



School Of Engineering Second Year CSE (Hons) Assignment in Big Data and Database Systems

Assignment Title

LSBU (London South Bank University) Cruise Liners Case Study analysed and design a data model using MS SQL Server and Tableau.

Student ID: 4120846

Submission Date

17.00 Thursday 18 April 2024

Module Leader

George Ubakanma

Table of contents:

1. Introduction.....	4
1.1. Coursework Overview.....	4
1.2. Project aim and Objectives.....	4
1.3. Scope of the projects.....	4
2. Case study Overview.....	4
2.1. Background Information.....	5
2.2. LSBU Cruise Liners Case study description.....	5
3. Database Design.....	5
3.1. Entity-relationship Diagram.....	5
3.1.1. ERD overview.....	5
3.1.2. Description of Entities and Relationships.....	6-7
3.2. Normalization.....	8
3.2.1. Functional Dependencies.....	8
3.2.2. Normal Forms Justifications.....	9
4. Database Implementation.....	9-10
4.1.1. Passenger Table.....	11-12
4.1.2. Voyage Table.....	12-13
4.1.3. Ship Table.....	14
4.1.4. Departure Table.....	15
4.1.5. Reservation Table.....	16-17
4.1.6. Employee Table.....	18
4.1.7. Seaport Table.....	19-20
4.1.8. OnBoardEnterTainment Table.....	20-21
5. SQL Queries and Procedures.....	21
5.1. Query for Reservation totals.....	21

5.2. Query for Available Cabins.....	22
5.3. Stored procedure for total revenue generated by On-Board Entertainment and Activity sales.....	23
5.4. Trigger for preventing double bookings.....	23-26
5.5. Stored procedure for reservation invoice.....	26-27
6. Dashboard Development.....	28
6.1. Design Consideration.....	28
6.2. Implementation in Tableau.....	29-33
7. Video Demonstration.....	33
7.1. Overview of the video content.....	33
7.2. Execution of SQL tasks.....	33
7.3. Dashboard showcase.....	34
8. Conclusion.....	34
8.1. Project summary.....	34
8.2. Key learnings and outcomes.....	34
9. References.....	35

1. Introduction.

1.1. Coursework Overview.

This coursework is part of the Big Data & Database Systems module, which constitutes a huge portion of the module assessment. The coursework requires us to demonstrate a comprehensive understanding of database design, implementation, and analysis. By focusing on a specific case study - LSBU Cruise Liners - My task with developing a relational database prototype, implementing SQL queries and stored procedures, and visualizing data through a dashboard.

1.2. Project aim and Objectives.

The primary aim of this project is to apply theoretical knowledge of database systems in a practical scenario, enhancing learning through real-world applications. The objectives of the project include:

- To design an efficient database schema based on the LSBU Cruise Liners case study, ensuring data integrity and normalization.
- To implement the designed schema in MS SQL Server, including the creation of tables and insertion of test data.
- To develop SQL queries and stored procedures addressing specific requirements, such as calculating reservation totals, identifying available cabins, generating revenue reports, preventing double bookings, and producing detailed reservation invoices.

1.3. Scope of the Project.

- Database Design and Normalization.
- SQL implementation.
- Data Visualization.
- Documentation and Demonstration.

2. Case Study Overview.

2.1. Background Information.

The Big Data & Database Systems coursework is designed in a practical case study approach, allowing me to apply database management concepts to real-world scenarios. The chosen case study for this project revolves around the cruise line industry, a sector that is data-intensive and offers numerous opportunities for optimizing operations, enhancing customer experiences, and improving business outcomes through data analysis and database management.

2.2. LSBU Cruise Liners Case Study Description.

LSBU Cruise Liners, a fictional low-cost cruise ship operator, provides the context for my case study. Operating mainly within the European Union, LSBU Cruise Liners have felt that an area of concern with their current system—using part manual and part computer-based reservation information—has produced inefficiency, for example, double bookings, and does not cater well for offering on-board entertainment and activity profiling. The following case study contains an outline of operations of the company from management, reservations, passenger details, and voyage schedules to planning activities and on-board entertainment.

Sailing out from several European seasons. Each will have its data points like seaport code, city, and country to which it associates, and it will have to link all of them to have a single route to and from different seaports. For my main task, I did the design for the relational database system of the complexities of LSBU Cruise Liners operations and implemented the same in a more effective manner. These include an entirely new, lean process of making the reservations, looking with respect to the management of onboard entertainment and activities, avoiding double bookings, and lastly, producing detailed invoices for the booking.

Moreover, the project details the development of a dashboard to visualize important business metrics. Therefore, this will help LSBU Cruise Liners in providing intelligent decisions while serving their clients.

3. Database design.

3.1. Entity-Relationship Diagram.

3.1.1. ERD Overview.

ERD of LSBU Cruise Liners presents an efficient model for the required data in terms of cruise liner management operations, reservations, voyages, passengers, and entertainment on board. It defines the entities in the database, their attributes, and the relationships among them. It correctly identifies primary and foreign keys to support data integrity, hence allowing space for relational functionalities within the database.

3.1.2. Description of Entities and Relationship.

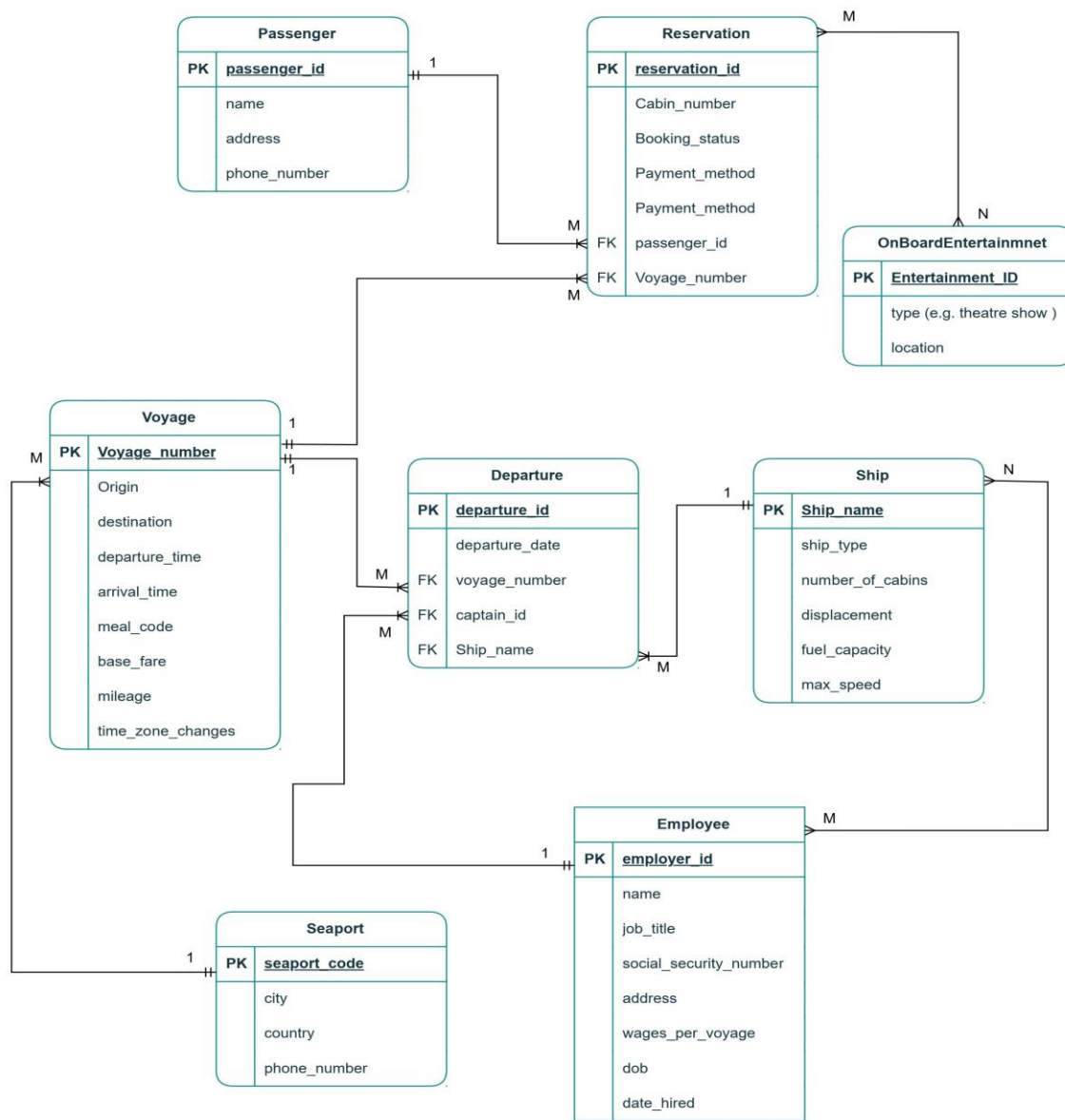


Figure 1.00: Entity Relationship Diagram for LSBU Cruise Liners.

1. Passenger:

Contains personal information of passengers. Attributes include 'passenger_id' (PK), 'name,' 'address,' and 'phone_number.'

2. Reservation.

Manages booking details. Attributes include 'reservation_id' (PK), 'passenger_id' (FK), 'voyage_number' (FK), 'cabin_number,' 'booking_status', and 'payment_method.' The inclusion of both 'passenger_id' and 'voyage_number' as foreign keys establishes relationships to the Passenger and Voyage entities, respectively.

3. Voyage.

Details of cruise voyages. Attributes include 'voyage_number' (PK), 'origin,' 'destination,' 'departure_time,' 'arrival_time,' 'meal_code,' 'base_fare,' 'mileage,' and 'time_zone_changes'.

4. Departure.

Records specific departures of voyages. Attributes include 'departure_id' (PK), 'voyage_number' (FK), 'departure_date,' 'captain_id' (FK), and 'ship_name' (FK). This entity links voyages to specific ships and captains.

5. Ship.

Information on cruise ships. Attributes include 'ship_name' (PK), 'ship_type,' 'number_of_cabins,' 'displacement,' 'fuel_capacity,' and 'max_speed.'

6. Employee.

Employee details, specifically for crew members including captains. Attributes include 'employee_id' (PK), 'name,' 'job_title,' 'social_security_number,' 'address,' 'wages_per_voyage,' 'dob,' and 'date_hired.'

7. Seaport.

Information on seaports where voyages start or end. Attributes include 'seaport_code' (PK), 'city,' 'country,' and 'phone_number.'

8. OnBoardEntertainment.

Manages onboard entertainment options. Attributes include 'entertainment_id' (PK), 'type,' and 'location.'

Relationships.

- Passenger to Reservation: One-to-Many (1:M), indicating a passenger can have multiple reservations.
- Reservation to Voyage: Many-to-One (M:1), reservations are linked to specific voyages.
- Voyage to Departure: One-to-Many (1:M), each voyage can have multiple departures.

- Departure to Ship and Employee. Many-to-One (M:1), linking departures to specific ships and captains.
- Voyage to Seaport (Origin and Destination): Many-to-One (M:1), voyages are associated with specific origin and destination seaports.
- Reservation to OnBoardEntertainment: Many-to-Many (M: N), reservations can include multiple entertainment options, and each option can be included in multiple reservations.

3.2. Normalization.

3.2.1. Functional Dependencies.

Functional dependencies are a fundamental aspect of database normalization that describes the relationship between attributes in a relational database. Here are the functional dependencies for each entity based on the above provided ERD:

1. Passenger.
 - 'passenger_id' → 'name, address, phone_number'
2. Reservation.
 - 'reservation_id' → 'passenger_id, voyage_number, cabin_number, booking_status, payment_method'
3. Voyage.
 - 'voyage_number' → 'origin, destination, departure_time, arrival_time, meal_code, base_fare, mileage, time_zone_changes'
4. Departure.
 - 'departure_id' → 'voyage_number, departure_date, captain_id, ship_name'
5. Ship.
 - 'ship_name' → 'ship_type, number_of_cabins, displacement, fuel_capacity, max_speed'
6. Employee.
 - 'employee_id' → 'name, job_title, social_security_number, address, wages_per_voyage, dob, date_hired'
7. Seaport.
 - 'seaport_code' → 'city, country, phone_number'
8. OnBoardEntertainment.
 - 'entertainment_id' → 'type, location'

3.2.2. Normal Forms Justification.

- 3NF Justification.

An entity is in 3NF if it is in 2NF, and all non-key attributes are not transitively dependent on the primary key.

Based on the functional dependencies listed above, every attribute in each entity is directly dependent on the primary key, satisfying 3NF. There are no transitive dependencies observed in the entities, ensuring that all attributes are only related to the primary key.

- BCNF Justification.

A table is in BCNF if it is in 3NF, and for every one of its non-trivial functional dependencies, $X \rightarrow Y$, X is a superkey. The functional dependencies listed above indicate that for all entities, the left-hand side of the dependency (before the arrow) is always a primary key (a superkey), which directly determines the other attributes in the entity. This adherence ensures the database design is in BCNF as well.

4. Database Implementation.

This section outlines the implementation steps for the database design of the LSBUCruiseLinersDB, focusing on table creation, data insertion, and data verification. Each table is designed to store specific information critical to the operations of LSBU Cruise Liners.as detailed below:

- Establishing a Connection with SQL server.

The initial step entailed launching Microsoft SQL Server Management Studio (SSMS) and establishing a connection to the desired SQL Server instance.

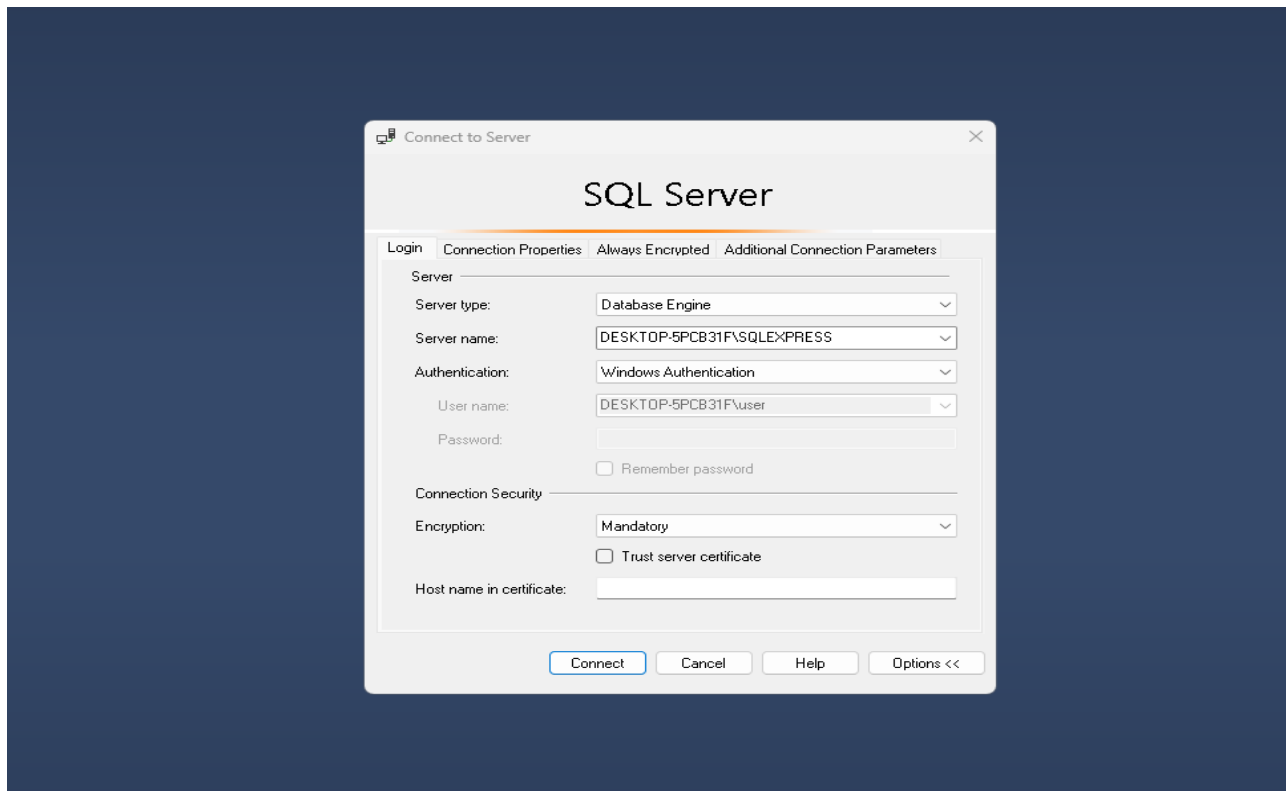


Figure 1.01: Server Connection.

- **Database Creation and Select the Database for use.**

Upon successful connection, the next part is to create a new database that would serve as the repository for the data structures associated with the cruise liner operations. Secondly, With the database successfully created, it is necessary to specify that subsequent operations and table creations were to be executed within the context of this newly established database. This was achieved by executing the USE statement, followed by the GO command to ensure that the command batch was processed correctly.

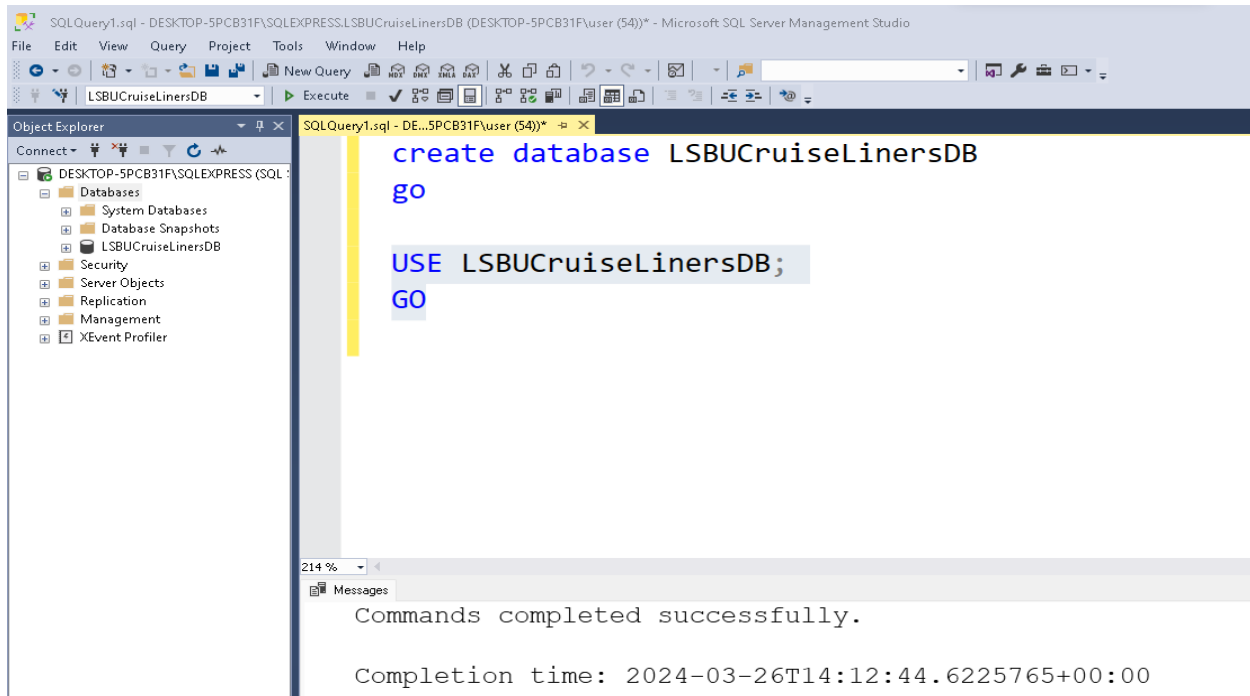


Figure 1.02: Create Database and Use.

4.1.1. Passenger Table.

The passenger table is created to Stores personal information about passengers who book voyages on the cruise liners. It includes identifiers, names, addresses, and contact details, essential for managing bookings and providing personalized services.

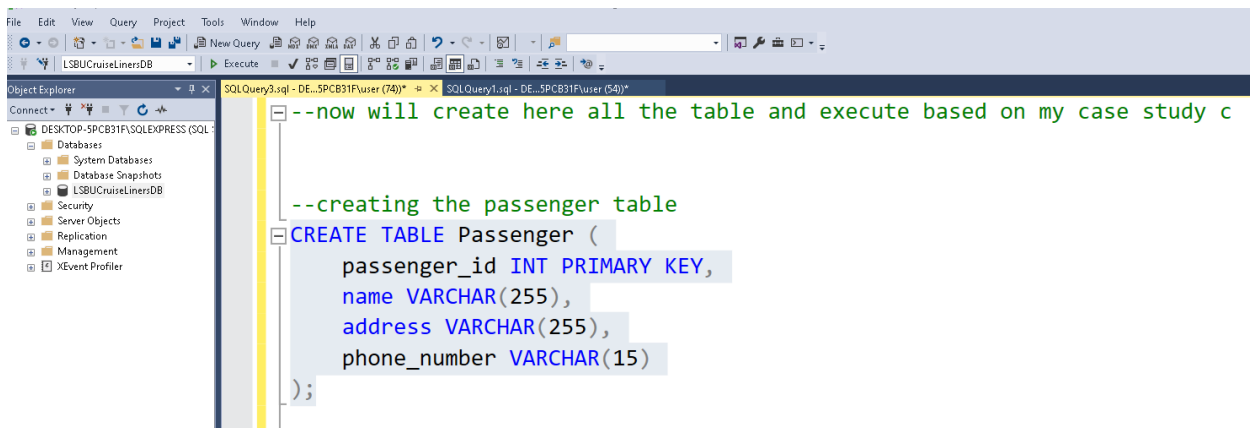


Figure 1.03: Create Passenger Table.

- **Verification.**

20 records were inserted for each table based on the coursework specification and 'SELECT' query was executed to ensure that all records were successfully inserted.

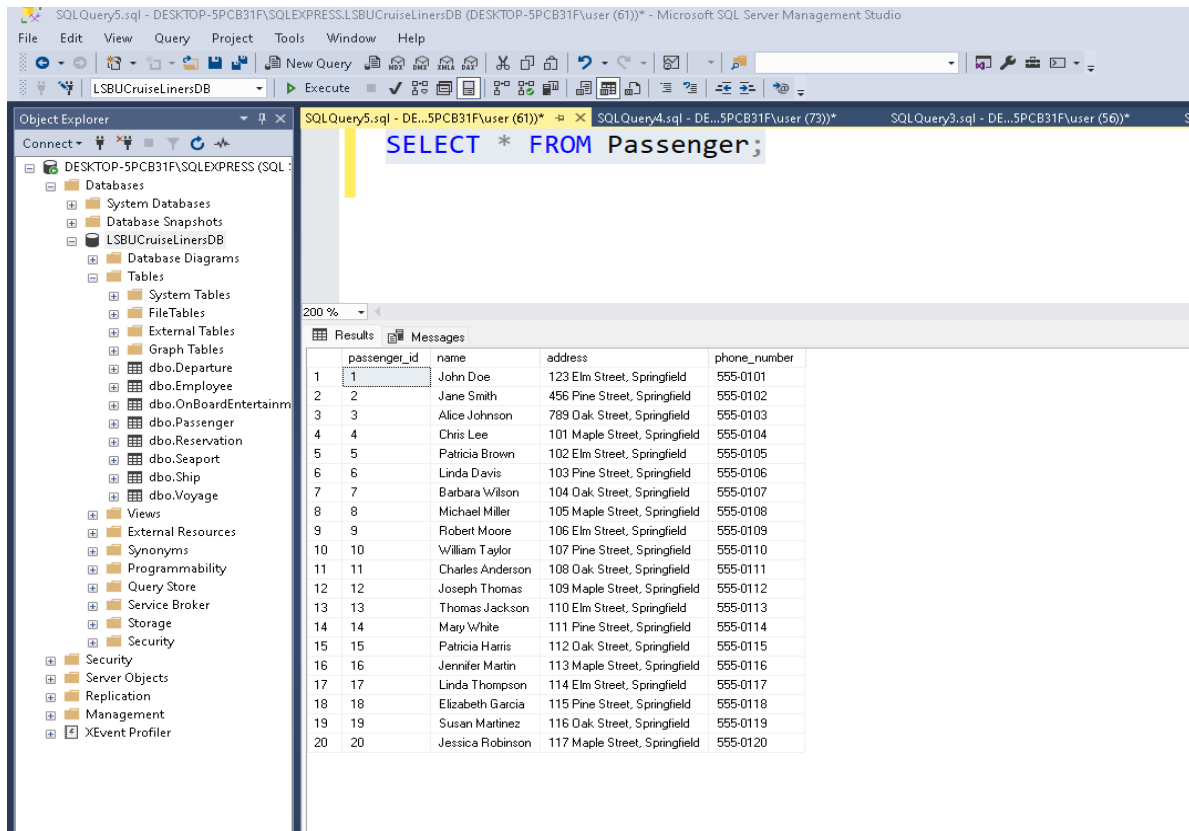


Figure 1.04. Verification of passenger data.

4.1.2. Voyage Table.

The Voyage table is created to Holds details about the cruise voyages offered, including unique voyage numbers, origins, destinations, departure and arrival times, along with pricing and meal options. This table is crucial for scheduling and voyage management.

```
-- Creating the Voyage Table
CREATE TABLE Voyage (
    voyage_number INT PRIMARY KEY,
    origin VARCHAR(100),
    destination VARCHAR(100),
    departure_time DATETIME,
    arrival_time DATETIME,
    meal_code VARCHAR(50),
    base_fare DECIMAL(10, 2),
    mileage INT,
    time_zone_changes INT
);
```

00 %

Messages

Commands completed successfully.

Completion time: 2024-03-26T14:37:45.2099118+00:00

Figure 1.05. Create Voyage Table.

- Verification.

SQLQuery5.sql - DESKTOP-5PCB31F\SQLEXPRESS\LSBUCruiseLinersDB (DESKTOP-5PCB31F(user (61))) - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

LSBUCruiseLinersDB

Object Explorer

Connect

DESKTOP-5PCB31F\SQLEXPRESS (SQL...

Databases

System Databases

Database Snapshots

LSBUCruiseLinersDB

Database Diagrams

Tables

System Tables

FileTables

External Tables

Graph Tables

dbo.Departure

dbo.Employee

dbo.OnBoardEntertainm

dbo.Passenger

dbo.Reservation

dbo.Seaport

dbo.Ship

dbo.Voyage

Views

External Resources

Synonyms

Programmability

Query Store

Service Broker

Storage

Security

Security

Server Objects

Replication

Management

XEvent Profiler

SQLQuery5.sql - DE...5PCB31F(user (61))*

SQLQuery4.sql - DE...5PCB31F(user (73))*

SQLQuery3.sql - DE...5PCB31F(user (56))*

SQLQuery2.sql - DE...5PCB31

SELECT * FROM Voyage;

200 %

Results Messages

	voyage_number	origin	destination	departure_time	arrival_time	meal_code	base_fare	mileage	time_zone_changes
1	101	London	New York	2023-05-01 10:00:00.000	2023-05-02 02:00:00.000	A	1200.00	3460	-5
2	102	New York	London	2023-05-03 15:00:00.000	2023-05-04 05:00:00.000	B	1100.00	3460	5
3	103	Tokyo	San Francisco	2023-06-01 11:00:00.000	2023-06-02 11:00:00.000	A	1500.00	5500	-8
4	104	San Francisco	Tokyo	2023-06-04 12:00:00.000	2023-06-05 16:00:00.000	B	1500.00	5500	9
5	105	Sydney	Los Angeles	2023-07-01 13:00:00.000	2023-07-02 10:00:00.000	A	1400.00	7500	-17
6	106	Los Angeles	Sydney	2023-07-03 14:00:00.000	2023-07-04 18:00:00.000	B	1400.00	7500	17
7	107	Mumbai	Dubai	2023-08-01 20:00:00.000	2023-08-02 00:00:00.000	A	700.00	1900	-1
8	108	Dubai	Mumbai	2023-08-03 21:00:00.000	2023-08-04 01:30:00.000	B	700.00	1900	1
9	109	Rio de Janeiro	Lisbon	2023-09-01 22:00:00.000	2023-09-02 10:00:00.000	A	900.00	4800	3
10	110	Lisbon	Rio de Janeiro	2023-09-03 23:00:00.000	2023-09-04 07:00:00.000	B	900.00	4800	-3
11	111	Cape Town	London	2023-10-01 16:00:00.000	2023-10-02 04:00:00.000	A	950.00	6000	2
12	112	London	Cape Town	2023-10-03 17:00:00.000	2023-10-04 05:00:00.000	B	950.00	6000	-2
13	113	Paris	Singapore	2023-11-01 18:00:00.000	2023-11-02 14:00:00.000	A	1300.00	6800	-7
14	114	Singapore	Paris	2023-11-03 19:00:00.000	2023-11-04 15:00:00.000	B	1300.00	6800	7
15	115	New York	Miami	2023-12-01 09:00:00.000	2023-12-01 12:00:00.000	A	300.00	1300	0
16	116	Miami	New York	2023-12-02 13:00:00.000	2023-12-02 16:00:00.000	B	300.00	1300	0
17	117	Beijing	Moscow	2024-01-01 08:00:00.000	2024-01-01 12:00:00.000	A	800.00	3600	-5
18	118	Moscow	Beijing	2024-01-02 14:00:00.000	2024-01-02 18:00:00.000	B	800.00	3600	5
19	119	Berlin	Rome	2024-02-01 07:00:00.000	2024-02-01 09:00:00.000	A	200.00	1184	1
20	120	Rome	Berlin	2024-02-02 10:00:00.000	2024-02-02 12:00:00.000	B	200.00	1184	-1

Figure 1.06. Verification of Voyage table data.

4.1.3. Ship Table.

The ship table is created to Contains information about the ships in the cruise liner's fleet, including ship names, types, cabin capacities, and technical specifications like displacement, fuel capacity, and maximum speed. It helps in managing the fleet and assigning ships to voyages.

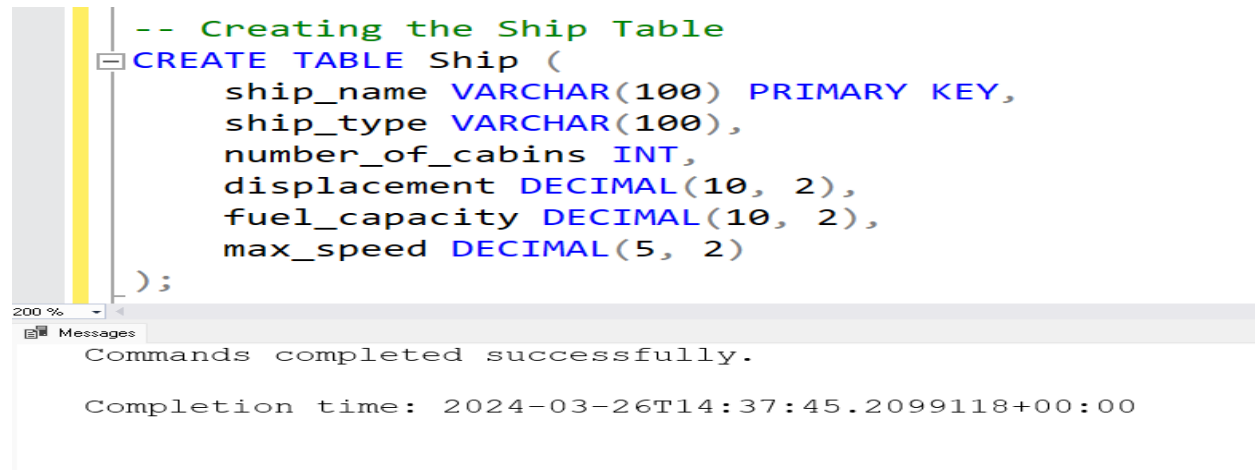


Figure 1.07. Create Ship Table.

- Verification.

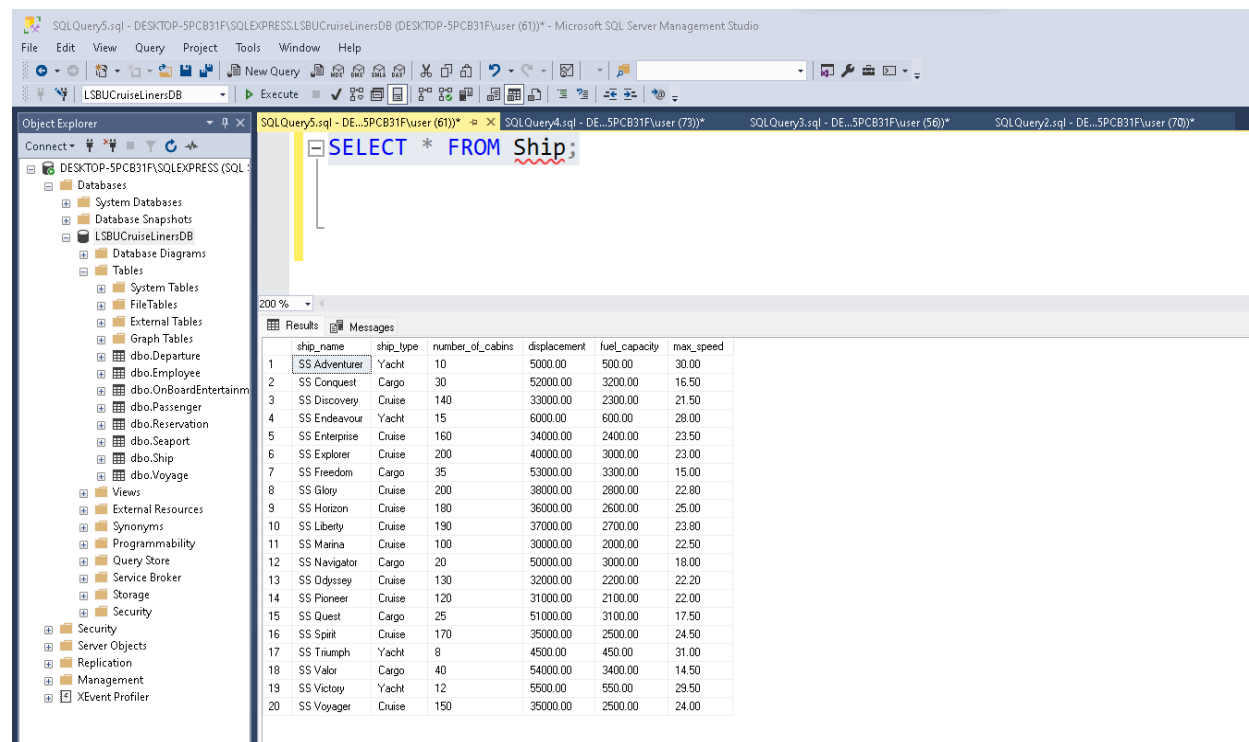


Figure 1.08. Verification of Ship Table data.

4.1.4. Departure Table.

The departure table is created to Records specific departure instances of voyages, linking voyages to dates, ships, and captains. This table facilitates the operational planning of each voyage, including crew assignments and ship scheduling.

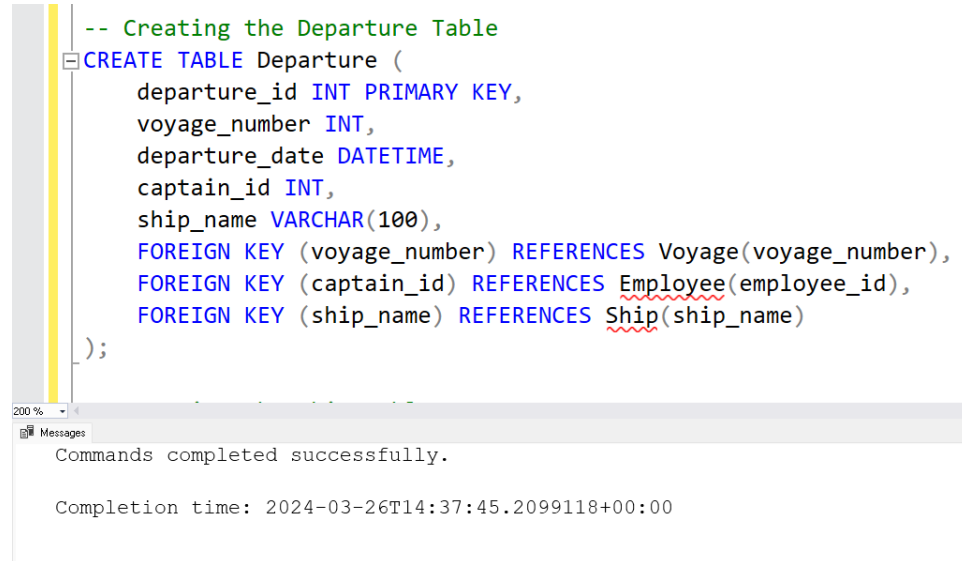


Figure 1.09. Create Departure Table.

- Verification.

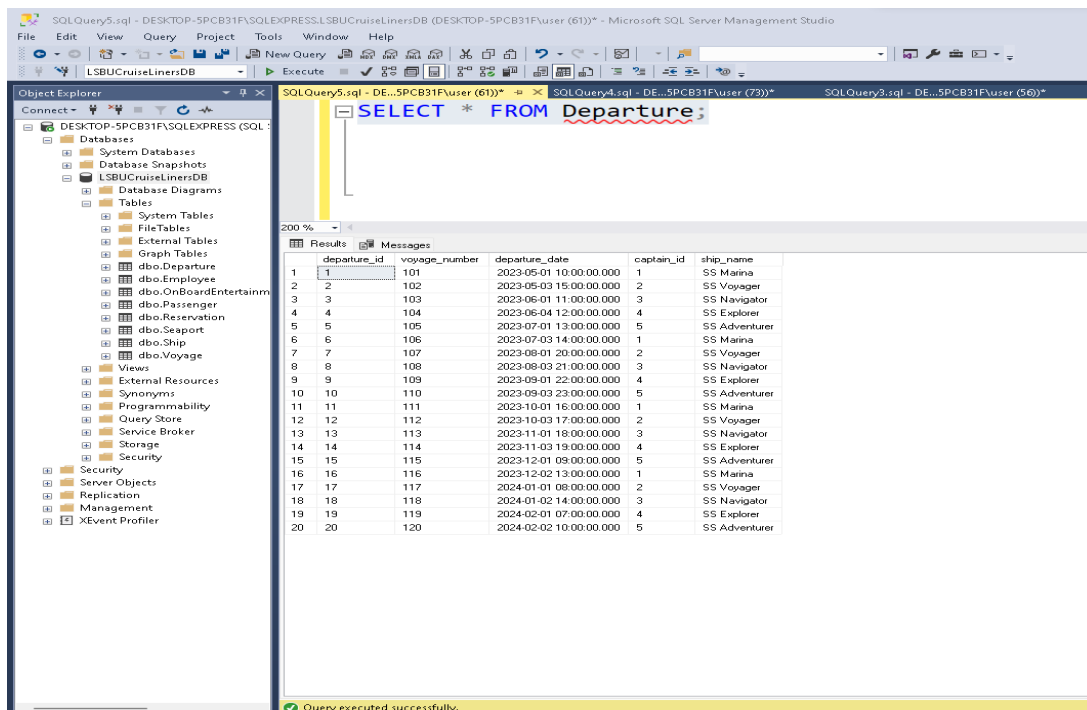
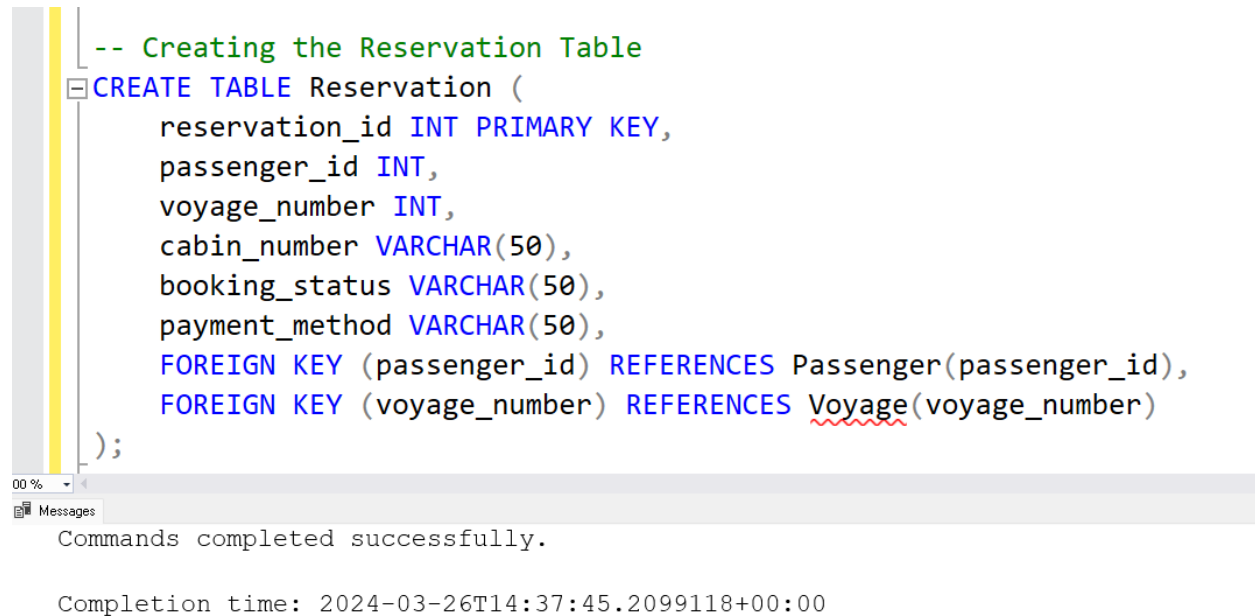


Figure 2.00. Verification of Departure table data.

4.1.5. Reservation Table.

This reservation table is created to Tracks bookings made by passengers, including reservation details, associated passenger and voyage information, cabin numbers, booking status, and payment methods. It is vital for managing passenger reservations and occupancy rates.



```
-- Creating the Reservation Table
CREATE TABLE Reservation (
    reservation_id INT PRIMARY KEY,
    passenger_id INT,
    voyage_number INT,
    cabin_number VARCHAR(50),
    booking_status VARCHAR(50),
    payment_method VARCHAR(50),
    FOREIGN KEY (passenger_id) REFERENCES Passenger(passenger_id),
    FOREIGN KEY (voyage_number) REFERENCES Voyage(voyage_number)
);
```

00 %

Messages

Commands completed successfully.

Completion time: 2024-03-26T14:37:45.2099118+00:00

Figure 2.01. Create Reservation Table.

- Verification.

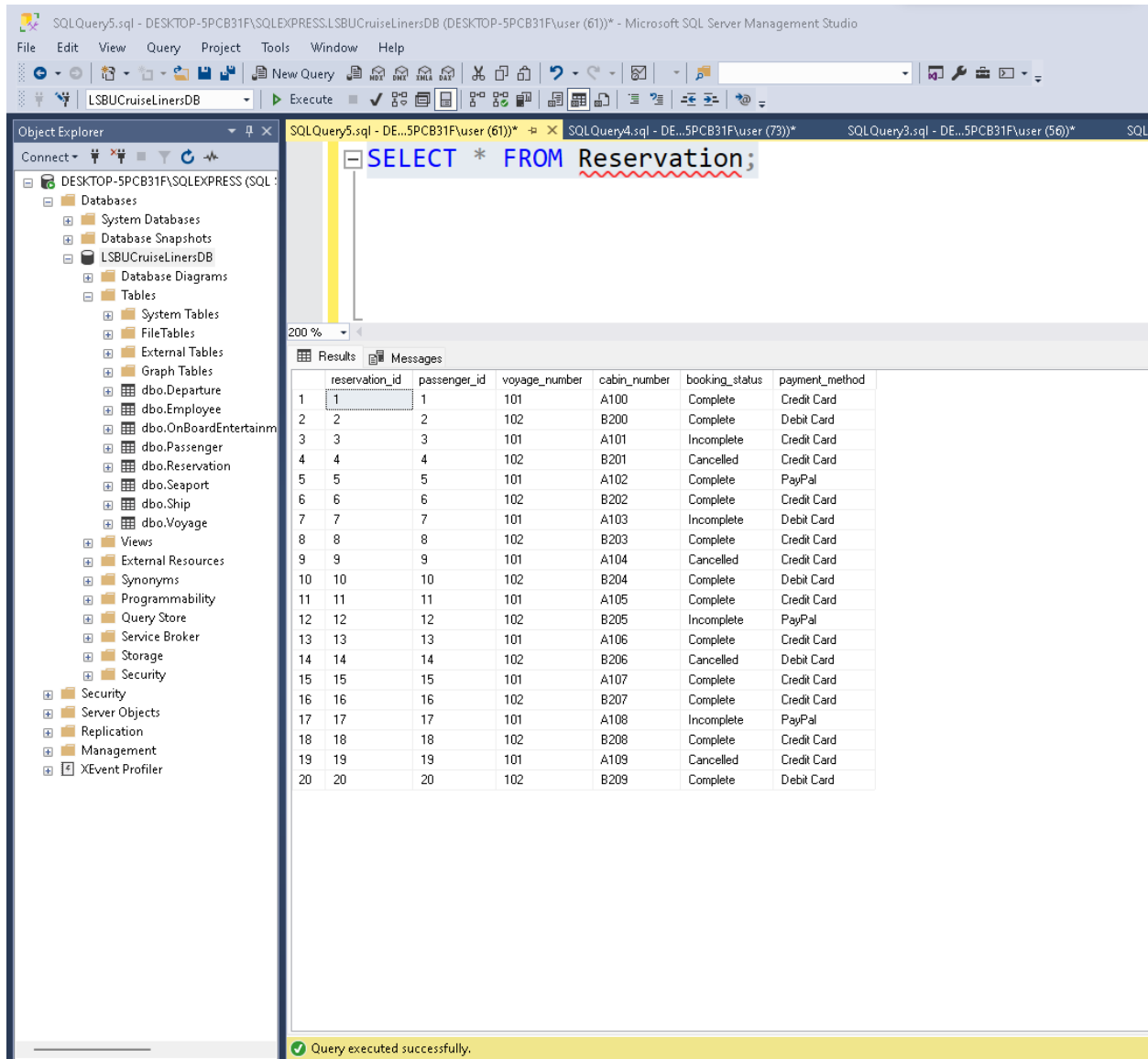


Figure 2.02. Verification of Reservation Table Data.

4.1.6. Employee Table.

The employee table is created to Details the cruise line's employees, covering roles aboard the ships and at ports. Attributes include employee identifiers, names, job titles, personal information, and employment details. This table supports human resources management and crew scheduling.

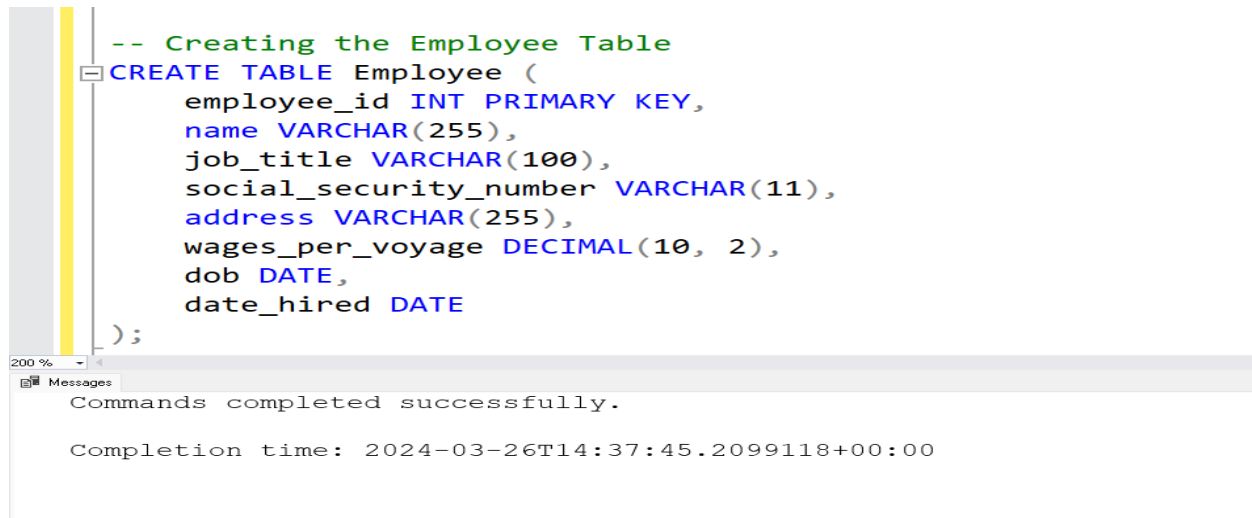


Figure 2.03. Create employee table.

- Verification.

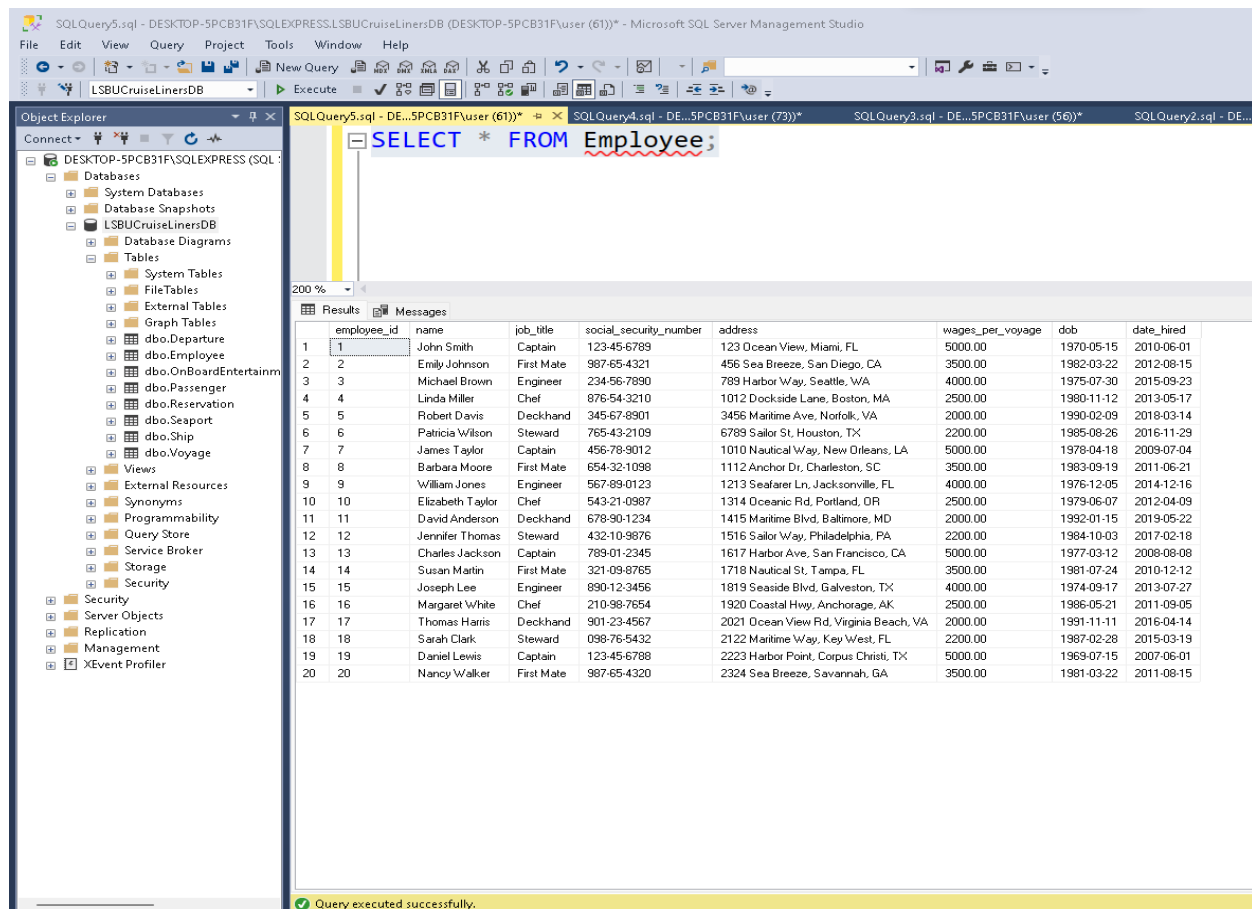


Figure 2.04. verification of employee table data.

4.1.7. Seaport Table.

The seaport table is created to Lists seaports where voyages originate, terminate, or make port calls, including seaport codes, city locations, countries, and contact information. It is essential for logistical planning and port management.

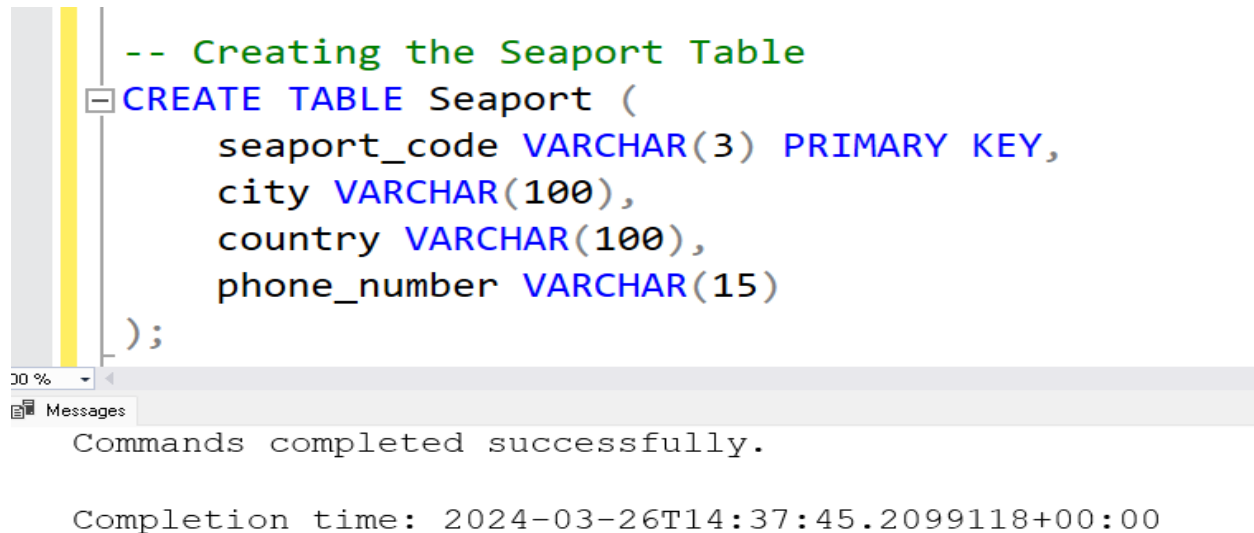


Figure 2.05. Seaport table creation.

- Verification.

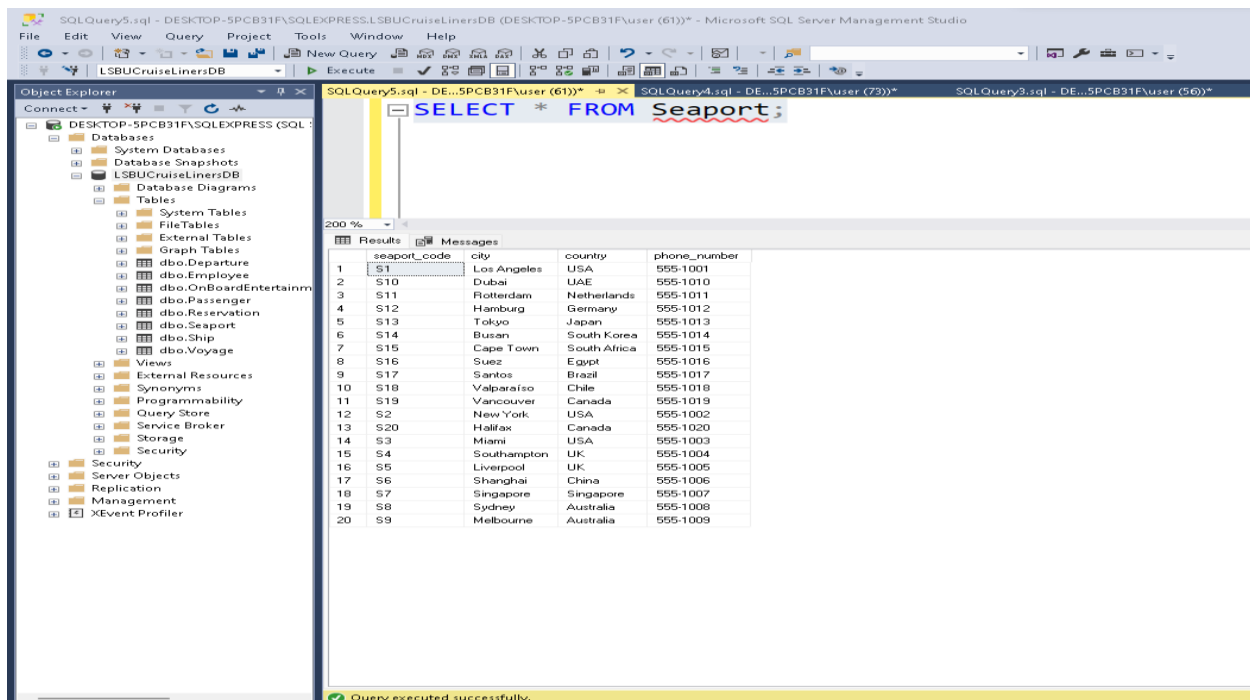


Figure 2.06. Verification of Seaport table data.

4.1.8. OnBoardEnterTainment.

The OnBoardEntertainment table is created to Catalogs the entertainment and activity options available onboard, with details on the type of entertainment, and where it takes place on the ship. This table enhances the passenger experience by facilitating entertainment scheduling and promotion.

```
-- Creating the OnBoardEntertainment Table
CREATE TABLE OnBoardEntertainment (
    entertainment_id INT PRIMARY KEY,
    type VARCHAR(100),
    location VARCHAR(100)
);
```

Commands completed successfully.

Completion time: 2024-03-26T14:37:45.2099118+00:00

Figure 2.07. Create OnBoardEnterTainment Table.

- Verification.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left displays the database structure for 'LSBUCruiseLinersDB', including tables like 'dbo.OnBoardEntertainment'. The SQL Query window shows the query: `SELECT * FROM OnBoardEntertainment;`. The Results pane displays the following data:

entertainment_id	type	location
1	Live Music	Main Deck
2	Comedy Show	Auditorium
3	Magic Show	Auditorium
4	Movie Night	Cinema
5	Dance Class	Dance Hall
6	Cooking Workshop	Kitchen Studio
7	Wine Tasting	Lounge
8	Art Auction	Gallery
9	Karaoke Night	Karaoke Bar
10	Casino Night	Casino
11	Theater Play	Theater
12	Jazz Night	Jazz Club
13	Pool Party	Pool Deck
14	Lecture Series	Lecture Hall
15	Book Club	Library
16	Craft Workshop	Workshop Room
17	Fitness Class	Gym
18	Yoga Session	Fitness Studio
19	DJ Session	Nightclub
20	Trivia Night	Lounge

Query executed successfully.

Figure 2.08. Verification of OnBoardEnterTainment Table data.

5. SQL Queries and Procedures.

5.1. Query for Reservation Totals.

- Query Explanation.

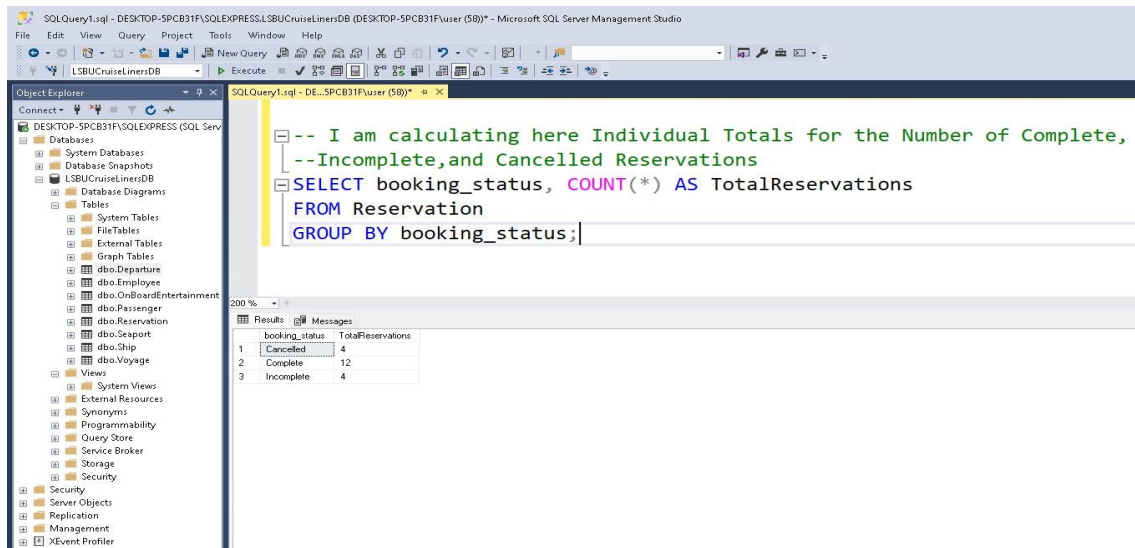


Figure 2.09. Reservation Total queries.

This SQL query is constructed to fulfill the requirement of determining the individual totals for the number of reservations that are marked as 'Complete', 'Incomplete', and 'Cancelled'. The objective is to provide a clear count of each status category, which is essential for understanding the reservation dynamics and for further managerial decisions.

- Output.

The output of the query shows a list with those columns – 'booking_status' and 'TotalReservations'. Each row represents a unique booking status and the total number of reservations for that status.

5.2. Query for Available Cabins.

- Objective of the Query.

The query is designed to provide a list of voyages, along with the details of passenger cabins on each voyage. Specifically, it aims to find out how many cabins have been booked and how many remain available for each ship on its respective voyage. This information is crucial for the cruise line's booking department to manage reservations effectively.

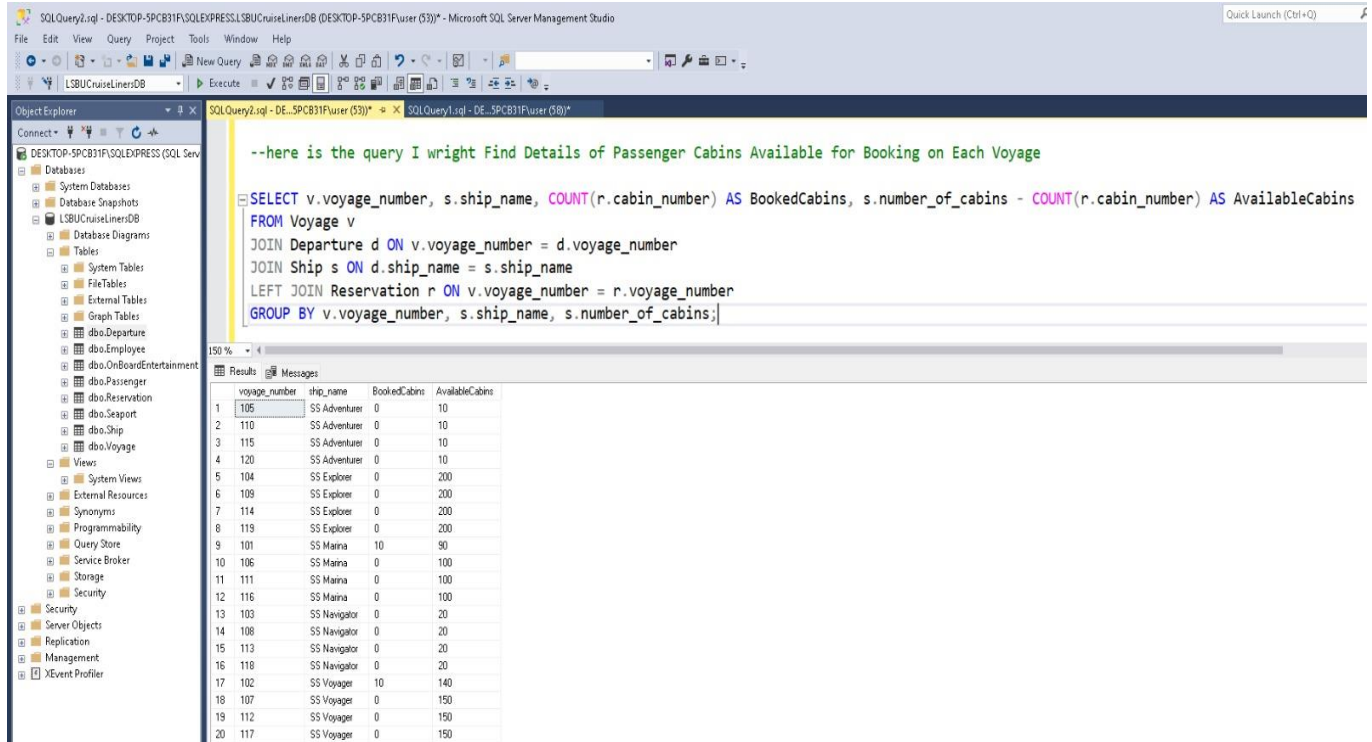


Figure 3.00. Query about available cabins.

- Results.

The result set provides a clear count of booked and available cabins for each ship on every voyage. For example, if a ship named "SS Adventurer" on voyage 105 has 200 cabins in total and none of them are booked, it means that all 200 cabins are available for booking.

5.3. Stored procedure for total revenue generated by On-Board Entertainment and Activity sales.

- Objective of the Query.

The stored procedure is aimed at calculating the total revenue generated by sales of each type of onboard entertainment and activity for each voyage. It provides a financial insight into the performance of entertainment offerings aboard the cruise liner.

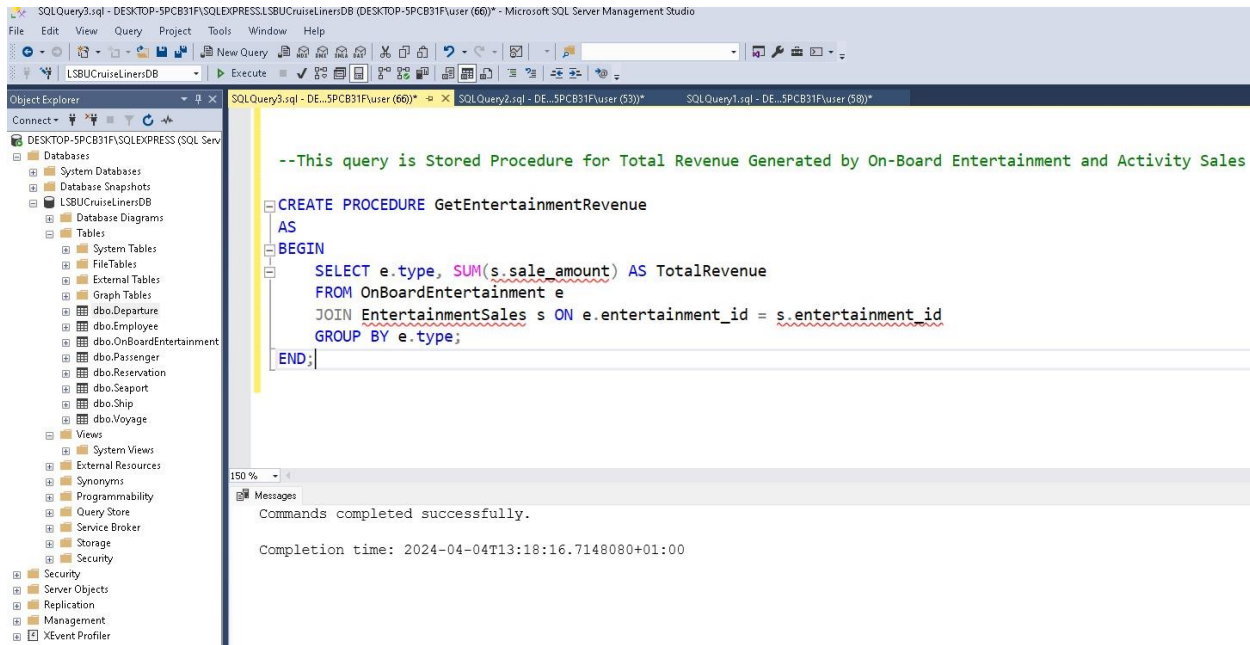


Figure 3.01. Stored procedure for total revenue.

- Results.

5.4. Trigger for Preventing double bookings.

- Objective.

To ensure the integrity of the booking system by preventing double-booking of cabins on the same voyage within the LSBUCruiseLinersDB database.

- Implementation.

A trigger named 'PreventDoubleBooking' was created with the specific task of verifying each booking attempt against existing reservations.

This trigger intercepts INSERT operations on the Reservation table and performs a check to confirm if the cabin number on a particular voyage already has a 'Complete' booking status.

- Validation and Results.

The Following steps were taken to validate the triggers functionality:

- Trigger Creation.

The following image demonstrates the SQL script used to create the PreventDoubleBooking trigger. The trigger's logic was designed to query existing reservations and block the insertion of duplicates.

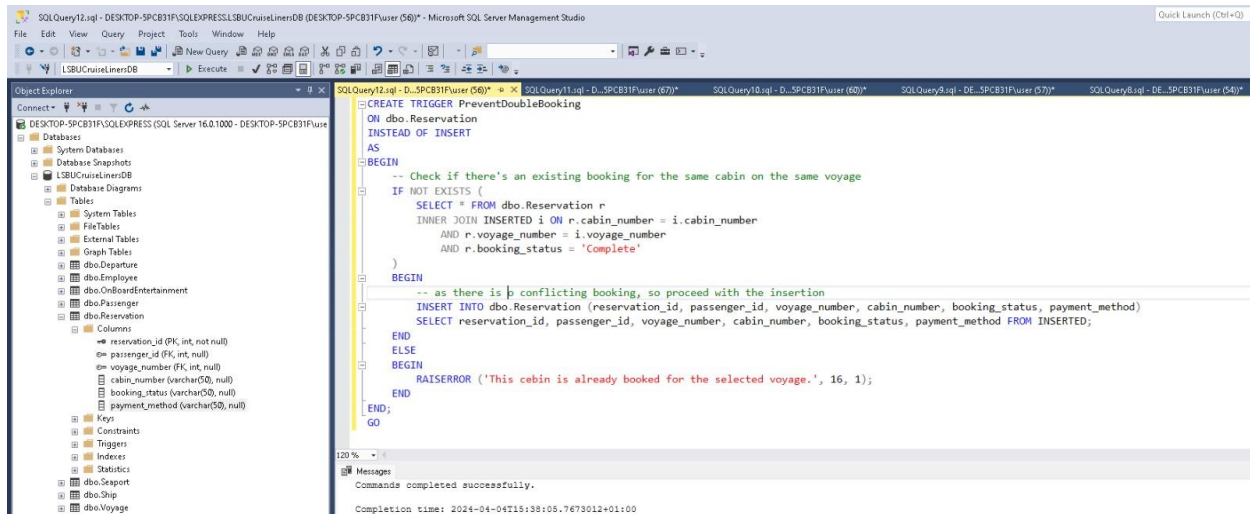


Figure 3.02. Trigger Creation.

- Initial Insertion.

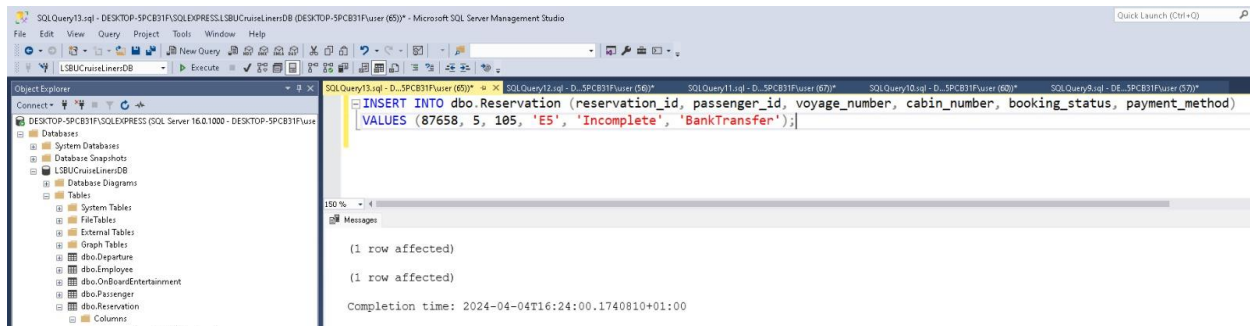


Figure 3.02. Initial insertion values.

This screenshot shows a successful insertion of a booking record, establishing a baseline for the subsequent test of the trigger.

- Double Booking Prevention.

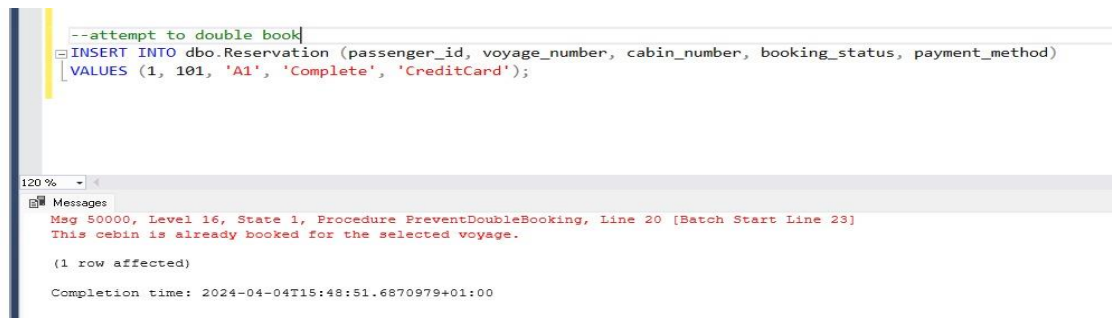
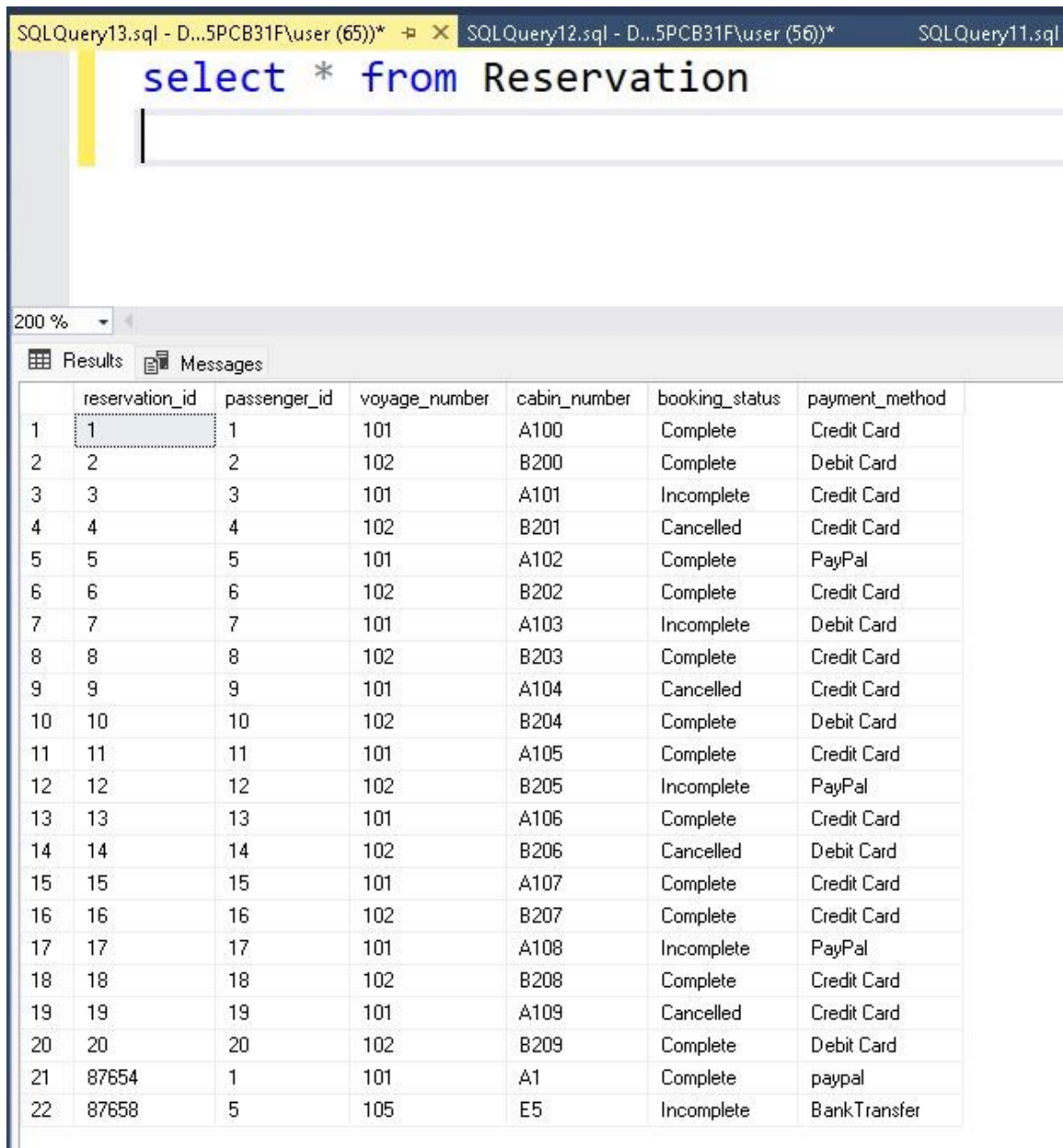


Figure 3.03. Preventing Double Booking.

- Post Trigger Validation.

Finally, this screenshot shows the result of a query on the 'Reservation' table after the trigger's execution. It confirms that no duplicate booking was entered into the database, verifying that the trigger successfully prevented the double-booking.



	reservation_id	passenger_id	voyage_number	cabin_number	booking_status	payment_method
1	1	1	101	A100	Complete	Credit Card
2	2	2	102	B200	Complete	Debit Card
3	3	3	101	A101	Incomplete	Credit Card
4	4	4	102	B201	Cancelled	Credit Card
5	5	5	101	A102	Complete	PayPal
6	6	6	102	B202	Complete	Credit Card
7	7	7	101	A103	Incomplete	Debit Card
8	8	8	102	B203	Complete	Credit Card
9	9	9	101	A104	Cancelled	Credit Card
10	10	10	102	B204	Complete	Debit Card
11	11	11	101	A105	Complete	Credit Card
12	12	12	102	B205	Incomplete	PayPal
13	13	13	101	A106	Complete	Credit Card
14	14	14	102	B206	Cancelled	Debit Card
15	15	15	101	A107	Complete	Credit Card
16	16	16	102	B207	Complete	Credit Card
17	17	17	101	A108	Incomplete	PayPal
18	18	18	102	B208	Complete	Credit Card
19	19	19	101	A109	Cancelled	Credit Card
20	20	20	102	B209	Complete	Debit Card
21	87654	1	101	A1	Complete	paypal
22	87658	5	105	E5	Incomplete	BankTransfer

Figure 3.04. Post Trigger Validation.

5.5. Stored procedure for reservation invoice.

- Objectives.

The aim of this stored procedure, 'GenerateInvoice,' is to dynamically produce a detailed reservation invoice for a given customer. The invoice includes comprehensive customer information, the reservation details, and the cost calculations before and after the application of a Value Added Tax (VAT) of 20%.

- Implementation.

The procedure is created with a parameter '@ReservationID' to specify which reservation needs an invoice. It retrieves relevant data by joining the 'Reservation,' 'Passenger,' and 'Voyage' tables. The essential details, such as the passenger's name, reservation ID, voyage number, and cabin number, are selected alongside the 'base_fare' from the Voyage table, which acts as the subtotal for the invoice. The VAT is calculated by multiplying the 'base_fare' by 1.20 (representing the 20% VAT rate), yielding the total invoice amount including VAT.

```

CREATE PROCEDURE GenerateInvoice @ReservationID INT
AS
BEGIN
    SELECT
        p.name AS PassengerName,
        r.reservation_id,
        r.voyage_number,
        r.cabin_number,
        r.booking_status,
        r.payment_method,
        v.base_fare AS SubTotal,
        v.base_fare * 1.20 AS TotalWithVAT -- VAT is 20% based on my coursework specification
    FROM
        dbo.Reservation r
        JOIN dbo.Passenger p ON r.passenger_id = p.passenger_id
        JOIN dbo.Voyage v ON r.voyage_number = v.voyage_number
    WHERE
        r.reservation_id = @ReservationID;
END;

```

150 %

Messages

Commands completed successfully.

Completion time: 2024-04-08T09:45:35.4840707+01:00

Figure 3.05. Invoice Generation via stored procedure.

- Execution and Result.

To validate the procedure, it is executed with a sample '@ReservationID'. The following command is used to generate the invoice.

Upon execution, the stored procedure successfully returned a formatted output displaying the invoice details for the specified reservation. The result included the customer's name, reservation details, and the financial breakdown with the VAT applied.

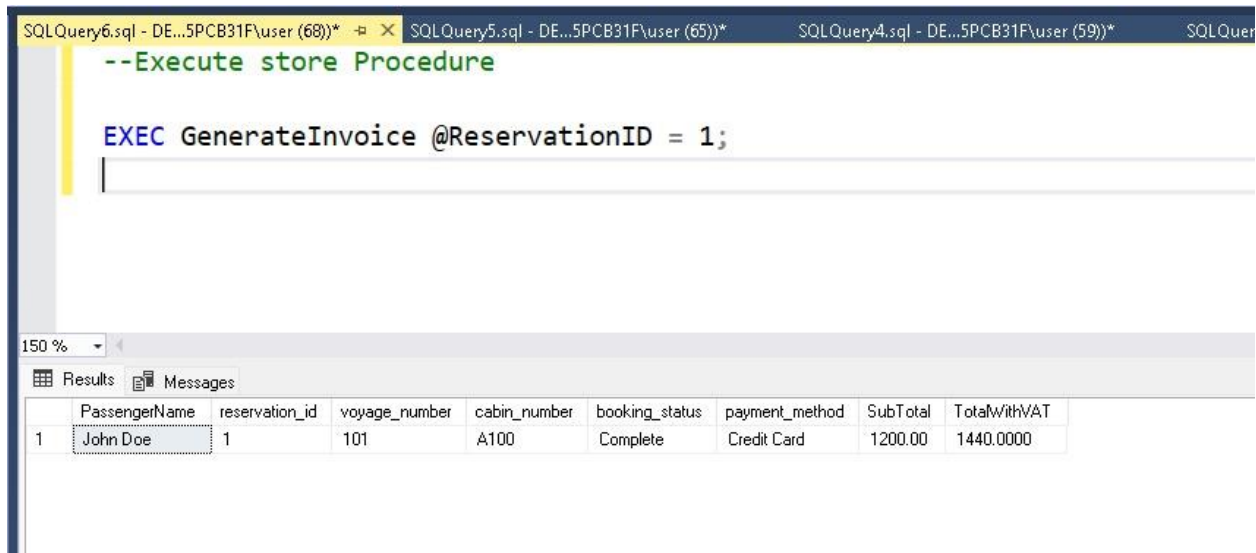


Figure 3.06. Execute store procedure.

6. Dashboard Development.

6.1. Design Consideration.

The LSBU Cruise Liners Operations Overview dashboard was conceptualized with the primary goal of offering stakeholders a quick, intuitive understanding of various operational aspects through visualized data. The dashboard's design centers around four key elements: booking trends over time, voyage popularity, cabin availability by voyage, and top destinations. These elements were chosen for their direct impact on business decisions and strategy formulation.

6.2. Implementation in Tableau.

The initial step in developing the LSBU Cruise Liners Operations Overview dashboard involved establishing a robust data connection. This was achieved by integrating Microsoft SQL Server with Tableau Desktop, providing direct access to the operational databases.

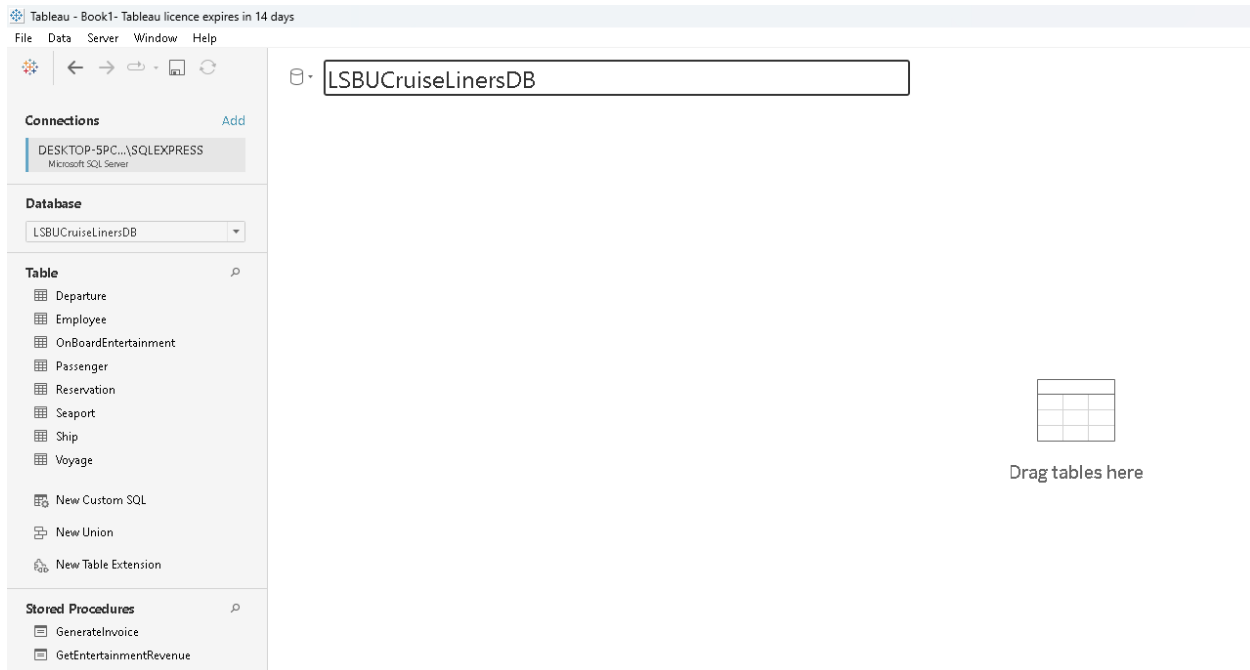


Figure 3.07. Connecting SQL server to Tableau.

The dashboard implementation in Tableau involved the creation of four distinct visualizations:

- **Booking trends Over Time:** A line chart visualization was created to depict the fluctuations in booking volumes over time, segmented by booking status for granular insights.

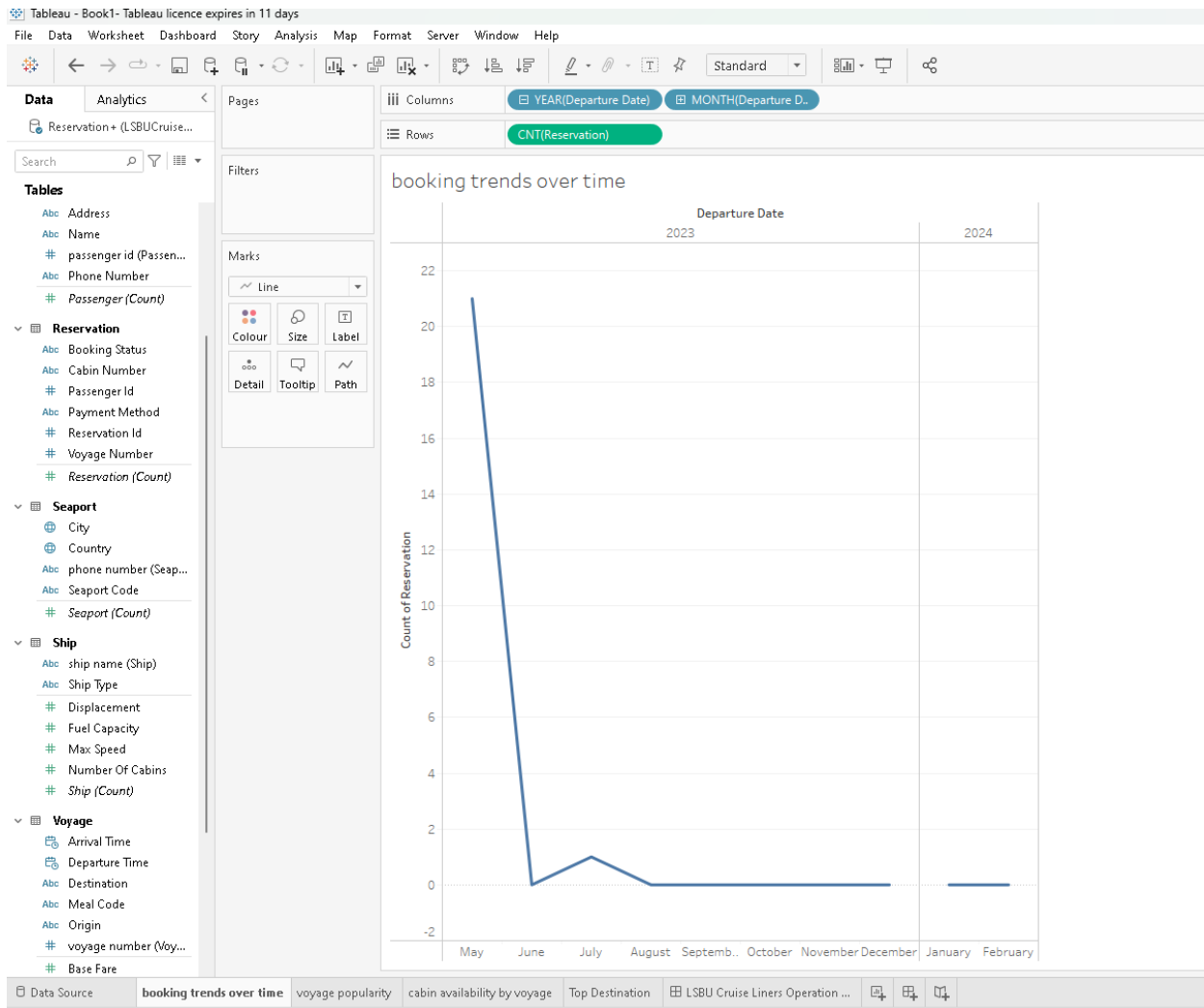


Figure 3.08. Booking trends over time.

- **Voyage Popularity:** A bar chart was utilized to rank voyages by booking count, offering a clear depiction of the most sought-after routes.

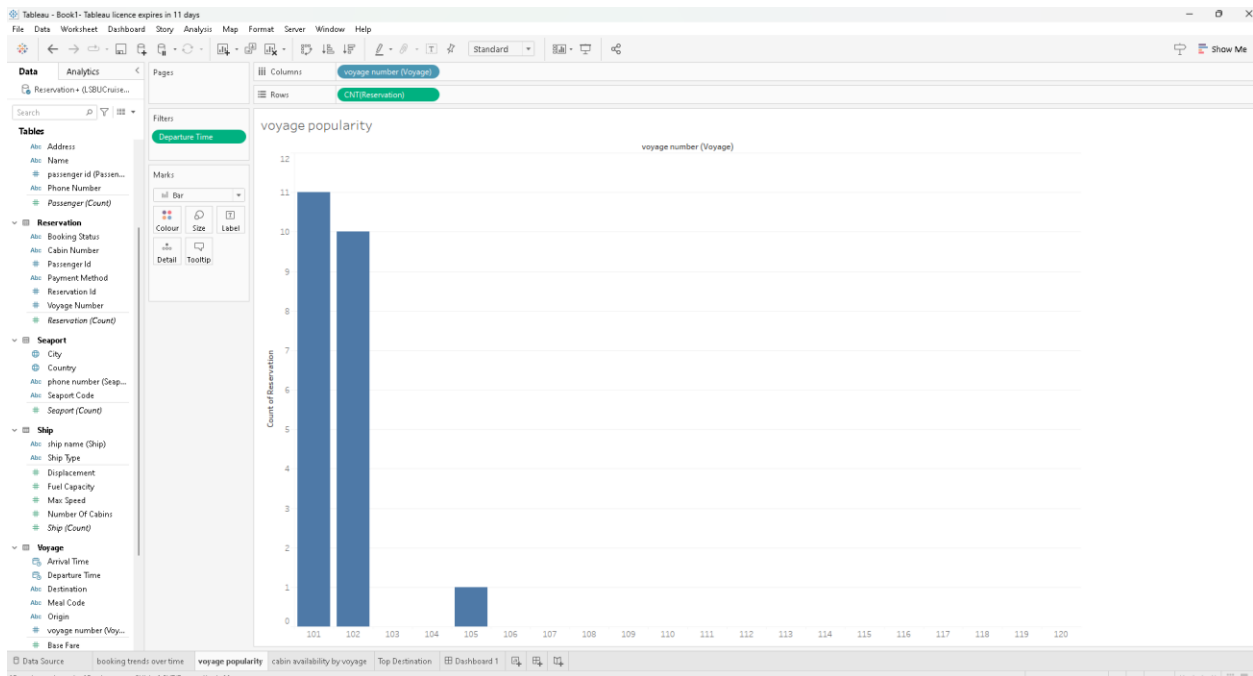


Figure 3.09. Voyage Popularity.

- Cabin Availability by Voyage: A heat map was generated to provide a visual on cabin occupancy rates, enabling quick decisions on pricing strategies and promotional offerings.

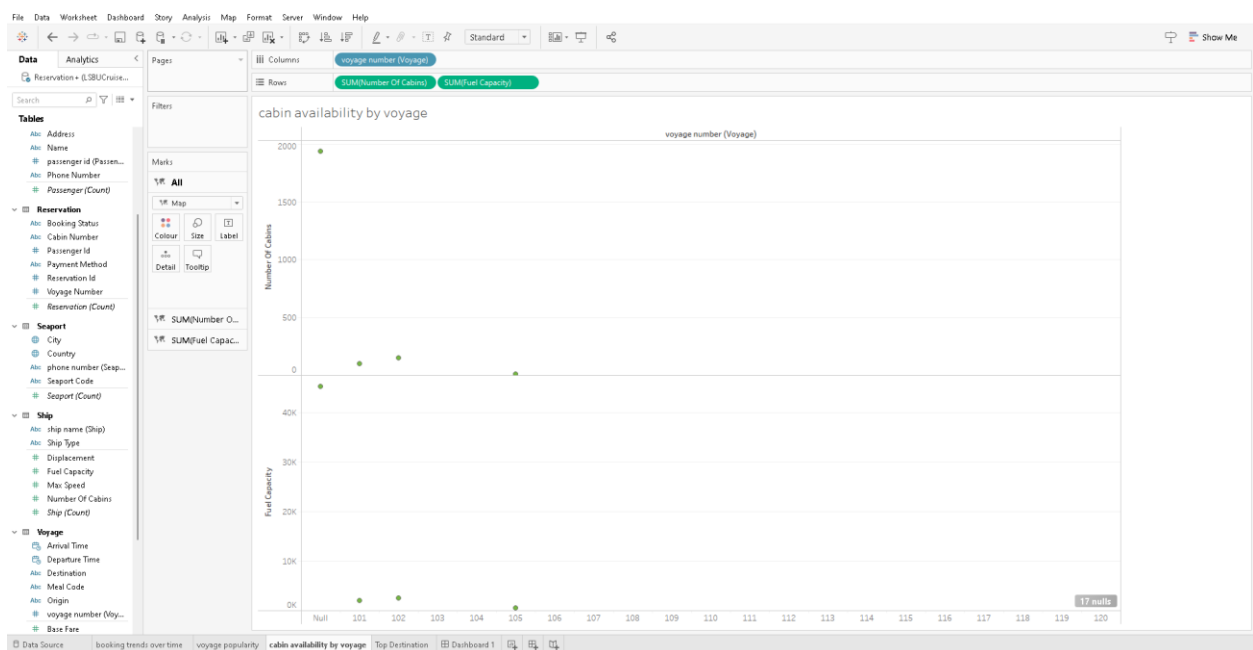


Figure 4.00. Cabin Availability by Voyage.

- Top Destination: A tree map was crafted to highlight the most popular destinations, based on booking volumes, serving as a guide for marketing focus.

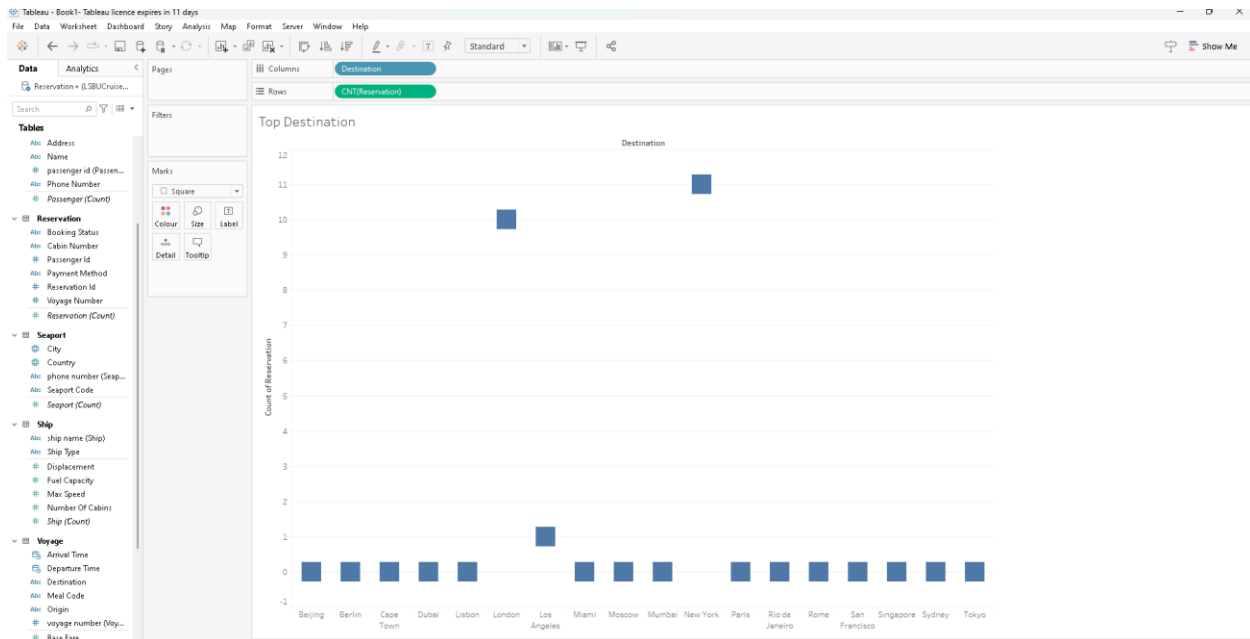


Figure 4.01. Top Destination.

- Tableau Dashboard.

In this following part, The LSBU Cruise Liners Operations dashboard highlighting key business insights, displaying booking trends, voyage popularity, cabin availability, and top destinations through a cohesive visual narrative to guide strategic decisions.

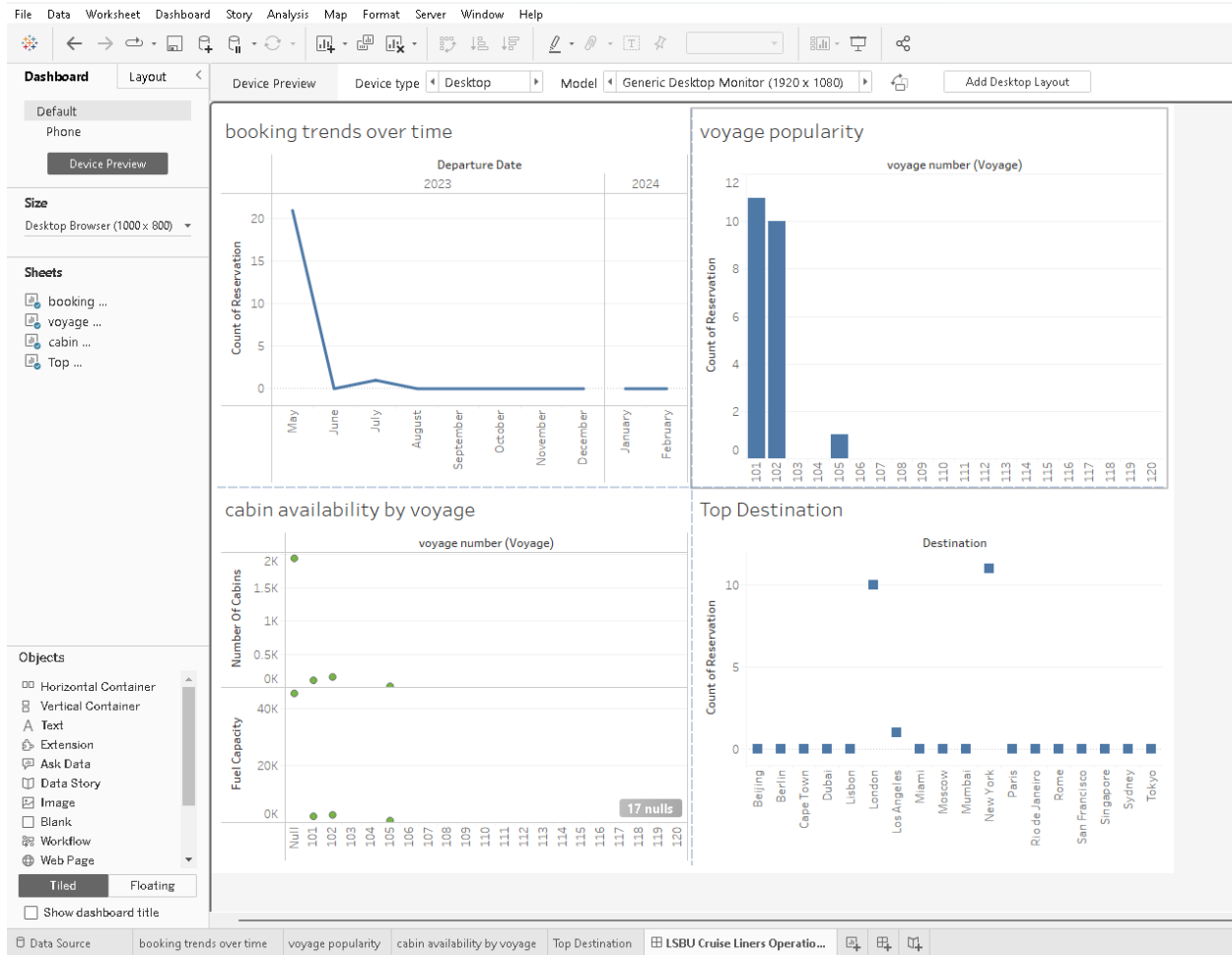


Figure 4.02. LSBU Cruise Liners Operations Overview.

7. Video Demonstration.

7.1. Overview of Video Content.

The video provides a walkthrough of key SQL operations and a showcase of the interactive Tableau dashboard, demonstrating the practical application and analysis of LSBU Cruise Liners' database.

7.2. Execution of SQL task.

The execution phase in SQL Server delves into the querying process that supports data retrieval for reservations, revealing the underlying patterns and efficiencies within the cruise liner's operations.

7.3. Dashboard Showcase.

In Tableau, I explore the constructed dashboard, illustrating the seamless integration of data visualizations that bring forth insights on booking trends, voyage popularity, cabin

availability, and top destinations. This visual representation facilitates strategic decision-making by highlighting critical business metrics.

Here is the video demonstration link:

https://stulsbuac-my.sharepoint.com/:v/g/personal/sufiunm_lsbu_ac_uk/EWWlhCGyWptMtGQczmhUopgB-DZa9RIFj-MvwrhQOdq1iQ

8. Conclusion.

8.1. Project Summary.

This report encapsulated the data-driven expedition of LSBU Cruise Liners' operational metrics. Task 4's SQL executions shed light on reservation dynamics, while Task 5's Tableau visualizations narrated the story behind the voyages, cabin logistics, and customer engagements.

8.2. Key Learnings and Outcomes.

The project highlighted the vitality of correlating data to extract meaningful insights. The visualization process not only simplified complex data interpretation but also provided actionable intelligence for enhancing customer experiences and operational workflows. Consequently, this has paved the way for an insightful, data-informed approach to business strategy for LSBU Cruise Liners.

9. References

- Data Modeling and the ER model (URL <https://www.youtube.com/watch?v=lfagkiHpljo> Last access on 1st of April 2024).
- Normalization in DBMS (URL <https://www.studytonight.com/dbms/database-normalization.php> Last access 2nd of April 2024).
- Install SQL server (URL <https://www.microsoft.com/en-gb/sql-server/sql-server-downloads> Last access 1st of April).
- Create table (URL <https://learn.microsoft.com/en-us/sql/t-sql/statements/create-table-transact-sql?view=sql-server-ver15> last access 3rd of April 2024).
- SQL tutorial (URL <https://www.w3schools.com/sql/> last access 5th April 2024).
- Update Transact-SQL (URL <https://learn.microsoft.com/en-us/sql/t-sql/queries/update-transact-sql?view=sql-server-ver15#UpdateExamples> last access on 5th of April 2024).

- Coalesce Transact-SQL (URL <https://learn.microsoft.com/en-us/sql/t-sql/language-elements/coalesce-transact-sql?view=sql-server-ver15> last access on 5th of April).
- MS SQL Server tutorials point (URL PDF file:///C:/Users/Dell%20Desktop/Downloads/ms_sql_server_tutorial.pdf last access 10th of April 2024).
- SQL Contains String (URL <https://www.freecodecamp.org/news/sql-contains-string-sql-regex-example-query/> Last access on 10th of April)
- Data Management Reporting Tools (URL <https://www.tableau.com/en-gb/academic/students> last access 16th of April 2024).
- Tableau Basics for Beginners (URL <https://www.youtube.com/watch?v=jEgVto5QME8> last access 17th of April 2024).
- Demo Calculation (URL <https://www.youtube.com/watch?v=mInT38ZzIIQ> last access 16th of April 2024).