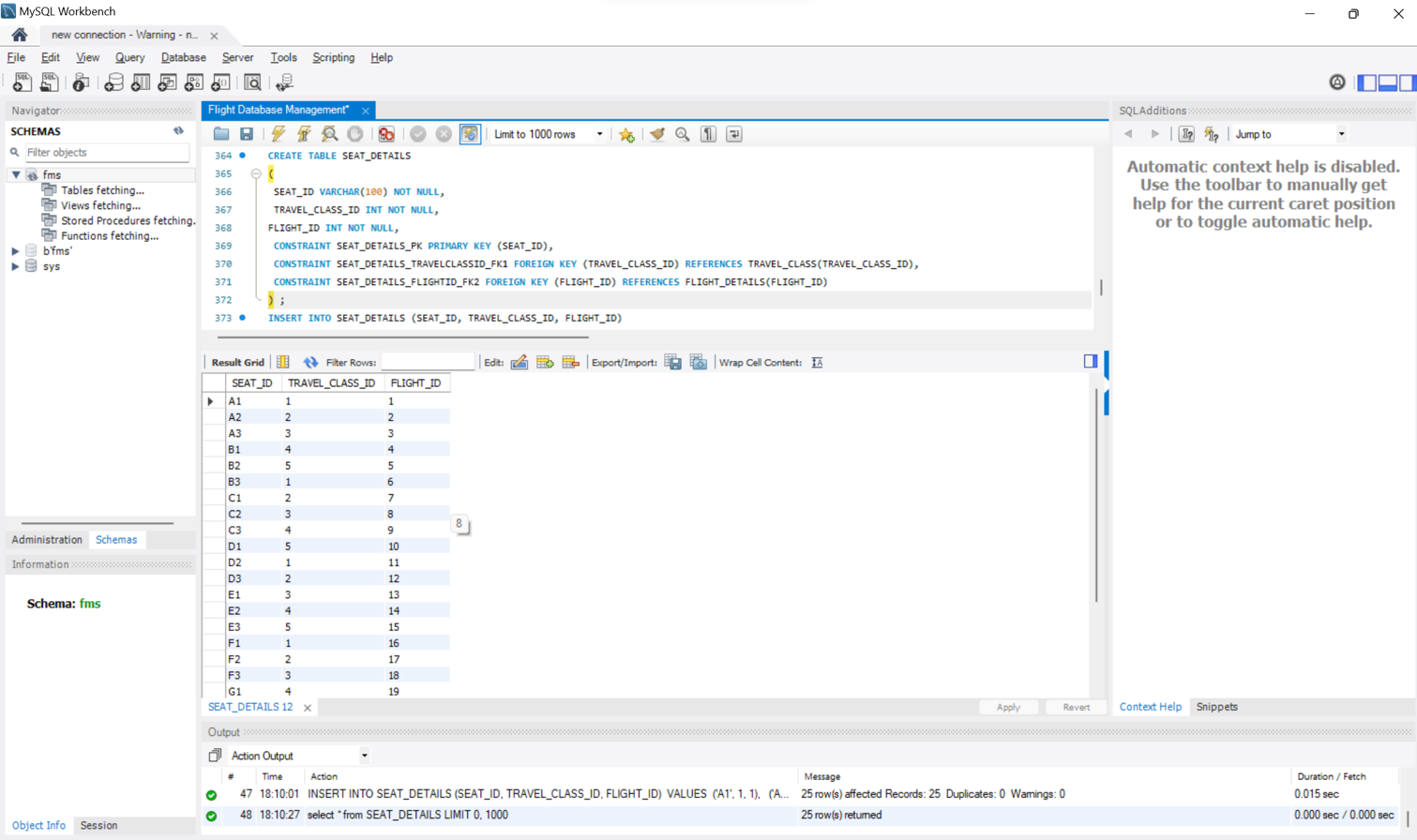
('H1', 2, 22),

('H2', 3, 23),

('H3', 4, 24),

('I1', 5, 25);

**RESULT:**



**Entity Generation and Data Entry for Table Reservation:**

**Statements Explanation:**

* Firstly, the "RESERVATION" table has been created with key columns including RESERVATION\_ID, PASSENGER\_ID, SEAT\_ID, and DATE\_OF\_RESERVATION to effectively manage reservation data.
* Secondly, crucial constraints such as RESERVATION\_PK as the primary key and foreign key constraints (RESERVATION\_PASSENGER\_ID\_FK1 and RESERVATION\_SEAT\_ID\_FK2) have been implemented to ensure data integrity and establish relationships with the "PASSENGER" and "SEAT\_DETAILS" tables, respectively.
* Thirdly, the DATE\_OF\_RESERVATION column is designed to default to the current date using GETDATE() if not explicitly specified during data insertion, ensuring consistent recording of reservation dates.
* Lastly, the data insertion process populates the table with diverse reservation records, capturing details such as the passenger, assigned seat, and the date of reservation. This comprehensive approach ensures a well-structured "RESERVATION" table that efficiently stores and manages reservation information in the airline reservation system.

**QUERY:** **TABLE Reservation**

CREATE TABLE RESERVATION (  
 RESERVATION\_ID INT NOT NULL,  
 PASSENGER\_ID INT NOT NULL,  
 SEAT\_ID VARCHAR(100) NOT NULL,  
 --DEFAULT VALUE RECORDED BELOW  
 DATE\_OF\_RESERVATION DATE DEFAULT(GETDATE()),  
 CONSTRAINT RESERVATION\_PK PRIMARY KEY (RESERVATION\_ID),  
 CONSTRAINT RESERVATION\_PASSENGER\_ID\_FK1 FOREIGN KEY (PASSENGER\_ID) REFERENCES PASSENGER(PASSENGER\_ID),  
 CONSTRAINT RESERVATION\_SEAT\_ID\_FK2 FOREIGN KEY (SEAT\_ID) REFERENCES SEAT\_DETAILS(SEAT\_ID)  
 );

**INSERT STATEMENTS:**

INSERT INTO Reservation (Reservation\_ID, Passenger\_ID, Seat\_ID, DATE\_OF\_RESERVATION)

VALUES

(1, 1, 'A1','2023-01-01'),

(2, 2, 'A2','2023-03-02'),

(3, 3, 'A3','2023-07-03'),

(4, 4, 'B1','2023-08-04'),

(5, 5, 'B2','2023-09-05'),

(6, 6, 'B3','2023-01-06'),

(7, 7, 'C1','2023-03-07'),

(8, 8, 'C2','2023-07-08'),

(9, 9, 'C3','2023-08-09'),

(10, 10, 'D1','2023-08-10'),

(11, 11, 'D2','2023-09-11'),

(12, 12, 'D3','2023-10-12'),

(13, 13, 'E1','2023-01-13'),

(14, 14, 'E2','2023-10-14'),

(15, 15, 'E3','2023-11-15'),

(16, 16, 'F1','2023-12-16'),

(17, 17, 'F2','2023-03-17'),

(18, 18, 'F3','2023-12-18'),

(19, 19, 'G1','2023-05-19'),

(20, 20, 'G2','2023-04-20'),

(21, 21, 'G3','2023-02-21'),

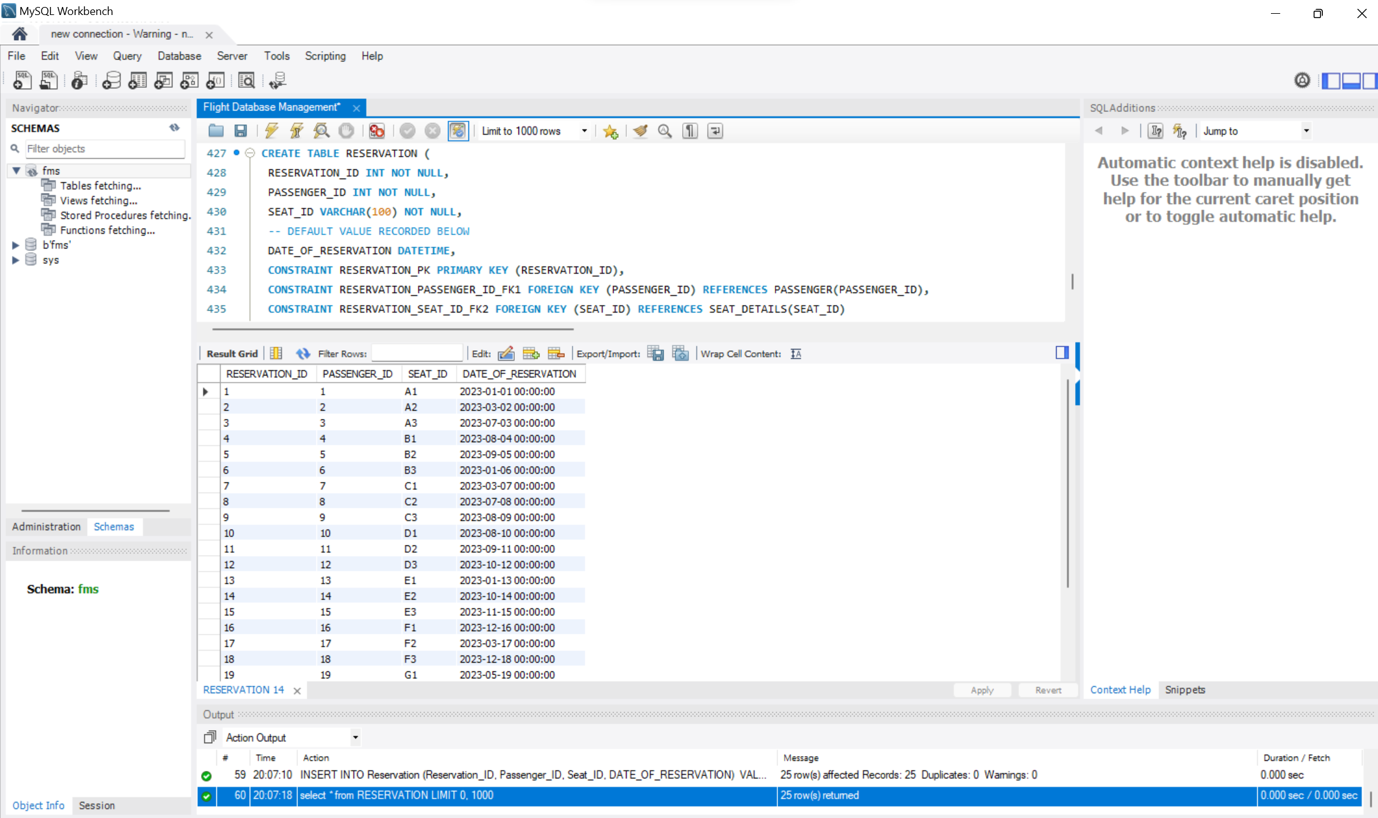
(22, 22, 'H1','2023-06-22'),

(23, 23, 'H2','2023-04-23'),

(24, 24, 'H3','2023-06-24'),

(25, 25, 'I1','2023-02-25');

**RESULT:**



**Entity Generation and Data Entry for Table Payment Status:**

**Statements Explanation:**

* Firstly, the "PAYMENT\_STATUS" table has been created to track payment-related information in the airline reservation system, with key columns including **PAYMENT\_ID**, **PAYMENT\_STATUS\_YN**, **PAYMENT\_DUE\_DATE**, **PAYMENT\_AMOUNT**, and **RESERVATION\_ID**. Secondly, the table design incorporates essential constraints, such as the primary key (**PAYMENT\_ID**) and a foreign key (**PAYMENT\_RESERVATION\_ID\_FK**) linking to the "RESERVATION" table, ensuring data integrity and establishing relationships between payment records and reservations. Thirdly, the column **PAYMENT\_STATUS\_YN** utilizes a check constraint to ensure that only 'Y' or 'N' values are accepted, indicating whether the payment is active or not.
* Lastly, diverse payment records have been inserted into the table, capturing details such as payment status, due date, amount, and the associated reservation ID. This comprehensive approach ensures the "PAYMENT\_STATUS" table effectively manages payment information, supporting accurate and organized tracking of payments within the reservation system.

**QUERY:** TABLE Payment\_Status

CREATE TABLE PAYMENT\_STATUS (

PAYMENT\_ID INT NOT NULL AUTO\_INCREMENT,

PAYMENT\_STATUS\_YN CHAR(1) CHECK(PAYMENT\_STATUS\_YN IN ('Y','N')),

PAYMENT\_DUE\_DATE DATE,

PAYMENT\_AMOUNT INT,

RESERVATION\_ID INT NOT NULL,

PRIMARY KEY (PAYMENT\_ID),

CONSTRAINT PAYMENT\_RESERVATION\_ID\_FK FOREIGN KEY (RESERVATION\_ID) REFERENCES RESERVATION(RESERVATION\_ID)

);

**INSERT STATEMENTS:**

INSERT INTO Payment\_Status (Payment\_Status\_YN, Payment\_Due\_Date, Payment\_Amount, Reservation\_ID)

VALUES

('Y', '2023-01-01', 100, 1),

('N', '2023-03-02', 150, 2),

('Y', '2023-07-03', 120, 3),

('N', '2023-08-04', 200, 4),

('Y', '2023-09-05', 180, 5),

('N', '2023-01-06', 90, 6),

('Y', '2023-03-07', 140, 7),

('N', '2023-07-08', 210, 8),

('Y', '2023-08-09', 170, 9),

('N', '2023-08-10', 110, 10),

('Y', '2023-09-11', 130, 11),

('N', '2023-10-12', 190, 12),

('Y', '2023-01-13', 220, 13),

('N', '2023-10-14', 200, 14),

('Y', '2023-11-15', 160, 15),

('N', '2023-12-16', 240, 16),

('Y', '2023-03-17', 130, 17),

('N', '2023-12-18', 180, 18),

('Y', '2023-05-19', 270, 19),

('N', '2023-04-20', 120, 20),

('Y', '2023-02-21', 150, 21),

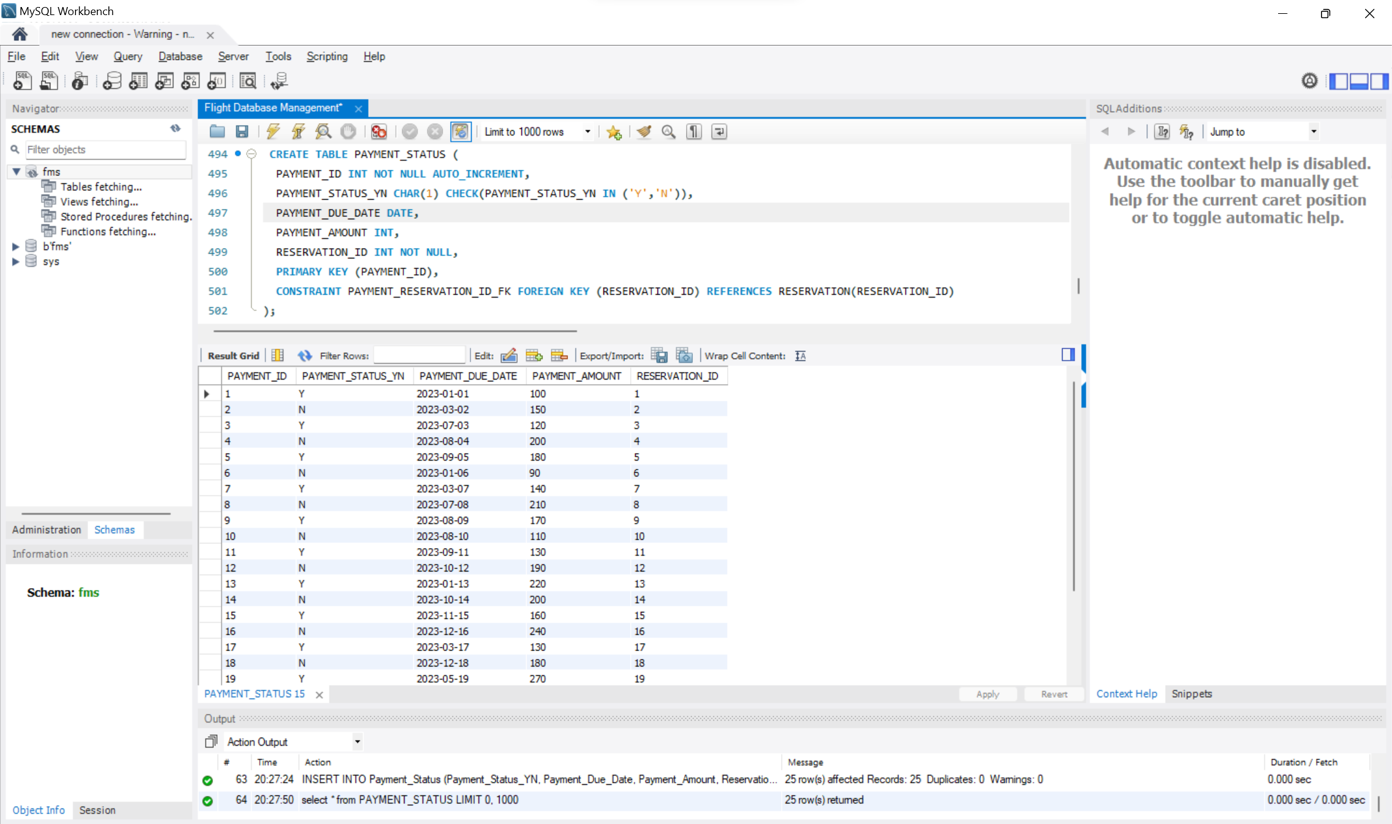
('N', '2023-06-22', 200, 22),

('Y', '2023-04-23', 280, 23),

('N', '2023-06-24', 110, 24),

('Y', '2023-02-25', 140, 25);

**RESULT:**



**Entity Generation and Data Entry for Table Service Offering:**

**Statements Explanation:**

**QUERY:** **SERVICE\_OFFERING**

CREATE TABLE SERVICE\_OFFERING

(

TRAVEL\_CLASS\_ID INT NOT NULL,

SERVICE\_ID INT NOT NULL,

OFFERED\_YN CHAR(1) CONSTRAINT CHECK\_CHARACTER\_OFFERED\_YN CHECK(OFFERED\_YN IN ('Y','N')),

FROM\_MONTH VARCHAR(20),

TO\_MONTH VARCHAR(20),

CONSTRAINT SERVICE\_OFFERING\_TCI\_FK1 FOREIGN KEY (TRAVEL\_CLASS\_ID) REFERENCES TRAVEL\_CLASS(TRAVEL\_CLASS\_ID),

CONSTRAINT SERVICE\_OFFERING\_SID\_FK2 FOREIGN KEY (SERVICE\_ID) REFERENCES FLIGHT\_SERVICE(SERVICE\_ID),

CONSTRAINT SERVICE\_OFFERING\_PK PRIMARY KEY (TRAVEL\_CLASS\_ID,SERVICE\_ID)

);

**INSERT STATEMENTS:**

INSERT INTO SERVICE\_OFFERING (TRAVEL\_CLASS\_ID, SERVICE\_ID, OFFERED\_YN, FROM\_MONTH, TO\_MONTH)

VALUES

(1, 1, 'Y', 'January', 'December'),

(2, 2, 'N', 'February', 'November'),

(3, 3, 'Y', 'March', 'October'),

(4, 4, 'N', 'April', 'September'),

(5, 5, 'Y', 'May', 'August'),

(6, 6, 'Y', 'June', 'July'),

(7, 7, 'N', 'July', 'June'),

(8, 8, 'Y', 'August', 'May'),

(9, 9, 'N', 'September', 'April'),

(10, 10, 'Y', 'October', 'March'),

(11, 11, 'Y', 'November', 'February'),

(12, 12, 'N', 'December', 'January'),

(13, 13, 'Y', 'January', 'December'),

(14, 14, 'N', 'February', 'November'),

(15, 15, 'Y', 'March', 'October'),

(16, 16, 'Y', 'April', 'September'),

(17, 17, 'N', 'May', 'August'),

(18, 18, 'Y', 'June', 'July'),

(19, 19, 'N', 'July', 'June'),

(20, 20, 'Y', 'August', 'May'),

(21, 21, 'Y', 'September', 'April'),

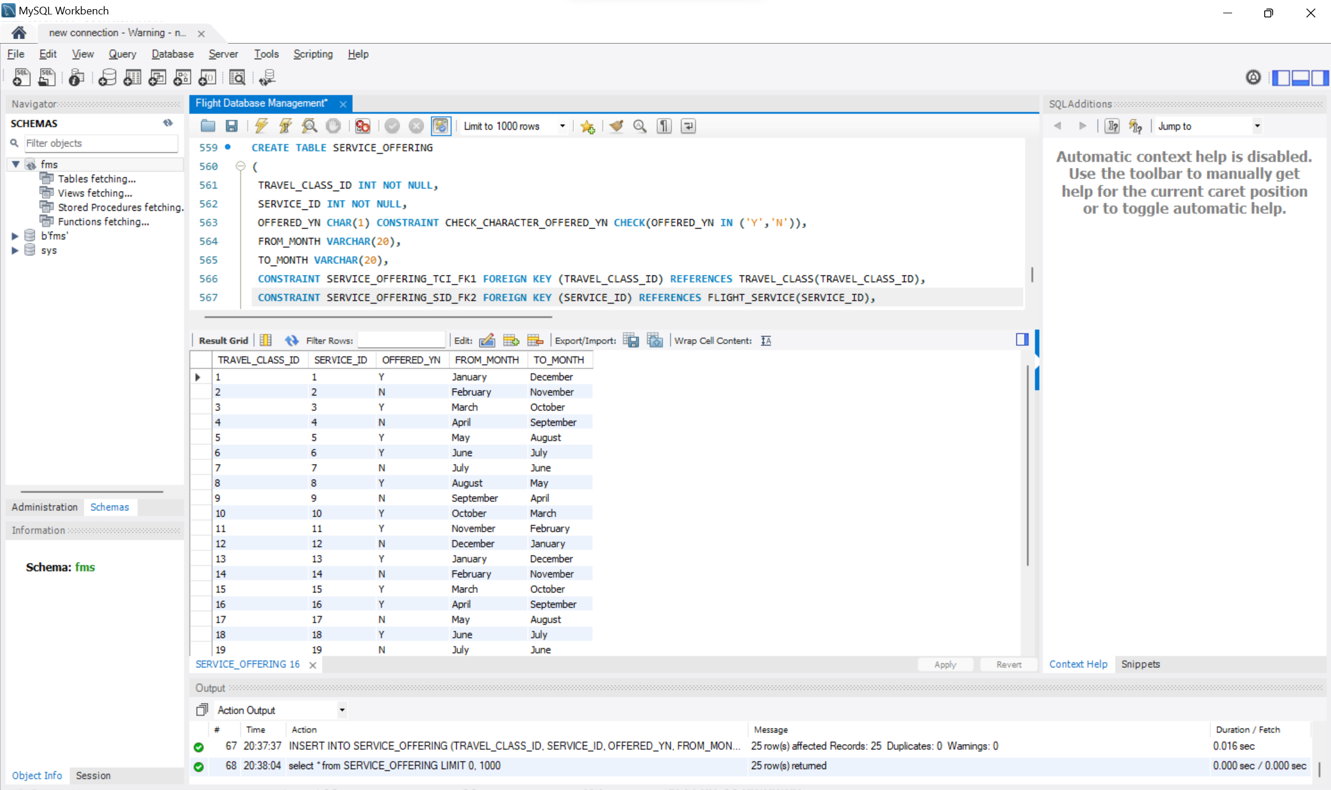
(22, 22, 'N', 'October', 'March'),

(23, 23, 'Y', 'November', 'February'),

(24, 24, 'N', 'December', 'January'),

(25, 25, 'Y', 'January', 'December');

**RESULT:**



**II. Data Retrieval and Reports:**

**QUESTION 1: LIST ALL AIRPORTS IN A SPECIFIC COUNTRY?**

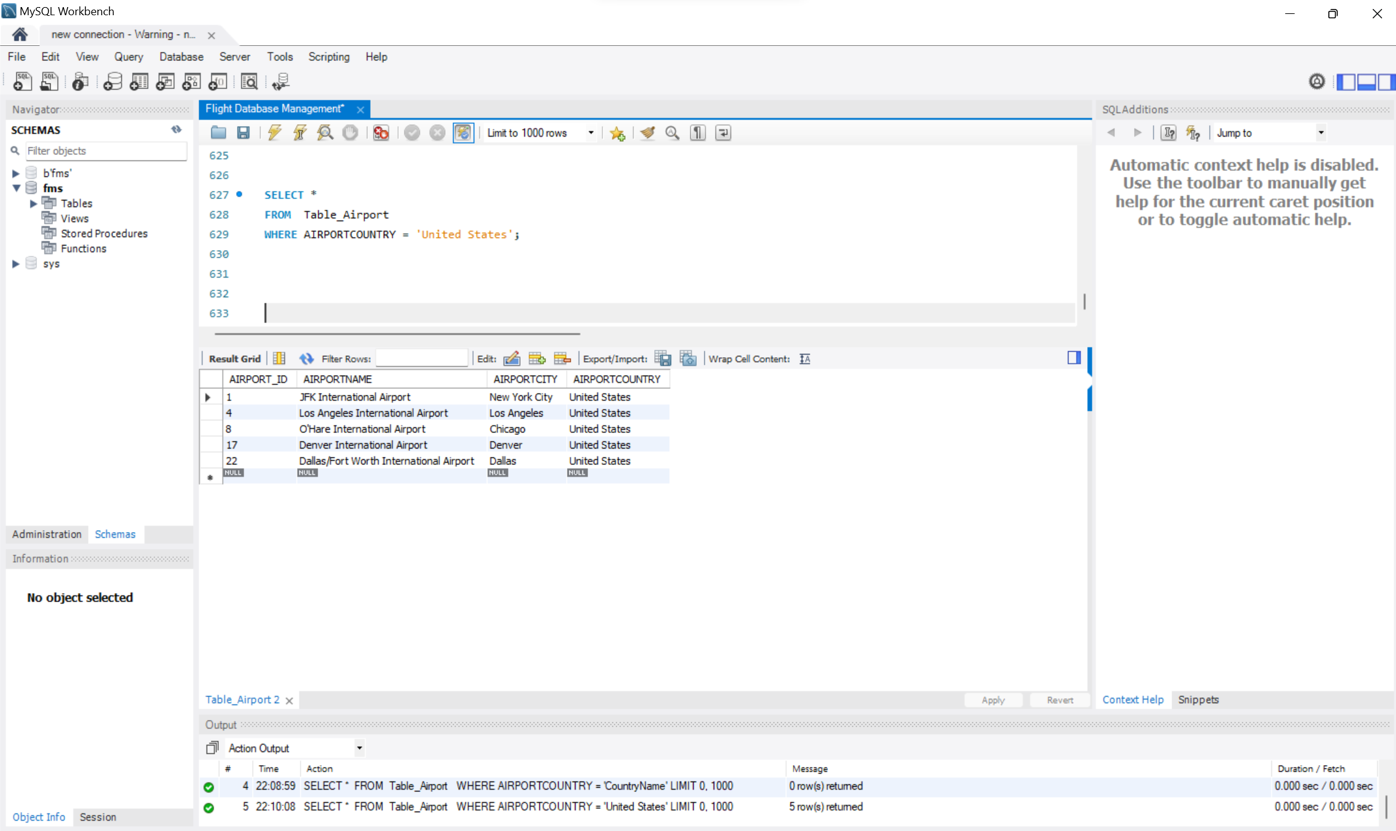
**QUERY:**

SELECT \*

FROM AIRPORT

WHERE Country = 'United States';

**RESULT:**



**QUESTION 2. RETRIEVE THE PAYMENT STATUS FOR RESERVATIONS MADE BY 'ALICE JOHNSON'?**

**QUERY:**

SELECT RS.RESERVATION\_ID, PS.PAYMENT\_STATUS\_YN, PS.PAYMENT\_DUE\_DATE, PS.PAYMENT\_AMOUNT

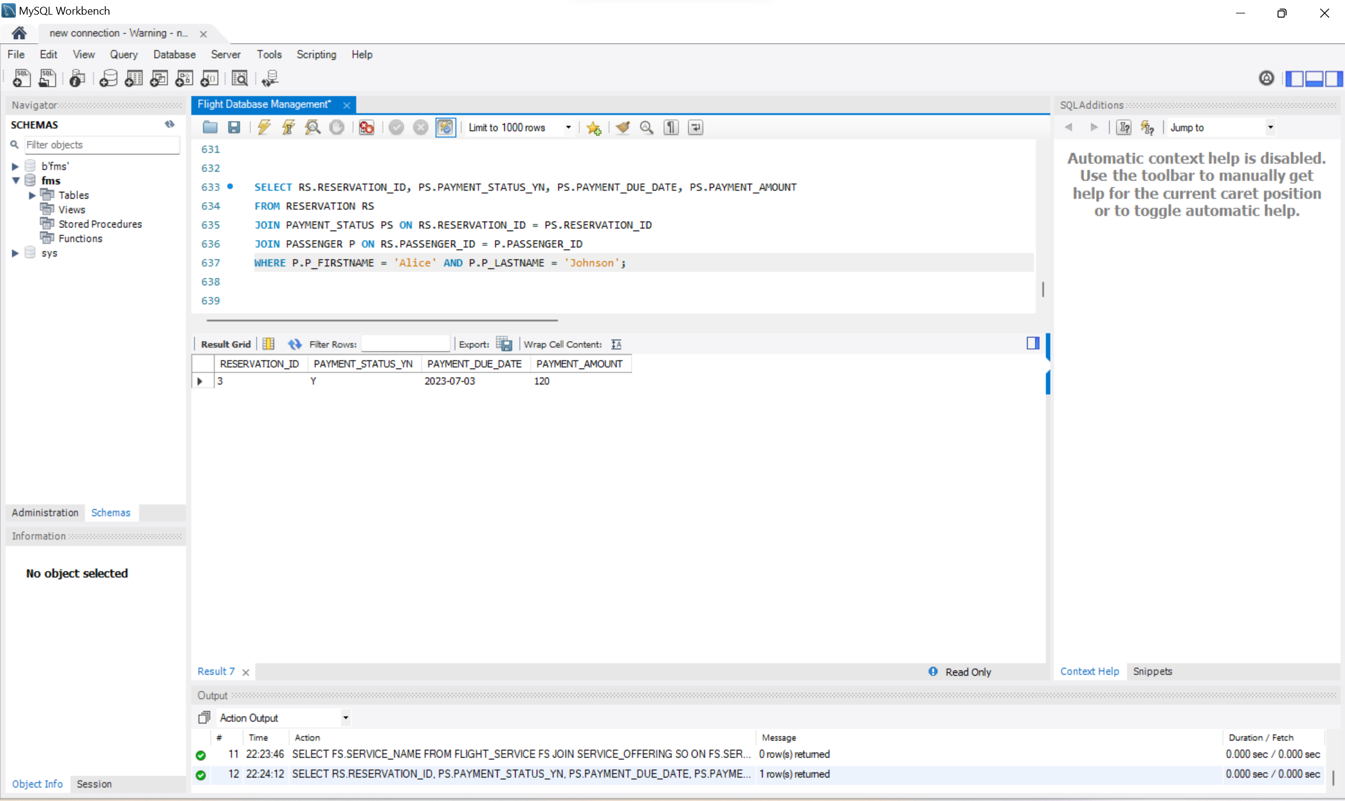
FROM RESERVATION RS

JOIN PAYMENT\_STATUS PS ON RS.RESERVATION\_ID = PS.RESERVATION\_ID

JOIN PASSENGER P ON RS.PASSENGER\_ID = P.PASSENGER\_ID

WHERE P.P\_FIRSTNAME = 'Alice' AND P.P\_LASTNAME = 'Johnson';

**RESULT:**



**QUESTION 3. DISPLAY THE TOTAL NUMBER OF PASSENGERS ON A SPECIFIC FLIGHT?**

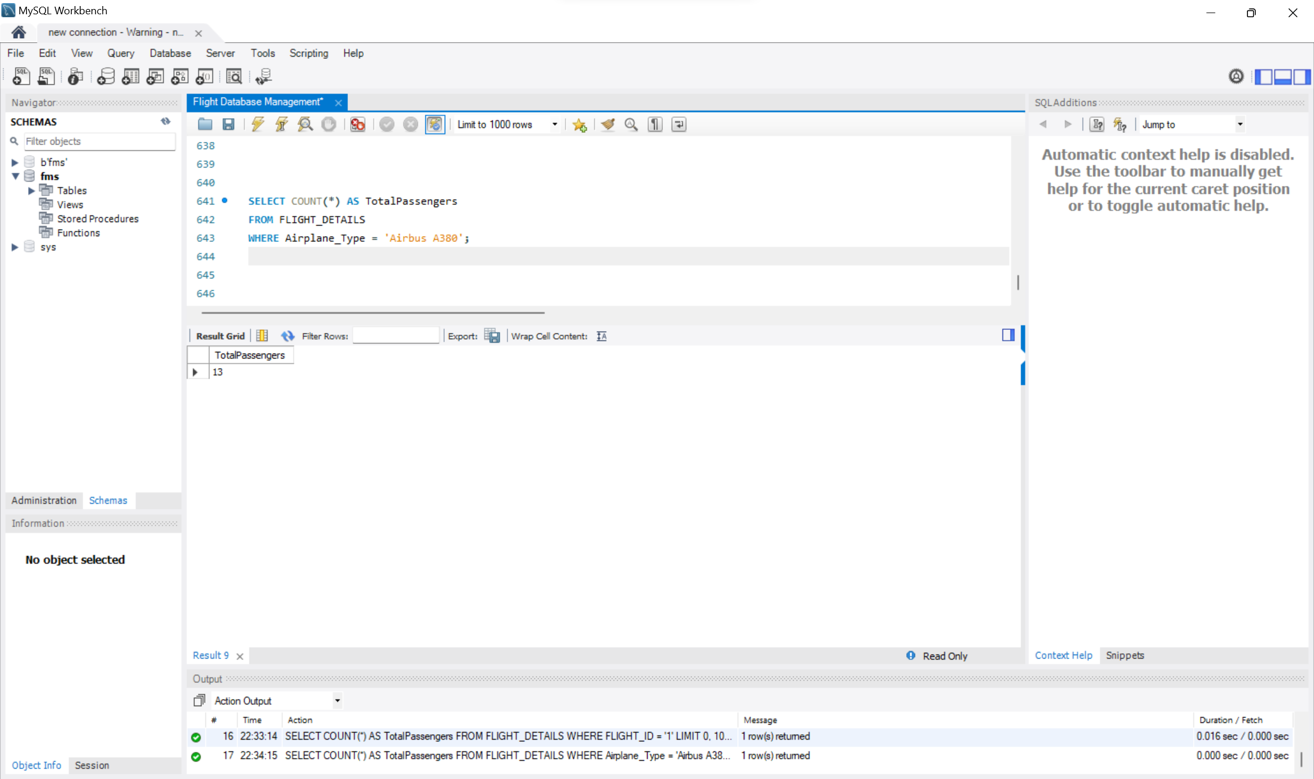
**QUERY:**

SELECT COUNT(\*) AS TotalPassengers

FROM FLIGHT\_DETAILS

WHERE Airplane\_Type = 'Airbus A380';

**RESULT:**



QUESTION 4. **RETRIEVE THE COUNT OF AVAILABLE SEATS IN EACH TRAVEL CLASS?**

**QUERY:**

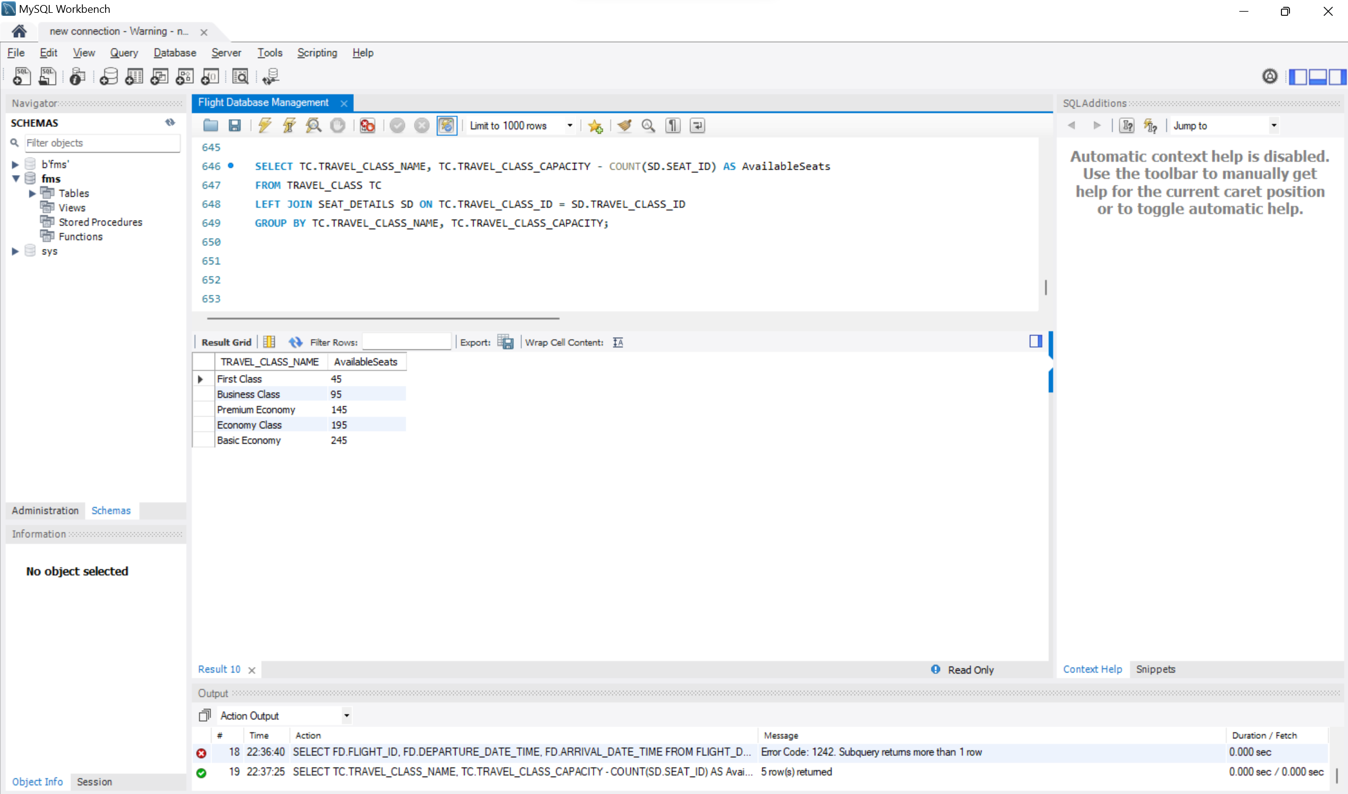
SELECT TC.TRAVEL\_CLASS\_NAME, TC.TRAVEL\_CLASS\_CAPACITY - COUNT(SD.SEAT\_ID) AS AvailableSeats

FROM TRAVEL\_CLASS TC

LEFT JOIN SEAT\_DETAILS SD ON TC.TRAVEL\_CLASS\_ID = SD.TRAVEL\_CLASS\_ID

GROUP BY TC.TRAVEL\_CLASS\_NAME, TC.TRAVEL\_CLASS\_CAPACITY;

**RESULT:**



**QUESTION 5. RETRIEVE FLIGHT DETAILS ALONG WITH THE ASSOCIATED AIRPORT NAMES?**

**QUERY:**

SELECT F.FLIGHT\_ID, F.DEPARTURE\_DATE\_TIME, F.ARRIVAL\_DATE\_TIME, F.AIRPLANE\_TYPE,

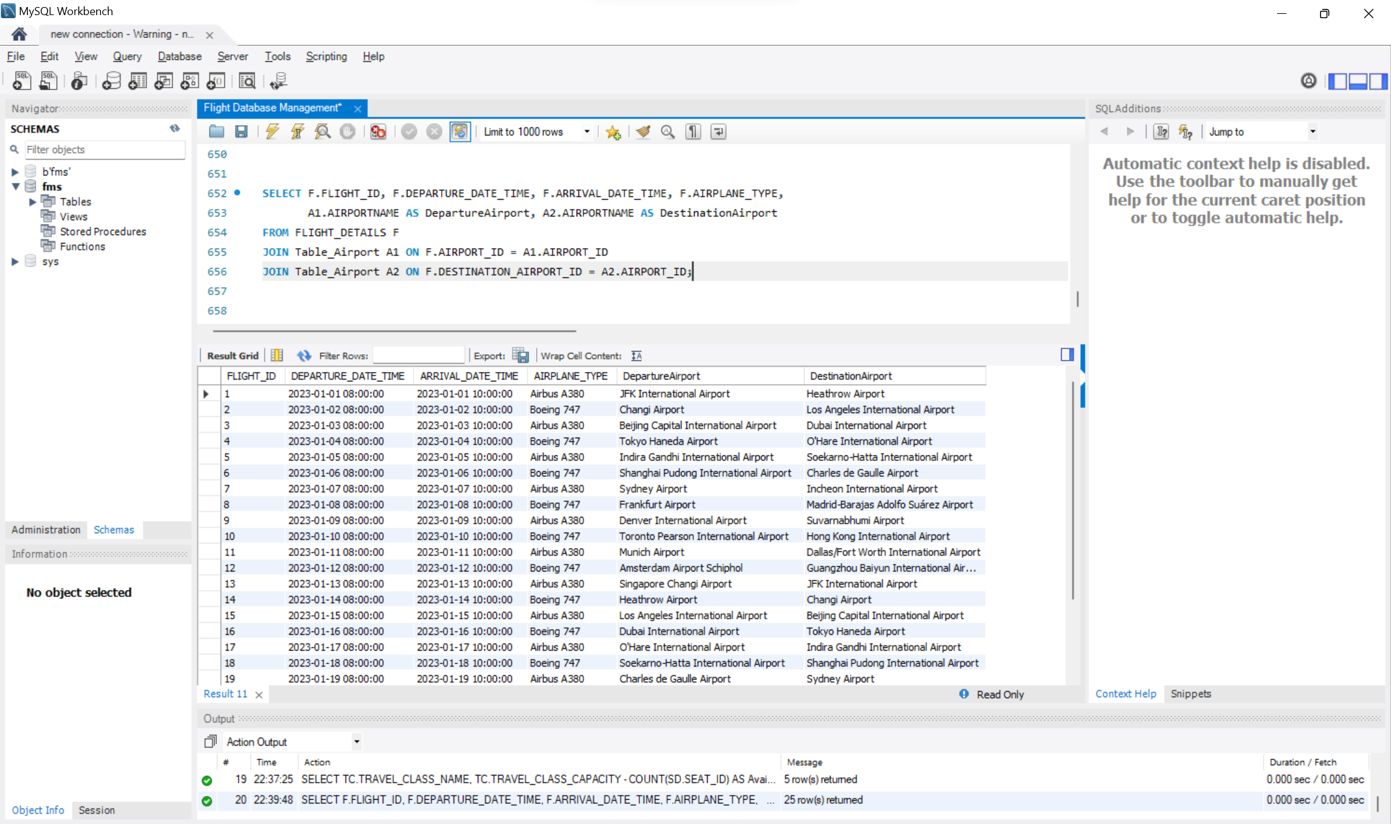
A1.AIRPORTNAME AS DepartureAirport, A2.AIRPORTNAME AS DestinationAirport

FROM FLIGHT\_DETAILS F

JOIN Table\_Airport A1 ON F.AIRPORT\_ID = A1.AIRPORT\_ID

JOIN Table\_Airport A2 ON F.DESTINATION\_AIRPORT\_ID = A2.AIRPORT\_ID;

**RESULT:**



**QUESTION 6. LIST ALL AVAILABLE SERVICES OFFERED IN FIRST CLASS?**

**QUERY:**

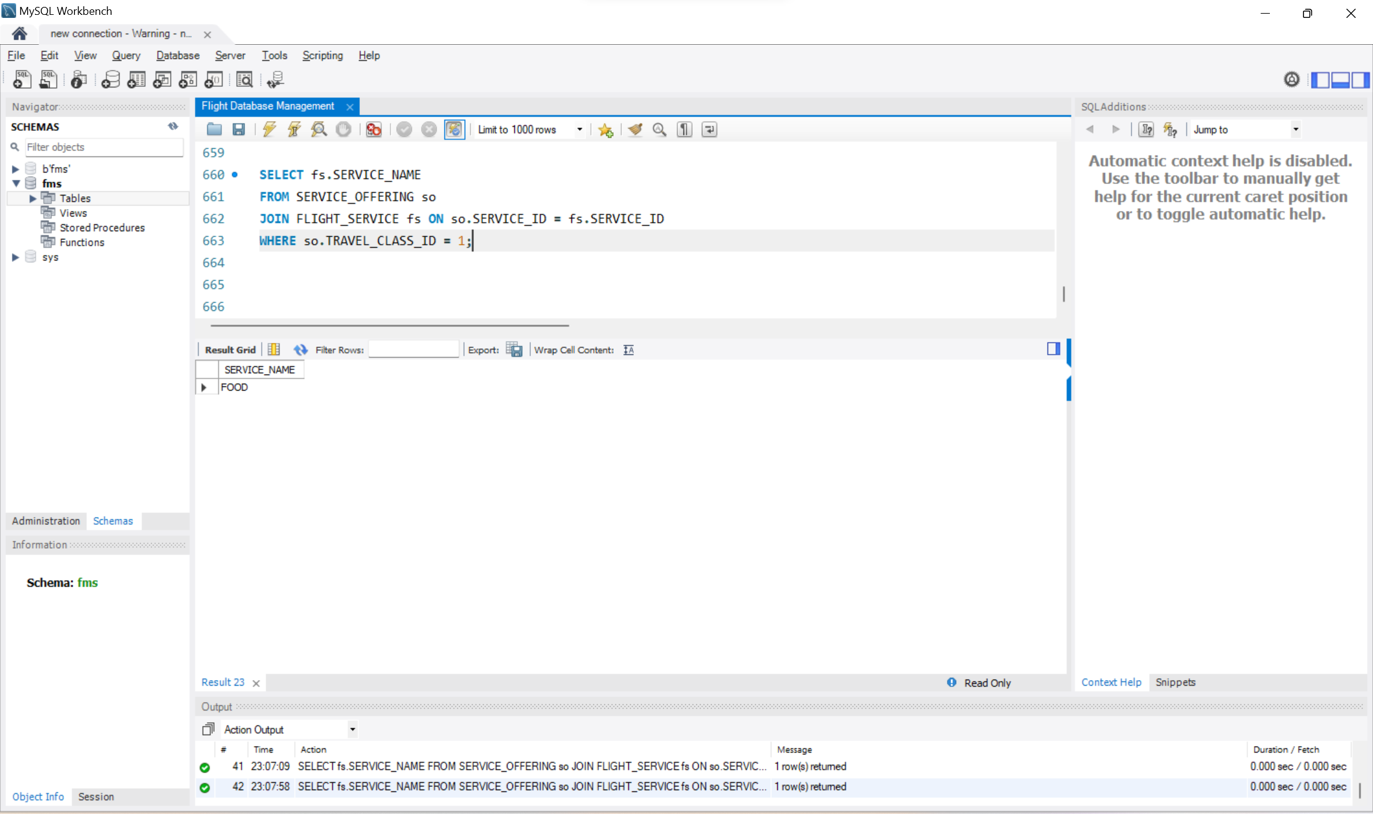
SELECT fs.SERVICE\_NAME

FROM SERVICE\_OFFERING so

JOIN FLIGHT\_SERVICE fs ON so.SERVICE\_ID = fs.SERVICE\_ID

WHERE so.TRAVEL\_CLASS\_ID = 1;

**RESULT:**



**CONCLUSION:**

The Flight Booking Database System transforms airline operations by prioritizing passengers first while ensuring data integrity. It promises streamlined reservations, personalized services, and enhanced data security through meticulous design and robust functionality. This system represents a significant step forward toward a more seamless and customer-focused approach, reshaping the future of air travel.

—--------THANK YOU—-----