Project Report

DAMG 6210 Data Management and Database Design

Team Name: POLARIS

Project Name: Baggage and Catering Management System (Airport)

Team Members	NEU ID
Robins Ranjan	002401386
Mudra Pandya	002314500
Adithya Anand	002851036
Durga Dhana Sree Chilukuri	002481050

TABLE OF CONTENTS

PROBLEM STATEMENT

Efficient management of airport operations, including passenger information, baggage tracking, and catering services, remains a challenge. Manual processes or fragmented systems often lead to inefficiencies, errors, and delays, affecting both customer satisfaction and operational transparency. The lack of an integrated system results in data mismatch, making real-time updates and accurate tracking of services difficult.

This project seeks to address these issues by creating a robust, scalable, and secure relational database system. It will streamline operations, ensure real-time updates, and provide comprehensive reporting and analytics for effective decision-making. Additionally, the system will eliminate redundancies and reduce the risk of human errors by automating data management tasks.

With an emphasis on real-time tracking, passengers can have greater visibility into their baggage status and meal preferences, improving their travel experience. Catering services will benefit from an organized inventory system, ensuring timely replenishment and efficient utilization of resources.

The integration of all these functionalities into one unified database will enhance coordination among various airport departments. Furthermore, advanced analytics and reporting capabilities will support data-driven decision-making and enable predictive insights for better operational planning. Ultimately, this solution aims to set a new standard in airport management systems.

DESIGNING PROCESS

The designing process begins with conceptualizing the database structure by identifying core entities like Passengers, Baggage, Flights, Meals, and Catering Inventory. These entities are represented as tables with carefully chosen attributes to accurately capture the real-world data they represent.

Relationships between entities are mapped using primary and foreign keys to enforce referential integrity and maintain logical connections between the data. The process incorporates normalization to eliminate redundancy, enhance storage efficiency, and ensure consistent and error-free data. Real-world scenarios are modeled into queries, constraints, and triggers to automate data validations, such as ensuring accurate baggage assignments and catering inventory updates.

Complex scenarios like meal preferences linked to specific flight routes and baggage transfers between connecting flights are considered during the design phase. We also tried to optimize query performance for large datasets, ensuring the system can handle high traffic volumes during peak operations. The design also includes provisions for scalability, allowing the database to accommodate growing passenger volumes and operational complexities. Finally, iterative testing, refinement, and feedback loops with stakeholders ensure that the database design aligns with the system's functional, security, and performance requirements, delivering a robust and reliable solution for airport management.

REQUIREMENT GATHERING

To begin the development of our Baggage and Catering Management System (BCMS), we undertook a thorough investigation of airport operations. This process involved reviewing existing systems, studying current practices, and gathering valuable insights from airline staff, airport personnel, and catering vendors. We examined how baggage handling and catering services are managed, identifying potential gaps and challenges in current operations.

To ensure that our system would address real-world needs, we also monitored recent industry trends and challenges. This approach allowed us to understand both

the theoretical aspects and the practical issues faced by staff during day-to-day operations, such as real-time baggage tracking and dynamic meal inventory management. By combining knowledge from online research, stakeholder input, and industry updates, we developed a comprehensive set of requirements that would guide the design of a system optimized for efficiency, accuracy, and operational effectiveness.

VISION

The objective of the project is to transform airport operations by integrating a database system that manages passengers, baggage, and catering. Using relational database technology, the system intends to enable seamless data access, real-time updates, and robust analytics. This innovation will increase operational transparency, improve the customer experience, and optimize resource allocation. Finally, the system aims to establish a standard for efficiency and dependability in airport administration.

BUSINESS PROBLEMS ADDRESSED

Our Baggage and Catering Management System (BCMS) directly tackles several significant business challenges:

Real-Time Baggage Tracking: A common issue in current airport systems is the lack of real-time updates on baggage status, leading to mismanagement and delays. BCMS introduces a real-time baggage tracking feature that provides up-to-date information on the status of passengers' baggage from check-in through to final delivery. This reduces manual tracking errors and enhances operational efficiency, ensuring a smoother process for both staff and passengers.

Dynamic Meal Inventory Management: Catering services often face difficulties due to outdated inventory tracking methods, which can lead to meal shortages or overstocking. BCMS addresses this by implementing a dynamic meal inventory system. This feature provides real-time updates on meal stock, ensuring accurate

monitoring and proper replenishment of meals. It helps reduce food wastage and ensures that passengers' meal preferences are met, particularly during busy times.

Integrated Operations Across Services: Airport operations require seamless coordination between various services such as baggage handling and in-flight catering. Existing systems often operate in silos, leading to inefficiencies and communication delays. BCMS integrates these critical services into one cohesive system, optimizing resource allocation and reducing delays. This results in enhanced operational efficiency and a better overall experience for passengers, from baggage check-in to meal service on flights.

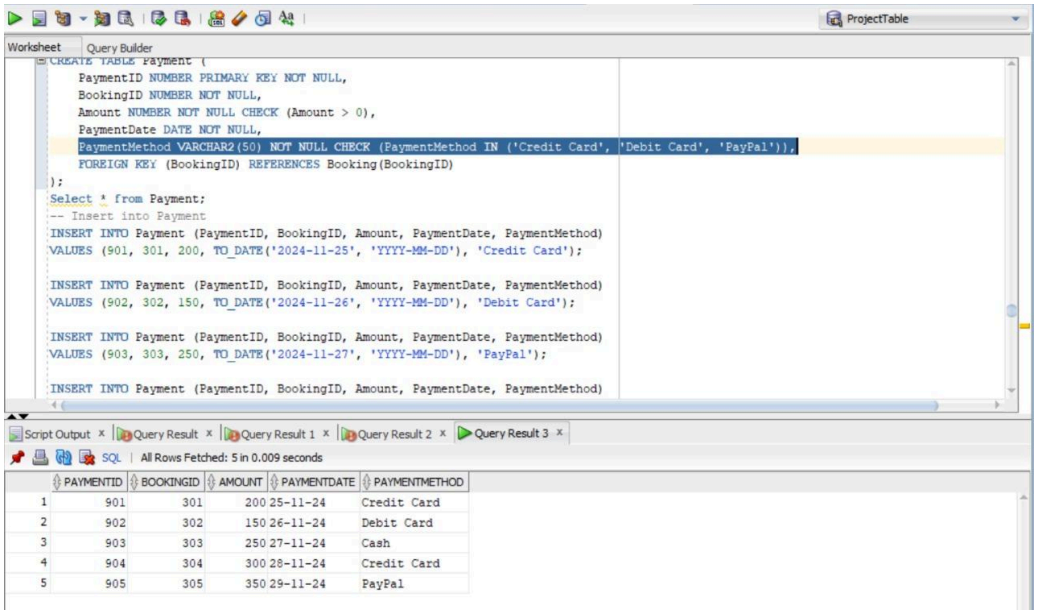
Improved Operational Planning via Passenger Data: Accurate passenger counts are essential for effective operational planning, especially in areas like catering and baggage handling. BCMS monitors passenger counts per flight in real time, enabling airport staff to better plan and allocate resources. This ensures that flights are properly catered, and sufficient baggage handling capacity is in place, avoiding overloading and ensuring smooth operations.

Vendor Performance and Procurement Optimization: Managing relationships with catering vendors and optimizing meal costs are crucial for airport operations. BCMS provides insights into vendor performance by tracking key metrics such as meal profitability and the utilization of different meal types. It helps airport management make informed decisions about vendor contracts and meal procurement, ensuring that meals are both cost-effective and meet passenger demand.

BUSINESS RULES

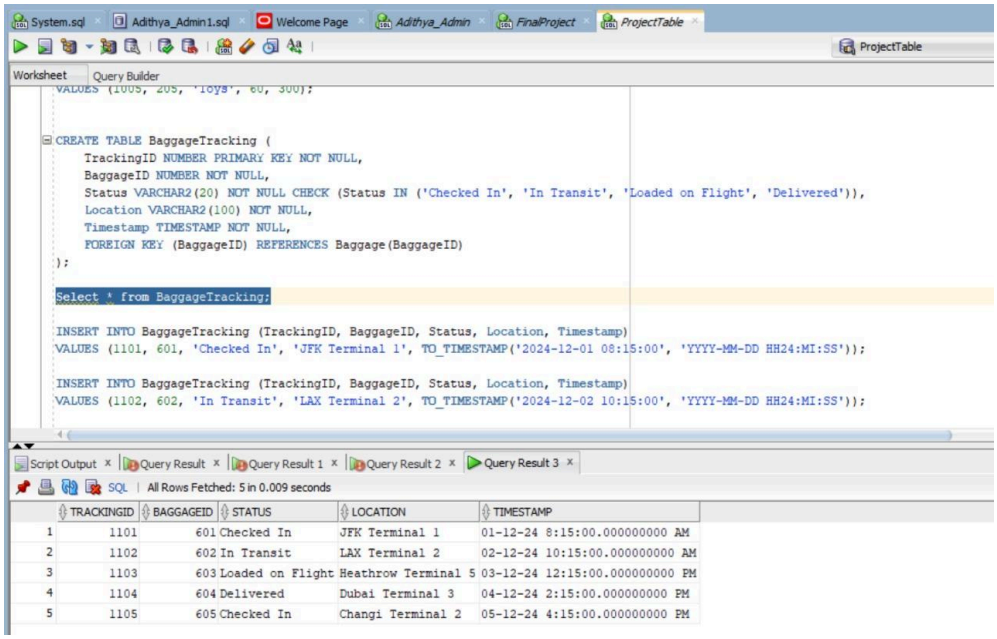
Business Rule Name: Valid Payment Methods

Description: The payment method used by passengers for transactions must be one of the predefined acceptable methods: Credit Card, Debit Card, or PayPal.



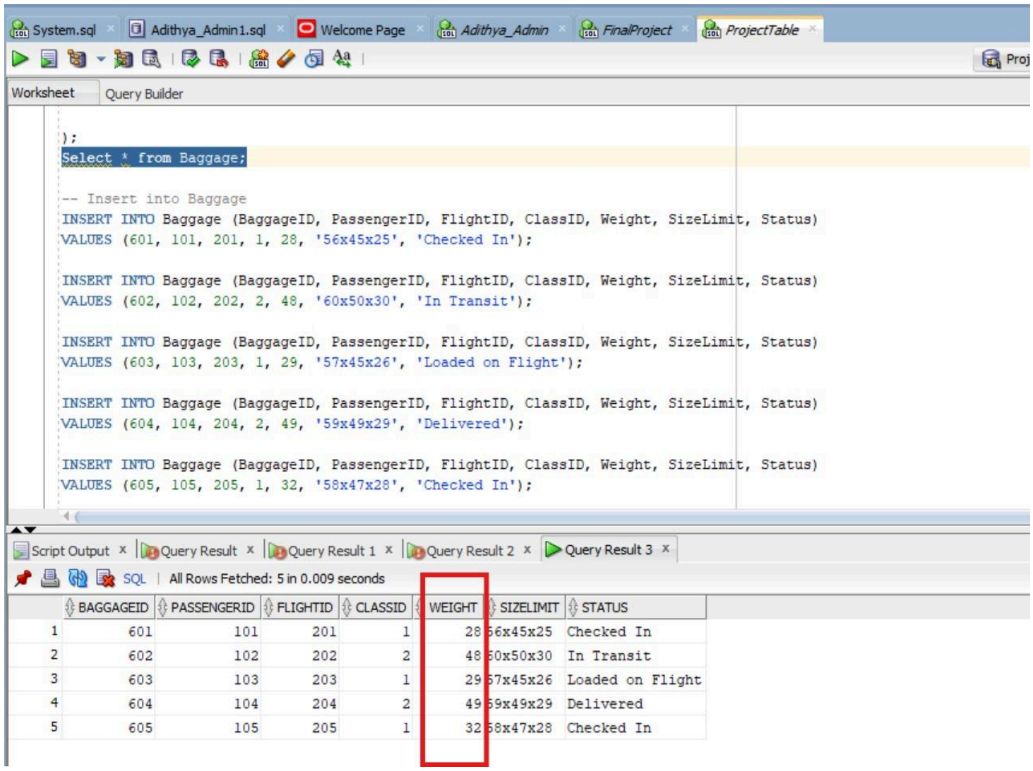
Business Rule Name: Valid Baggage Status

Description: The status of baggage must always be one of the predefined states: Checked In, In Transit, Loaded on Flight, or Delivered.



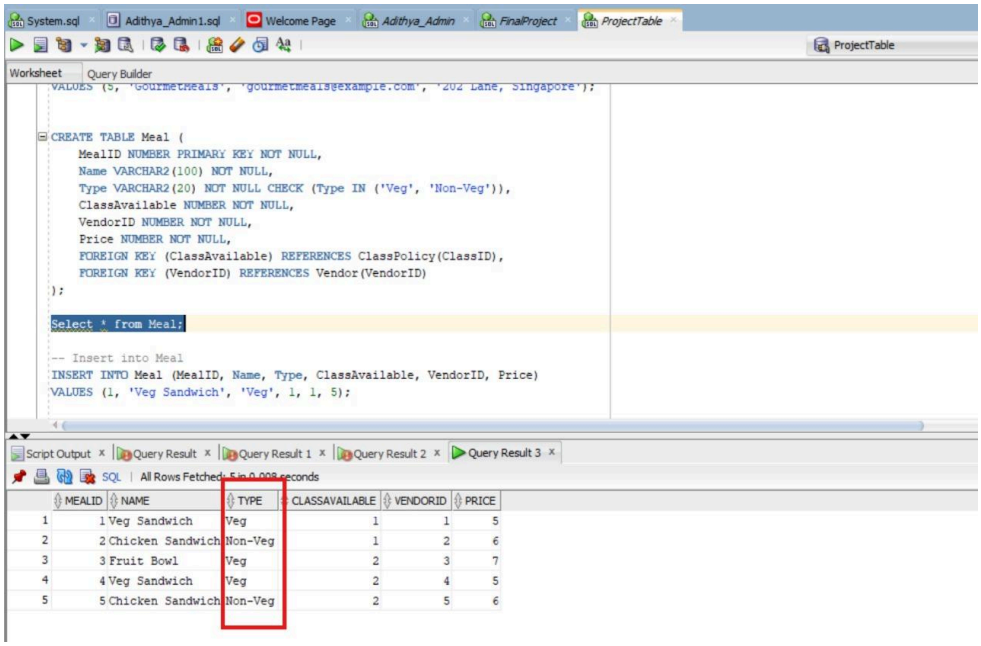
Business Rule Name: Maximum Baggage Weight Limit

Description: The weight of a baggage item must not exceed 60 units (e.g., kilograms).



Business Rule Name: Meal Type Restriction

Description: The type of meal provided must be either "Veg" or "Non-Veg.



RELATIONSHIPS/ASSOCIATIONS

Passenger to Booking:

- **One-to-Many relationship**: One passenger can have multiple bookings.
- Foreign Key: PassengerID in the Booking table links to PassengerID in the Passenger table.

Booking to Payment:

- **One-to-One relationship**: Each booking is associated with one payment.
- Foreign Key: BookingID in the Payment table links to BookingID in the Booking table.

Booking to Flight:

- **Many-to-One relationship**: Multiple bookings can be linked to one flight.
- Foreign Key: FlightID in the Booking table links to FlightID in the Flight table.

Flight to Airport:

- **Many-to-One relationship**: Each flight is associated with a departure and an arrival airport.
- Foreign Key: DepartureAirportID and ArrivalAirportID in the Flight table link to AirportID in the Airport table.

Passenger to Baggage:

- **One-to-Many relationship**: One passenger can have multiple pieces of baggage.
- Foreign Key: PassengerID in the Baggage table links to PassengerID in the Passenger table.

Flight to Baggage:

- **One-to-Many relationship**: One flight can have multiple pieces of baggage.
- Foreign Key: FlightID in the Baggage table links to FlightID in the Flight table.

Baggage to ClassPolicy:

- **Many-to-One relationship**: Multiple pieces of baggage can be associated with one class policy.
- Foreign Key: ClassID in the Baggage table links to ClassID in the ClassPolicy table.

ClassPolicy to BaggagePolicy:

- **One-to-Many relationship**: One class policy can have multiple baggage policies.
- Foreign Key: ClassID in the BaggagePolicy table links to ClassID in the ClassPolicy table.

Passenger to PassengerMeal:

- **One-to-Many relationship**: One passenger can have multiple meal selections.

- Foreign Key: PassengerID in the PassengerMeal table links to PassengerID in the Passenger table.

Meal to PassengerMeal:

- **One-to-Many relationship**: One meal can be selected by multiple passengers.
- Foreign Key: MealID in the PassengerMeal table links to MealID in the Meal table.

Vendor to Meal:

- **One-to-Many relationship**: One vendor can provide multiple meals.
- Foreign Key: VendorID in the Meal table links to VendorID in the Vendor table.

Meal to MealInventory:

- **One-to-Many relationship**: One meal can have multiple inventory entries.
- Foreign Key: MealID in the MealInventory table links to MealID in the Meal table.

Flight to InFlightSales:

- **One-to-Many relationship**: One flight can have multiple in-flight sales.
- Foreign Key: FlightID in the InFlightSales table links to FlightID in the Flight table.

Baggage to BaggageTracking:

- **One-to-Many relationship**: One piece of baggage can have multiple tracking records.
- Foreign Key: BaggageID in the BaggageTracking table links to BaggageID in the Baggage table.

Baggage to BaggageFees:

- **One-to-Many relationship**: One piece of baggage can be associated with multiple fees based on different class policies.

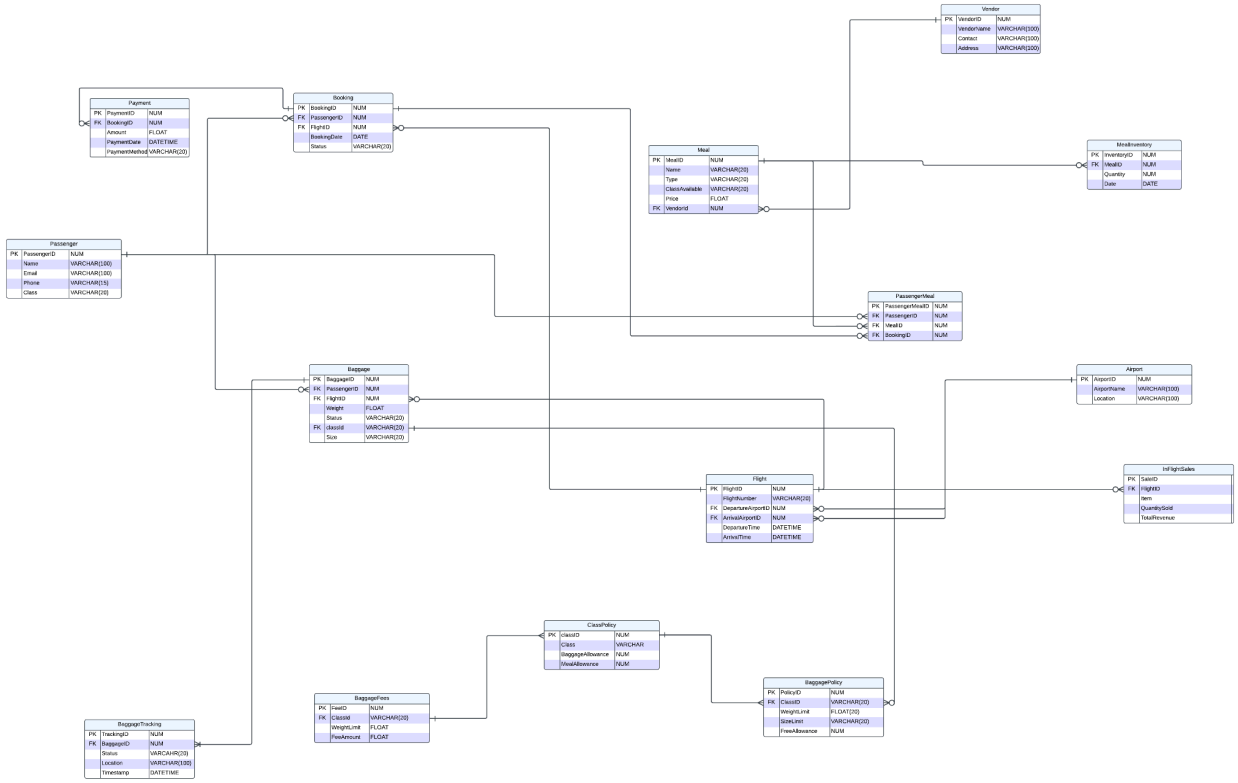
- Foreign Key: ClassID in the BaggageFees table links to ClassID in the Baggage table.

ER DIAGRAM - INITIAL

The initial ER Diagram visually represents the key entities and relationships required to manage airport baggage and catering operations.

Entities such as Passengers, Flights, Baggage, and Meals are interconnected through relationships, ensuring referential integrity.

Each relationship is supported by primary and foreign keys to enforce data consistency. The ER Diagram serves as the foundation for designing a scalable and normalized database.



NORMALIZATION

Normalization Example from Your Schema From UNF: A single table for passengers storing all details like bookings, flights, baggage, and meals.

To 1NF: Break into: Passenger: PassengerID, Name, Email, Phone, ClassBooking: BookingID, PassengerID, FlightID, BookingDate, Status Flight: FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID, etc.

To 2NF: Separate baggage policies, meals, and class-specific policies into distinct tables:BaggagePolicy: PolicyID, Class, WeightLimit, SizeLimit, FreeAllowanceClassPolicy: PolicyID, Class, BaggageAllowance, MealAllowance

To 3NF: Remove transitive dependencies: Move vendor details into a separate Vendor table linked to meals or other resources. Ensure fields like ClassAvailable in the Meal table directly depend on MealID.

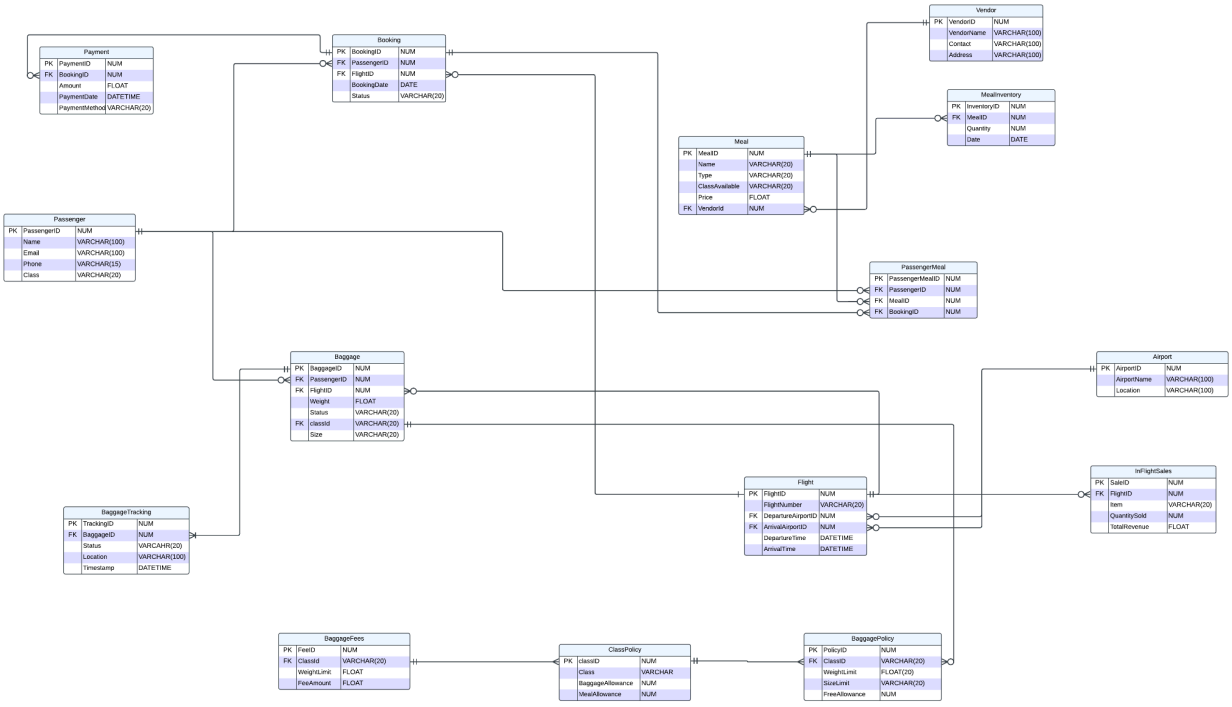
NORMALIZATION DECISIONS AND ACHIEVEMENTS

Decisions Made:

- Attributes were grouped into appropriate tables to avoid redundancy.
- Proper primary and foreign keys were assigned to maintain data integrity. Achievements:
- Improved data retrieval efficiency and reduced redundancy.
- Structured database design aligned with functional requirements.

ER DIAGRAM - FINAL

The final ER Diagram incorporates normalization adjustments and feedback from the design review process. It ensures a structured and logical flow of data, eliminating redundancy and addressing all identified requirements. Relationships between entities are clearly defined, ensuring seamless data integration and scalability.



PHYSICAL DATA STORE ORGANIZATION

Each table's structure was optimized for efficient storage and retrieval:

- Passenger: Stores passenger details, linked to ClassPolicy.
- Flight: Tracks flight schedules, connected to Airports and Bookings.
- Baggage: Monitors baggage status and attributes like weight and size.
- Meal: Stores meal details, linked to Vendors and PassengerMeal.
- Payment: Logs booking payments with constraints for valid methods.

Table Name	Column Name	Data Type	Constraints
ClassPolicy	ClassID	NUMBER	PRIMARY KEY, NOT NULL
	ClassName	VARCHAR2(20)	NOT NULL, UNIQUE
	BaggageAllowance	NUMBER	NOT NULL

	MealAllowance	NUMBER	NOT NULL
Passenger	PassengerID	NUMBER	PRIMARY KEY, NOT NULL
	Name	VARCHAR2(100)	NOT NULL
	Email	VARCHAR2(100)	NOT NULL, UNIQUE
	Phone	VARCHAR2(15)	NOT NULL, UNIQUE
	ClassID	NUMBER	NOT NULL, FOREIGN KEY (references ClassPolicy)
Airport	AirportID	NUMBER	PRIMARY KEY, NOT NULL
	AirportName	VARCHAR2(100)	NOT NULL
	Location	VARCHAR2(100)	NOT NULL
Flight	FlightID	NUMBER	PRIMARY KEY, NOT NULL
	FlightNumber	VARCHAR2(20)	NOT NULL, UNIQUE
	DepartureAirportID	NUMBER	NOT NULL, FOREIGN KEY (references Airport)
	ArrivalAirportID	NUMBER	NOT NULL, FOREIGN KEY (references Airport)
	DepartureTime	DATE	NOT NULL
	ArrivalTime	DATE	NOT NULL
Booking	BookingID	NUMBER	PRIMARY KEY, NOT NULL
	PassengerID	NUMBER	NOT NULL, FOREIGN KEY (references Passenger)
	FlightID	NUMBER	NOT NULL, FOREIGN KEY (references Flight)
	BookingDate	DATE	NOT NULL
	Status	VARCHAR2(20)	NOT NULL, CHECK (Status IN ('Confirmed', 'Canceled'))
BaggagePolicy	PolicyID	NUMBER	PRIMARY KEY, NOT NULL

	ClassID	NUMBER	NOT NULL, FOREIGN KEY (references ClassPolicy)
	WeightLimit	NUMBER	NOT NULL
	SizeLimit	VARCHAR2(50)	NOT NULL
	FreeAllowance	NUMBER	NOT NULL, CHECK (FreeAllowance >= 0)
BaggageFees	FeeID	NUMBER	PRIMARY KEY, NOT NULL
	ClassID	NUMBER	NOT NULL, FOREIGN KEY (references ClassPolicy)
	WeightLimit	NUMBER	NOT NULL
	FeeAmount	NUMBER	NOT NULL
Baggage	BaggageID	NUMBER	PRIMARY KEY, NOT NULL
	PassengerID	NUMBER	NOT NULL, FOREIGN KEY (references Passenger)
	FlightID	NUMBER	NOT NULL, FOREIGN KEY (references Flight)
	ClassID	NUMBER	NOT NULL, FOREIGN KEY (references ClassPolicy)
	Weight	NUMBER	NOT NULL, CHECK (Weight <= 60)
	SizeLimit	VARCHAR2(50)	NOT NULL
	Status	VARCHAR2(20)	NOT NULL, CHECK (Status IN ('Checked In', 'In Transit', 'Loaded on Flight', 'Delivered'))
Vendor	VendorID	NUMBER	PRIMARY KEY, NOT NULL
	VendorName	VARCHAR2(100)	NOT NULL
	Contact	VARCHAR2(100)	NOT NULL
	Address	VARCHAR2(200)	NOT NULL
Meal	MealID	NUMBER	PRIMARY KEY, NOT NULL

	Name	VARCHAR2(100)	NOT NULL
	Type	VARCHAR2(20)	NOT NULL, CHECK (Type IN ('Veg', 'Non-Veg'))
	ClassAvailable	NUMBER	NOT NULL, FOREIGN KEY (references ClassPolicy)
	VendorID	NUMBER	NOT NULL, FOREIGN KEY (references Vendor)
	Price	NUMBER	NOT NULL
MealInventory	InventoryID	NUMBER	PRIMARY KEY, NOT NULL
	MealID	NUMBER	NOT NULL, FOREIGN KEY (references Meal)
	Quantity	NUMBER	NOT NULL, CHECK (Quantity >= 0)
	Date	TIMESTAMP	NOT NULL
PassengerMeal	PassengerMealID	NUMBER	PRIMARY KEY, NOT NULL
	PassengerID	NUMBER	NOT NULL, FOREIGN KEY (references Passenger)
	MealID	NUMBER	NOT NULL, FOREIGN KEY (references Meal)
	BookingID	NUMBER	NOT NULL, FOREIGN KEY (references Booking)
Payment	PaymentID	NUMBER	PRIMARY KEY, NOT NULL
	BookingID	NUMBER	NOT NULL, FOREIGN KEY (references Booking)
	Amount	NUMBER	NOT NULL, CHECK (Amount > 0)
	PaymentDate	DATE	NOT NULL

	PaymentMethod	VARCHAR2(50)	NOT NULL, CHECK (PaymentMethod IN ('Credit Card', 'Debit Card', 'PayPal', 'Cash'))
InFlightSales	SaleID	NUMBER	PRIMARY KEY, NOT NULL
	FlightID	NUMBER	NOT NULL, FOREIGN KEY (references Flight)
	Item	VARCHAR2(100)	NOT NULL
	QuantitySold	NUMBER	NOT NULL, CHECK (QuantitySold >= 0)
	TotalRevenue	NUMBER(10, 2)	NOT NULL
Baggage Tracking	TrackingID	NUMBER	PRIMARY KEY, NOT NULL
	BaggageID	NUMBER	NOT NULL, FOREIGN KEY (references Baggage)
	Status	VARCHAR2(20)	NOT NULL, CHECK (Status IN ('Checked In', 'In Transit', 'Loaded on Flight', 'Delivered'))
	Location	VARCHAR2(100)	NOT NULL
	Timestamp	TIMESTAMP	NOT NULL

PROJECT SCRIPT

```
 -- Drop Tables in Reverse Order of Dependency
--DDL

DROP TABLE BaggageTracking CASCADE CONSTRAINTS;
DROP TABLE InFlightSales CASCADE CONSTRAINTS;
DROP TABLE Payment CASCADE CONSTRAINTS;
DROP TABLE MealInventory CASCADE CONSTRAINTS;
DROP TABLE PassengerMeal CASCADE CONSTRAINTS;
DROP TABLE Meal CASCADE CONSTRAINTS;
```

```
DROP TABLE Vendor CASCADE CONSTRAINTS;
DROP TABLE Baggage CASCADE CONSTRAINTS;
DROP TABLE BaggageFees CASCADE CONSTRAINTS;
DROP TABLE BaggagePolicy CASCADE CONSTRAINTS;
DROP TABLE Booking CASCADE CONSTRAINTS;
DROP TABLE Flight CASCADE CONSTRAINTS;
DROP TABLE Airport CASCADE CONSTRAINTS;
DROP TABLE Passenger CASCADE CONSTRAINTS;
DROP TABLE ClassPolicy CASCADE CONSTRAINTS;

CREATE TABLE ClassPolicy (
    ClassID NUMBER PRIMARY KEY NOT NULL,
    ClassName VARCHAR2(20) NOT NULL UNIQUE,
    BaggageAllowance NUMBER NOT NULL,
    MealAllowance NUMBER NOT NULL
);

CREATE TABLE Passenger (
    PassengerID NUMBER PRIMARY KEY NOT NULL,
    Name VARCHAR2(100) NOT NULL,
    Email VARCHAR2(100) NOT NULL UNIQUE,
    Phone VARCHAR2(15) NOT NULL UNIQUE,
    ClassID NUMBER NOT NULL,
    FOREIGN KEY (ClassID) REFERENCES ClassPolicy(ClassID)
);
CREATE TABLE Airport (
    AirportID NUMBER PRIMARY KEY NOT NULL,
    AirportName VARCHAR2(100) NOT NULL,
    Location VARCHAR2(100) NOT NULL
);
CREATE TABLE Flight (
    FlightID NUMBER PRIMARY KEY NOT NULL,
    FlightNumber VARCHAR2(20) NOT NULL UNIQUE,
    DepartureAirportID NUMBER NOT NULL,
    ArrivalAirportID NUMBER NOT NULL,
    DepartureTime DATE NOT NULL,
    ArrivalTime DATE NOT NULL,
    FOREIGN KEY (DepartureAirportID) REFERENCES Airport(AirportID),
    FOREIGN KEY (ArrivalAirportID) REFERENCES Airport(AirportID)
);

CREATE TABLE Booking (
    BookingID NUMBER PRIMARY KEY NOT NULL,
    PassengerID NUMBER NOT NULL,
```

```
    FlightID NUMBER NOT NULL,
    BookingDate DATE NOT NULL,
    Status VARCHAR2(20) NOT NULL CHECK (Status IN ('Confirmed', 'Canceled')),
    FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID),
    FOREIGN KEY (FlightID) REFERENCES Flight(FlightID)
);

CREATE TABLE BaggagePolicy (
    PolicyID NUMBER PRIMARY KEY NOT NULL,
    ClassID NUMBER NOT NULL,
    WeightLimit NUMBER NOT NULL,
    SizeLimit VARCHAR2(50) NOT NULL,
    FreeAllowance NUMBER NOT NULL CHECK (FreeAllowance >= 0),
    FOREIGN KEY (ClassID) REFERENCES ClassPolicy(ClassID)
);


CREATE TABLE BaggageFees (
    FeeID NUMBER PRIMARY KEY NOT NULL,
    ClassID NUMBER NOT NULL,
    WeightLimit NUMBER NOT NULL,
    FeeAmount NUMBER NOT NULL,
    FOREIGN KEY (ClassID) REFERENCES ClassPolicy(ClassID)
);
CREATE TABLE Baggage (
    BaggageID NUMBER PRIMARY KEY NOT NULL,
    PassengerID NUMBER NOT NULL,
    FlightID NUMBER NOT NULL,
    ClassID NUMBER NOT NULL,
    Weight NUMBER NOT NULL CHECK (Weight <= 60),
    SizeLimit VARCHAR2(50) NOT NULL,
    Status VARCHAR2(20) NOT NULL CHECK (Status IN ('Checked In', 'In Transit', 'Loaded on
Flight', 'Delivered')),
    FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID),
    FOREIGN KEY (FlightID) REFERENCES Flight(FlightID),
    FOREIGN KEY (ClassID) REFERENCES ClassPolicy(ClassID)
);
 CREATE TABLE Vendor (
    VendorID NUMBER PRIMARY KEY NOT NULL,
    VendorName VARCHAR2(100) NOT NULL,
    Contact VARCHAR2(100) NOT NULL,
    Address VARCHAR2(200) NOT NULL
);
```

```
CREATE TABLE Meal (
    MealID NUMBER PRIMARY KEY NOT NULL,
    Name VARCHAR2(100) NOT NULL,
    Type VARCHAR2(20) NOT NULL CHECK (Type IN ('Veg', 'Non-Veg')),
    ClassAvailable NUMBER NOT NULL,
    VendorID NUMBER NOT NULL,
    Price NUMBER NOT NULL,
    FOREIGN KEY (ClassAvailable) REFERENCES ClassPolicy(ClassID),
    FOREIGN KEY (VendorID) REFERENCES Vendor(VendorID)
);


CREATE TABLE MealInventory (
    InventoryID NUMBER PRIMARY KEY NOT NULL,
    MealID NUMBER NOT NULL,
    Quantity NUMBER NOT NULL CHECK (Quantity >= 0),
    "Date" TIMESTAMP NOT NULL,
    FOREIGN KEY (MealID) REFERENCES Meal(MealID)
);


CREATE TABLE PassengerMeal (
    PassengerMealID NUMBER PRIMARY KEY NOT NULL,
    PassengerID NUMBER NOT NULL,
    MealID NUMBER NOT NULL,
    BookingID NUMBER NOT NULL,
    FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID),
    FOREIGN KEY (MealID) REFERENCES Meal(MealID),
    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID)
);


CREATE TABLE Payment (
    PaymentID NUMBER PRIMARY KEY NOT NULL,
    BookingID NUMBER NOT NULL,
    Amount NUMBER NOT NULL CHECK (Amount > 0),
    PaymentDate DATE NOT NULL,
    PaymentMethod VARCHAR2(50) NOT NULL CHECK (PaymentMethod IN ('Credit Card',
'Debit Card', 'PayPal', 'Cash')),
    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID)
);
```

```
CREATE TABLE InFlightSales (
   SaleID NUMBER PRIMARY KEY NOT NULL,
   FlightID NUMBER NOT NULL,
   Item VARCHAR2(100) NOT NULL,
   QuantitySold NUMBER NOT NULL CHECK (QuantitySold >= 0),
   TotalRevenue NUMBER(10, 2) NOT NULL,
   FOREIGN KEY (FlightID) REFERENCES Flight(FlightID)
);


CREATE TABLE BaggageTracking (
   TrackingID NUMBER PRIMARY KEY NOT NULL,
   BaggageID NUMBER NOT NULL,
   Status VARCHAR2(20) NOT NULL CHECK (Status IN ('Checked In', 'In Transit', 'Loaded on
Flight', 'Delivered')),
   Location VARCHAR2(100) NOT NULL,
   Timestamp TIMESTAMP NOT NULL,
   FOREIGN KEY (BaggageID) REFERENCES Baggage(BaggageID)
);

--DML

-- INSERT INTO CLASS POLICY
INSERT INTO ClassPolicy (ClassID, ClassName, BaggageAllowance, MealAllowance)
VALUES (1, 'Economy', 46, 1);

INSERT INTO ClassPolicy (ClassID, ClassName, BaggageAllowance, MealAllowance)
VALUES (2, 'Business', 60, 2);



-- INSERT INTO PASSENGER
INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (101, 'John Doe', 'john.doe@example.com', '1234567890', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (102, 'Jane Smith', 'jane.smith@example.com', '0987654321', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (103, 'Robert Brown', 'robert.brown@example.com', '1122334455', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (104, 'Mary Johnson', 'mary.johnson@example.com', '2233445566', 2);
```

```
INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (105, 'William Lee', 'william.lee@example.com', '3344556677', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (106, 'Eva Green', 'eva.green@example.com', '4455667788', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (107, 'Liam Neeson', 'liam.neeson@example.com', '5566778899', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (108, 'Scarlett Johansson', 'scarlett.johansson@example.com', '6677889900', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (109, 'Chris Hemsworth', 'chris.hemsworth@example.com', '7788990011', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (110, 'Tom Hiddleston', 'tom.hiddleston@example.com', '8899001122', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (111, 'Natalie Portman', 'natalie.portman@example.com', '9900112233', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (112, 'George Clooney', 'george.clooney1@example.com', '7788991122', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (113, 'Morgan Freeman', 'morgan.freeman113@example.com', '8000011223', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (114, 'Angelina Jolie', 'angelina.jolie114@example.com', '8000112234', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (115, 'Brad Pitt', 'brad.pitt115@example.com', '8000223345', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (116, 'Tom Cruise', 'tom.cruise116@example.com', '8000334456', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (117, 'Johnny Depp', 'johnny.depp117@example.com', '8000445567', 1);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (118, 'Emma Watson', 'emma.watson118@example.com', '8000556678', 2);

INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (119, 'Keanu Reeves', 'keanu.reeves119@example.com', '8000667789', 1);
```

```
INSERT INTO Passenger (PassengerID, Name, Email, Phone, ClassID)
VALUES (120, 'Charlize Theron', 'charlize.theron120@example.com', '8000778890', 2);

-- INSERT INTO AIRPORT
INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (1, 'JFK Airport', 'New York, USA');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (2, 'LAX Airport', 'Los Angeles, USA');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (3, 'Heathrow Airport', 'London, UK');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (4, 'Dubai International Airport', 'Dubai, UAE');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (5, 'Changi Airport', 'Singapore');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (6, 'Sydney Airport', 'Sydney, Australia');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (7, 'Tokyo Narita Airport', 'Tokyo, Japan');


INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (8, 'London Heathrow Airport', 'London, UK');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (9, 'Los Angeles International Airport', 'Los Angeles, USA');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (10, 'Dubai International Airport', 'Dubai, UAE');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (11, 'Paris Charles de Gaulle Airport', 'Paris, France');

INSERT INTO Airport (AirportID, AirportName, Location)
VALUES (12, 'Singapore Changi Airport', 'Singapore');

 --INSERT INTO FLIGHT
```

```
INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (201, 'AA101', 1, 2, TO_DATE('2024-12-01 08:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-01 11:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (202, 'BA202', 3, 4, TO_DATE('2024-12-02 10:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-02 14:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (203, 'DL303', 2, 3, TO_DATE('2024-12-03 12:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-03 16:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (204, 'EK404', 4, 5, TO_DATE('2024-12-04 14:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-04 18:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (205, 'SQ505', 5, 1, TO_DATE('2024-12-05 16:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-05 20:00:00', 'YYYY-MM-DD HH24:MI:SS'));


INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (206, 'QF506', 1, 6, TO_DATE('2024-12-06 07:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-06 10:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (207, 'JL701', 7, 2, TO_DATE('2024-12-07 09:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-07 13:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (208, 'BA249', 8, 2, TO_DATE('2024-12-06 13:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-06 18:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
```

```
VALUES (209, 'AA900', 9, 3, TO_DATE('2024-12-07 11:00:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-07 16:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (210, 'EK215', 10, 4, TO_DATE('2024-12-08 07:30:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-08 12:30:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (211, 'AF335', 11, 5, TO_DATE('2024-12-09 14:45:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-09 19:30:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO Flight (FlightID, FlightNumber, DepartureAirportID, ArrivalAirportID,
DepartureTime, ArrivalTime)
VALUES (212, 'SQ323', 12, 1, TO_DATE('2024-12-10 10:15:00', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2024-12-10 14:00:00', 'YYYY-MM-DD HH24:MI:SS'));


-- INSERT INTO BOOKING
INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (301, 101, 201, TO_DATE('2024-11-25', 'YYYY-MM-DD'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (302, 102, 202, TO_DATE('2024-11-26', 'YYYY-MM-DD'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (303, 103, 203, TO_DATE('2024-11-27', 'YYYY-MM-DD'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (304, 104, 204, TO_DATE('2024-11-28', 'YYYY-MM-DD'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (305, 105, 205, TO_DATE('2024-11-29', 'YYYY-MM-DD'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (306, 106, 206, TO_DATE('2024-12-01', 'YYYY-MM-DD'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (307, 107, 207, TO_DATE('2024-12-02', 'YYYY-MM-DD'), 'Canceled');


INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (308, 108, 208, TO_DATE('01-12-24', 'DD-MM-YY'), 'Confirmed');
```

```
INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (309, 109, 209, TO_DATE('02-12-24', 'DD-MM-YY'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (310, 110, 210, TO_DATE('03-12-24', 'DD-MM-YY'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (311, 111, 211, TO_DATE('04-12-24', 'DD-MM-YY'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (312, 112, 201, TO_DATE('05-12-24', 'DD-MM-YY'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (313, 113, 202, TO_DATE('06-12-24', 'DD-MM-YY'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (314, 114, 203, TO_DATE('07-12-24', 'DD-MM-YY'), 'Confirmed');
INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (315, 115, 204, TO_DATE('08-12-24', 'DD-MM-YY'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (316, 116, 205, TO_DATE('09-12-24', 'DD-MM-YY'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (317, 117, 206, TO_DATE('10-12-24', 'DD-MM-YY'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (318, 118, 207, TO_DATE('11-12-24', 'DD-MM-YY'), 'Confirmed');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (319, 119, 208, TO_DATE('12-12-24', 'DD-MM-YY'), 'Canceled');

INSERT INTO Booking (BookingID, PassengerID, FlightID, BookingDate, Status)
VALUES (320, 120, 209, TO_DATE('13-12-24', 'DD-MM-YY'), 'Confirmed');


-- INSERT INTO BAGGAGEPOLICY
INSERT INTO BaggagePolicy (PolicyID, ClassID, WeightLimit, SizeLimit, FreeAllowance)
VALUES (1, 1, 46, '56x45x25', 1);

INSERT INTO BaggagePolicy (PolicyID, ClassID, WeightLimit, SizeLimit, FreeAllowance)
VALUES (2, 2, 60, '60x50x30', 2);
```

```
 -- INSERT INTO BAGGAGEFEES
INSERT INTO BaggageFees (FeeID, ClassID, WeightLimit, FeeAmount)
VALUES (1, 1, 30, 50);

INSERT INTO BaggageFees (FeeID, ClassID, WeightLimit, FeeAmount)
VALUES (2, 2, 50, 100);

 --INSERT INTO BAGGAGE

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (601, 101, 201, 1, 28, '56x45x25', 'Checked In');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (602, 102, 202, 2, 48, '60x50x30', 'In Transit');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (603, 103, 203, 1, 29, '57x45x26', 'Loaded on Flight');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (604, 104, 204, 2, 49, '59x49x29', 'Delivered');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (605, 105, 205, 1, 32, '58x47x28', 'Checked In');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (606, 106, 206, 2, 52, '60x50x30', 'Checked In');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (607, 107, 207, 1, 30, '55x45x25', 'In Transit');


INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (608, 108, 208, 1, 33, '58x47x28', 'Checked In');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (609, 109, 209, 2, 50, '60x50x30', 'In Transit');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (610, 110, 210, 1, 34, '57x46x27', 'Loaded on Flight');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (611, 111, 211, 2, 51, '59x48x28', 'Delivered');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
```

```
VALUES (612, 112, 201, 1, 29, '56x45x25', 'Checked In');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (613, 113, 202, 2, 47, '60x50x30', 'In Transit');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (614, 114, 203, 1, 30, '57x45x26', 'Loaded on Flight');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (615, 115, 204, 2, 49, '59x49x29', 'Delivered');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (616, 116, 205, 1, 32, '58x47x28', 'Checked In');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (617, 117, 206, 2, 50, '60x50x30', 'In Transit');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (618, 118, 207, 1, 31, '57x46x27', 'Loaded on Flight');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (619, 119, 208, 2, 52, '59x48x28', 'Delivered');

INSERT INTO Baggage (BaggageID, PassengerID, FlightID, ClassID, Weight, SizeLimit, Status)
VALUES (620, 120, 209, 1, 33, '58x47x28', 'Checked In');


--INSERT INTO VENDOR
INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
VALUES (1, 'FoodCo', 'foodco@example.com', '123 Street, New York');

INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
VALUES (2, 'MealCorp', 'mealcorp@example.com', '456 Avenue, Los Angeles');

INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
VALUES (3, 'SnackExpress', 'snackexpress@example.com', '789 Boulevard, London');

INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
VALUES (4, 'AirBites', 'airbites@example.com', '101 Road, Dubai');

INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
VALUES (5, 'GourmetMeals', 'gourmetmeals@example.com', '202 Lane, Singapore');

INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
```

```
VALUES (6, 'McDonald', 'MacDonald@example.com', '202 Lane, Singapore');

INSERT INTO Vendor (VendorID, VendorName, Contact, Address)
VALUES (7, 'KFC', 'KFC@example.com', '203 dupont, Toronto');

 -- INSERT INTO MEAL
INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (1, 'Veg Sandwich', 'Veg', 1, 1, 5);

INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (2, 'Chicken Sandwich', 'Non-Veg', 1, 2, 6);

INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (3, 'Fruit Bowl', 'Veg', 2, 3, 7);

INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (4, 'Veg Sandwich', 'Veg', 2, 4, 5);

INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (5, 'Chicken Sandwich', 'Non-Veg', 2, 5, 6);

 INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (6, 'Vegetarian Pasta', 'Veg', 1, 6, 7);

INSERT INTO Meal (MealID, Name, Type, ClassAvailable, VendorID, Price)
VALUES (7, 'Beef Stew', 'Non-Veg', 2, 7, 8);

 -- INSERT INTO MEALINVENTORY
INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
VALUES (701, 1, 100, TO_TIMESTAMP('2024-11-25 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
VALUES (702, 2, 50, TO_TIMESTAMP('2024-11-26 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
VALUES (703, 3, 70, TO_TIMESTAMP('2024-11-27 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
VALUES (704, 4, 120, TO_TIMESTAMP('2024-11-28 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
VALUES (705, 5, 60, TO_TIMESTAMP('2024-11-29 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
```

```
VALUES (706, 6, 90, TO_TIMESTAMP('2024-12-01 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO MealInventory (InventoryID, MealID, Quantity, "Date")
VALUES (707, 7, 40, TO_TIMESTAMP('2024-12-02 08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

 -- INSERT INTO PASSENGERMEAL
INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (801, 101, 1, 301);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (802, 102, 2, 302);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (803, 103, 3, 303);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (804, 104, 4, 304);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (805, 105, 5, 305);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (806, 106, 6, 306);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (807, 107, 7, 307);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (808, 108, 1, 308);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (809, 109, 2, 309);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (810, 110, 3, 310);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (811, 111, 4, 311);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (815, 115, 1, 315);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (816, 116, 2, 316);
```

```
INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (817, 117, 3, 317);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (818, 118, 4, 318);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (819, 119, 5, 319);

INSERT INTO PassengerMeal (PassengerMealID, PassengerID, MealID, BookingID)
VALUES (820, 120, 6, 320);


-- INSERT INTO PAYMENT
INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (901, 301, 200, TO_DATE('2024-11-25', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (902, 302, 150, TO_DATE('2024-11-26', 'YYYY-MM-DD'), 'Debit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (903, 303, 250, TO_DATE('2024-11-27', 'YYYY-MM-DD'), 'Cash');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (904, 304, 300, TO_DATE('2024-11-28', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (905, 305, 350, TO_DATE('2024-11-29', 'YYYY-MM-DD'), 'PayPal');


INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (906, 306, 270, TO_DATE('2024-12-01', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (907, 307, 180, TO_DATE('2024-12-02', 'YYYY-MM-DD'), 'Debit Card');


INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (908, 308, 220, TO_DATE('2024-12-03', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (909, 309, 210, TO_DATE('2024-12-04', 'YYYY-MM-DD'), 'Debit Card');
```

```
INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (910, 310, 260, TO_DATE('2024-12-05', 'YYYY-MM-DD'), 'Cash');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (911, 311, 280, TO_DATE('2024-12-06', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (912, 312, 320, TO_DATE('2024-12-07', 'YYYY-MM-DD'), 'PayPal');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (913, 313, 230, TO_DATE('2024-12-08', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (914, 314, 180, TO_DATE('2024-12-09', 'YYYY-MM-DD'), 'Debit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (915, 315, 200, TO_DATE('2024-12-10', 'YYYY-MM-DD'), 'Cash');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (916, 316, 250, TO_DATE('2024-12-11', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (917, 317, 190, TO_DATE('2024-12-12', 'YYYY-MM-DD'), 'Debit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (918, 318, 270, TO_DATE('2024-12-13', 'YYYY-MM-DD'), 'PayPal');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (919, 319, 300, TO_DATE('2024-12-14', 'YYYY-MM-DD'), 'Credit Card');

INSERT INTO Payment (PaymentID, BookingID, Amount, PaymentDate, PaymentMethod)
VALUES (920, 320, 350, TO_DATE('2024-12-15', 'YYYY-MM-DD'), 'Cash');


--INSERT INTO INFLIGHTSALES
INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1001, 201, 'Snacks', 100, 500);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1002, 202, 'Drinks', 80, 240);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1003, 203, 'Magazines', 50, 150);
```

```
INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1004, 204, 'Headsets', 120, 600);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1005, 205, 'Toys', 60, 300);


INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1006, 206, 'Headphones', 90, 450);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1007, 207, 'Magazines', 60, 180);


INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1008, 201, 'Snacks', 110, 550);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1009, 202, 'Drinks', 90, 270);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1010, 203, 'Magazines', 60, 180);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1011, 204, 'Headsets', 130, 650);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1012, 205, 'Toys', 70, 350);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1013, 206, 'Headphones', 100, 500);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1014, 207, 'Magazines', 80, 240);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1015, 208, 'Snacks', 120, 600);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1016, 209, 'Drinks', 100, 300);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1017, 210, 'Magazines', 70, 210);
```

```
INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1018, 211, 'Headsets', 140, 700);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1019, 212, 'Toys', 80, 400);


INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1020, 201, 'Headphones', 110, 550);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1021, 202, 'Magazines', 90, 270);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1022, 203, 'Snacks', 130, 650);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1023, 204, 'Drinks', 110, 330);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1024, 205, 'Magazines', 80, 240);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1025, 206, 'Headsets', 150, 750);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1026, 207, 'Toys', 90, 450);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1027, 208, 'Headphones', 120, 600);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1028, 209, 'Magazines', 100, 300);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1029, 210, 'Snacks', 140, 700);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1030, 211, 'Drinks', 120, 360);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1031, 212, 'Magazines', 90, 270);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
```

```
VALUES (1032, 201, 'Headsets', 110, 550);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1033, 202, 'Toys', 100, 500);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1034, 203, 'Headphones', 120, 600);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1035, 204, 'Magazines', 130, 390);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1036, 205, 'Snacks', 100, 500);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1037, 206, 'Drinks', 90, 270);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1038, 207, 'Magazines', 110, 330);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1039, 208, 'Headsets', 120, 600);

INSERT INTO InFlightSales (SaleID, FlightID, Item, QuantitySold, TotalRevenue)
VALUES (1040, 209, 'Toys', 130, 650);

--INSERT INTO BAGGAGE TRACKING

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1101, 601, 'Checked In', 'JFK Terminal 1', TO_TIMESTAMP('2024-12-01 08:15:00',
'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1102, 602, 'In Transit', 'LAX Terminal 2', TO_TIMESTAMP('2024-12-02 10:15:00',
'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1103, 603, 'Loaded on Flight', 'Heathrow Terminal 5', TO_TIMESTAMP('2024-12-03
12:15:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1104, 604, 'Delivered', 'Dubai Terminal 3', TO_TIMESTAMP('2024-12-04 14:15:00',
'YYYY-MM-DD HH24:MI:SS'));
```

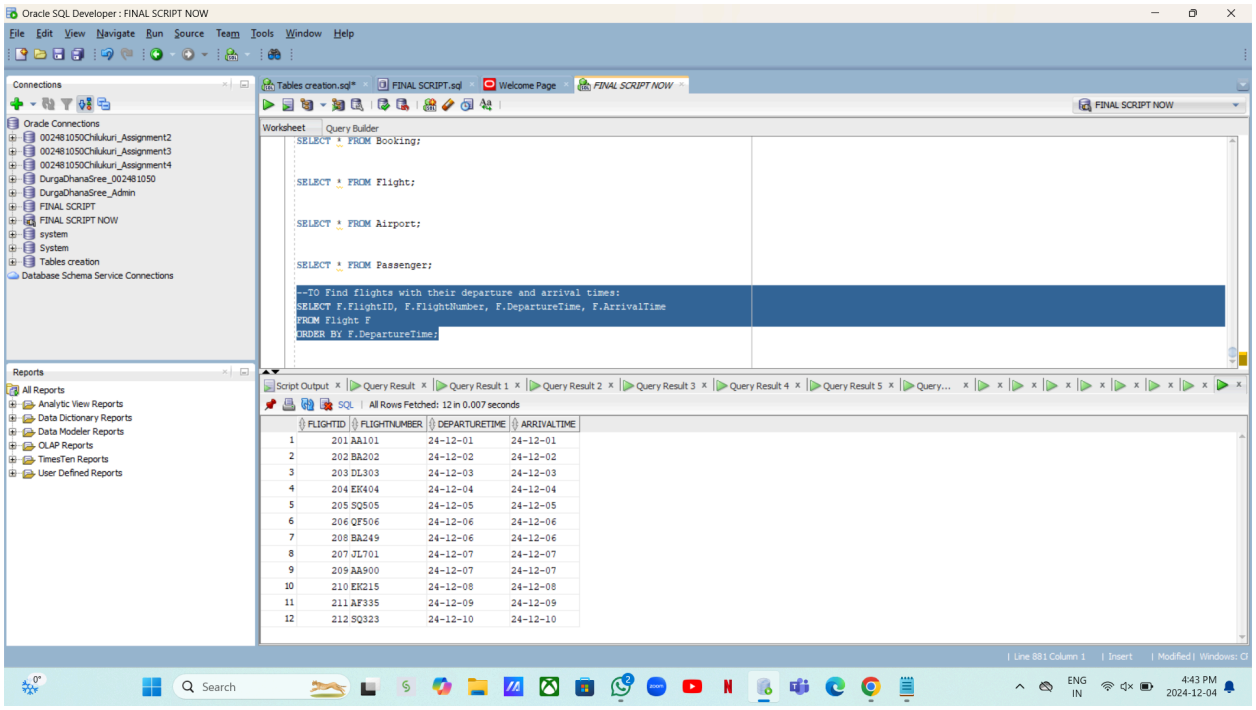
```
INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1105, 605, 'Checked In', 'Changi Terminal 2', TO_TIMESTAMP('2024-12-05 16:15:00',
'YYYY-MM-DD HH24:MI:SS'));


INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1106, 606, 'Checked In', 'JFK Terminal 2', TO_TIMESTAMP('2024-12-06 07:30:00',
'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1107, 607, 'In Transit', 'Tokyo Narita Terminal 3', TO_TIMESTAMP('2024-12-07
09:30:00', 'YYYY-MM-DD HH24:MI:SS'));


INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1108, 608, 'In Transit', 'London Heathrow Terminal 5', TO_TIMESTAMP('2024-12-07
10:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1109, 609, 'Checked In', 'Dubai International Terminal 2',
TO_TIMESTAMP('2024-12-08 09:15:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1110, 610, 'In Transit', 'Los Angeles International Terminal 1',
TO_TIMESTAMP('2024-12-09 07:30:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1111, 611, 'Checked In', 'Tokyo Narita Terminal 3', TO_TIMESTAMP('2024-12-10
08:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1112, 612, 'In Transit', 'JFK Terminal 2', TO_TIMESTAMP('2024-12-11 09:30:00',
'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1113, 613, 'Checked In', 'London Heathrow Terminal 5', TO_TIMESTAMP('2024-12-12
07:45:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1114, 614, 'In Transit', 'Dubai International Terminal 2', TO_TIMESTAMP('2024-12-13
08:15:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
```

```
VALUES (1115, 615, 'Checked In', 'Los Angeles International Terminal 1',
TO_TIMESTAMP('2024-12-14 09:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1116, 616, 'In Transit', 'Tokyo Narita Terminal 3', TO_TIMESTAMP('2024-12-15
07:30:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1117, 617, 'Checked In', 'JFK Terminal 2', TO_TIMESTAMP('2024-12-16 08:00:00',
'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1118, 618, 'In Transit', 'London Heathrow Terminal 5', TO_TIMESTAMP('2024-12-17
09:00:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1119, 619, 'Checked In', 'Dubai International Terminal 2',
TO_TIMESTAMP('2024-12-18 07:45:00', 'YYYY-MM-DD HH24:MI:SS'));

INSERT INTO BaggageTracking (TrackingID, BaggageID, Status, Location, Timestamp)
VALUES (1120, 620, 'In Transit', 'Los Angeles International Terminal 1',
TO_TIMESTAMP('2024-12-19 08:15:00', 'YYYY-MM-DD HH24:MI:SS'));
```

COMPLEX QUERIES AND JOINS

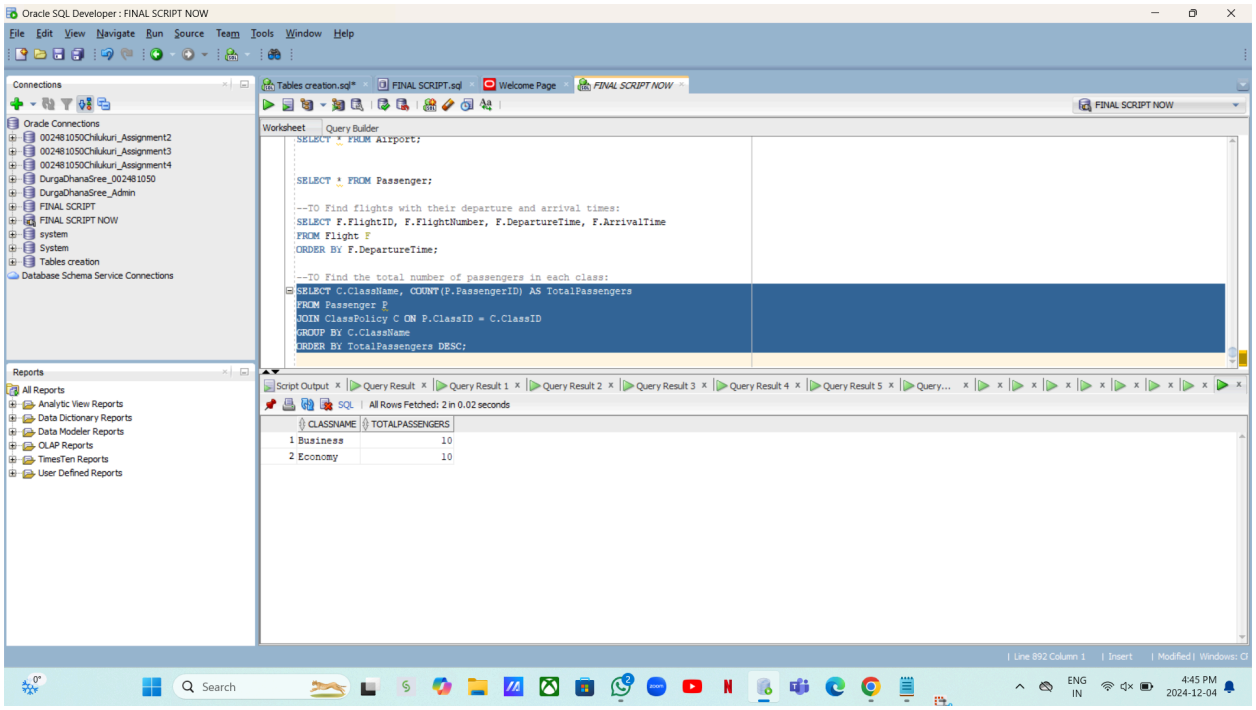
1)TO Find flights with their departure and arrival times:

```
SELECT F.FlightID, F.FlightNumber, F.DepartureTime, F.ArrivalTime
FROM Flight F
ORDER BY F.DepartureTime;
```



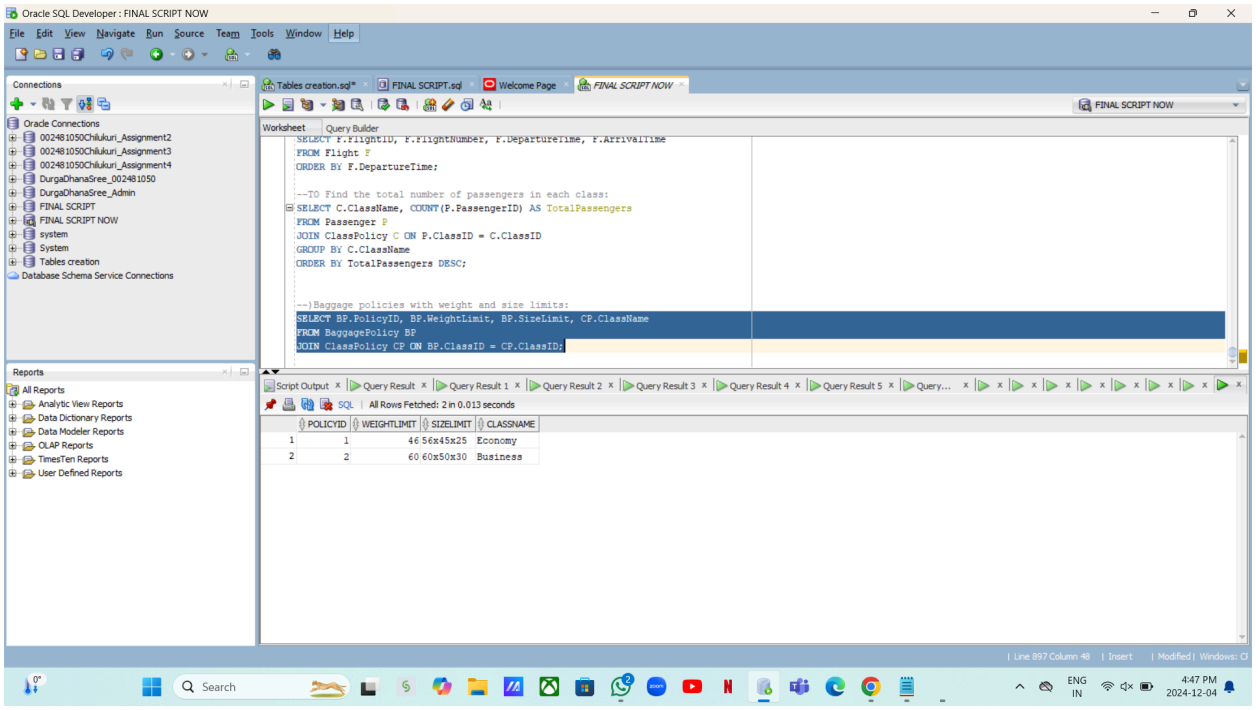
2)TO Find the total number of passengers in each class:

```
SELECT C.ClassName, COUNT(P.PassengerID) AS TotalPassengers
FROM Passenger P
JOIN ClassPolicy C ON P.ClassID = C.ClassID
GROUP BY C.ClassName
ORDER BY TotalPassengers DESC;
```



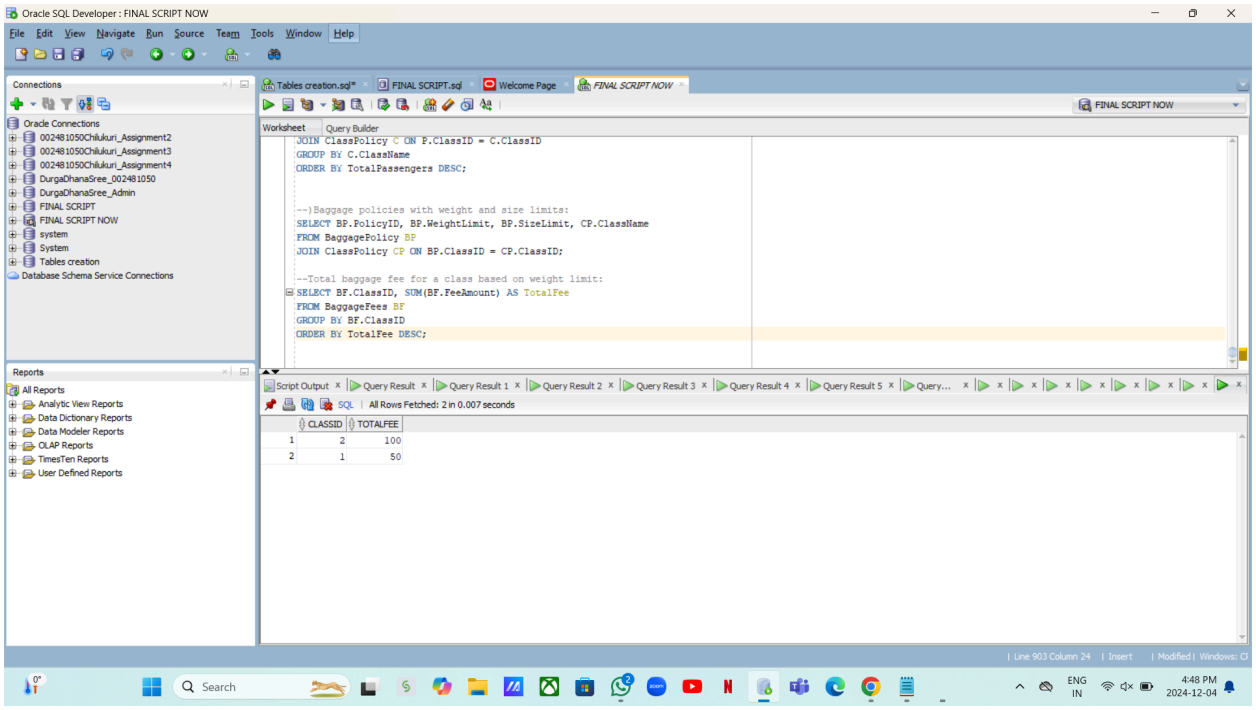
3)Baggage policies with weight and size limits:

SELECT BP.PolicyID, BP.WeightLimit, BP.SizeLimit, CP.ClassName
FROM BaggagePolicy BP
JOIN ClassPolicy CP ON BP.ClassID = CP.ClassID;



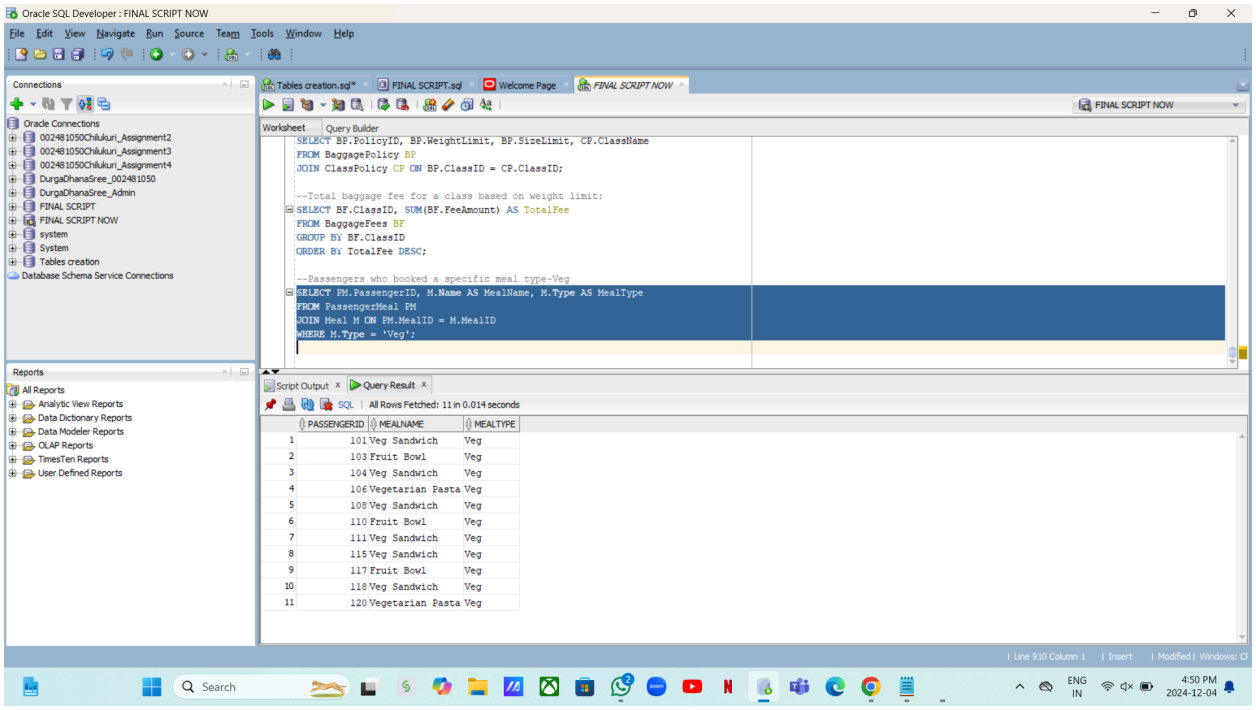
4)Total baggage fee for a class based on weight limit:

SELECT BF.ClassID, SUM(BF.FeeAmount) AS TotalFee
FROM BaggageFees BF
GROUP BY BF.ClassID
ORDER BY TotalFee DESC;



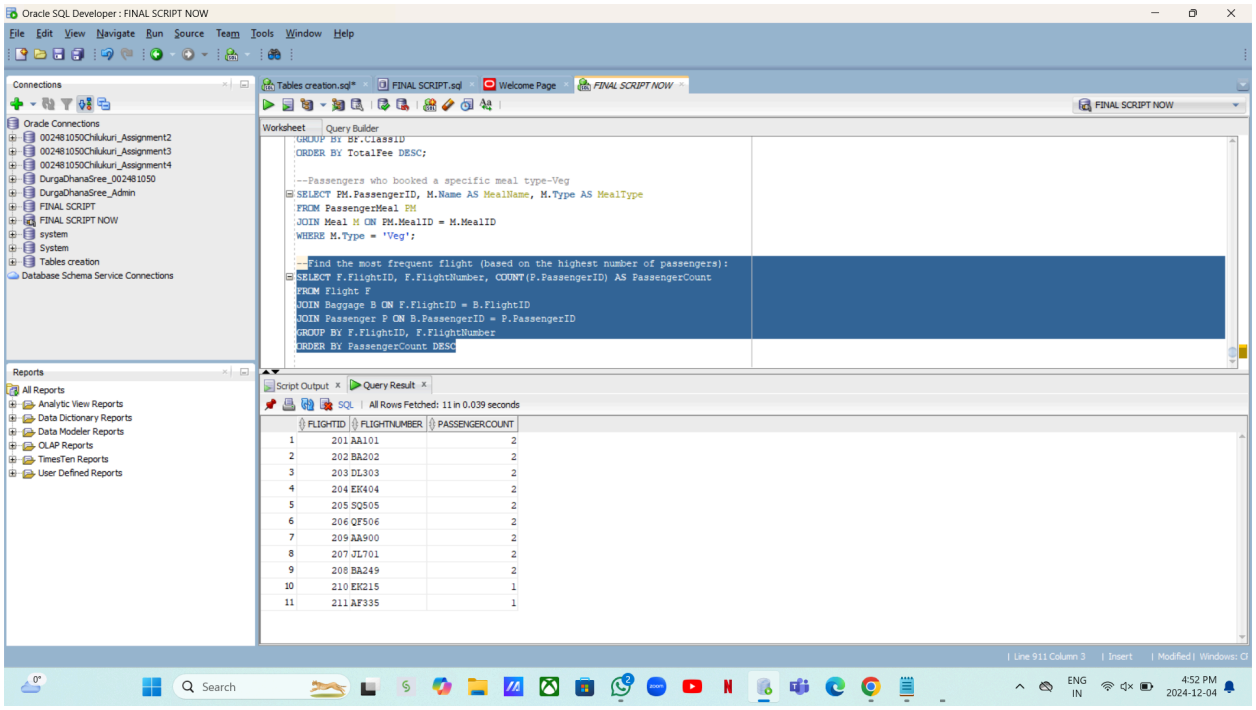
5)Passengers who booked a specific meal type-Veg

```
SELECT PM.PassengerID, M.Name AS MealName, M.Type AS MealType
FROM PassengerMeal PM
JOIN Meal M ON PM.MealID = M.MealID
WHERE M.Type = 'Veg';
```



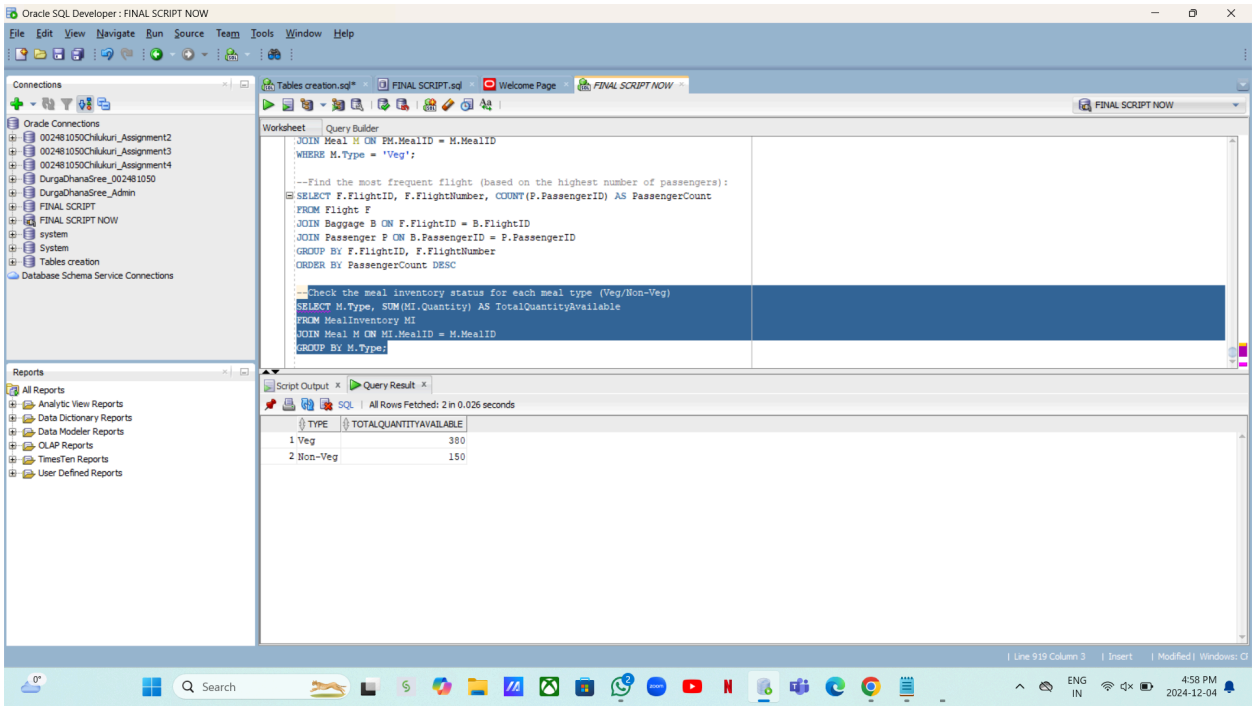
6)Find the most frequent flight (based on the highest number of passengers):

SELECT F.FlightID, F.FlightNumber, COUNT(P.PassengerID) AS PassengerCount
FROM Flight F
JOIN Baggage B ON F.FlightID = B.FlightID
JOIN Passenger P ON B.PassengerID = P.PassengerID
GROUP BY F.FlightID, F.FlightNumber
ORDER BY PassengerCount DESC



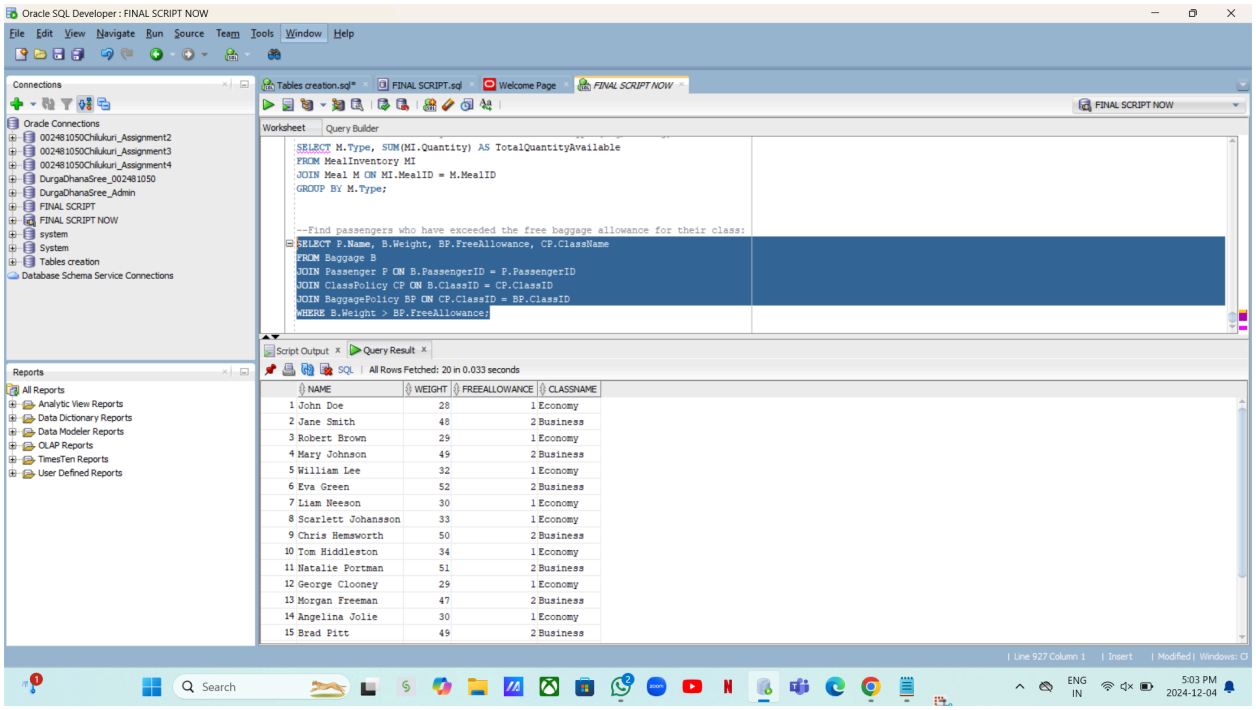
7)Check the meal inventory status for each meal type (Veg/Non-Veg)

```
SELECT M.Type, SUM(MI.Quantity) AS TotalQuantityAvailable
FROM MealInventory MI
JOIN Meal M ON MI.MealID = M.MealID
GROUP BY M.Type;
```



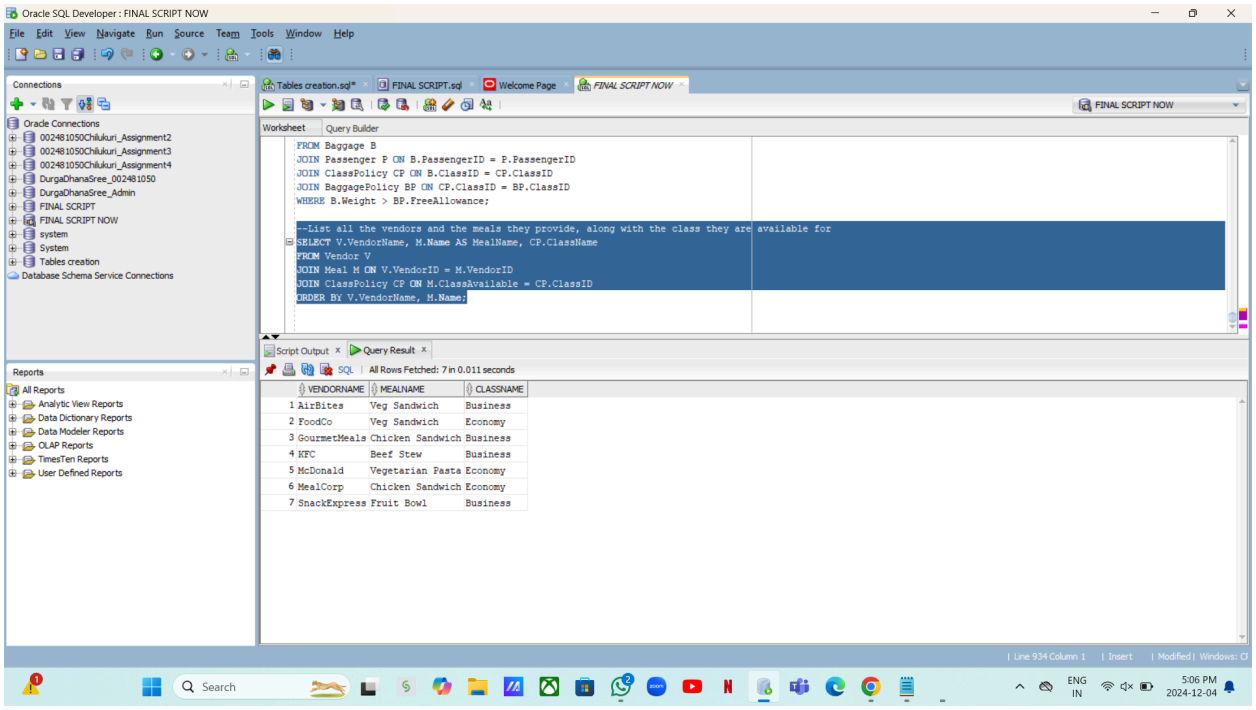
8)Find passengers who have exceeded the free baggage allowance for their class:

```
SELECT P.Name, B.Weight, BP.FreeAllowance, CP.ClassName
FROM Baggage B
JOIN Passenger P ON B.PassengerID = P.PassengerID
JOIN ClassPolicy CP ON B.ClassID = CP.ClassID
JOIN BaggagePolicy BP ON CP.ClassID = BP.ClassID
WHERE B.Weight > BP.FreeAllowance;
```



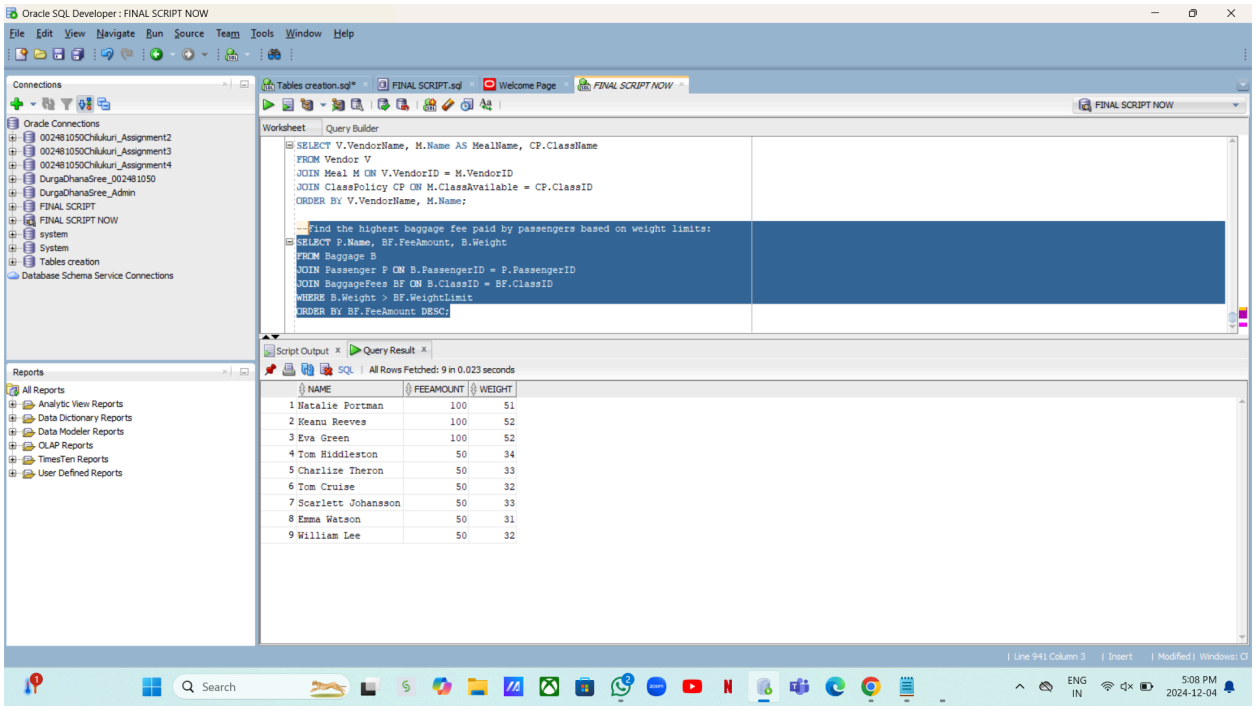
9)List all the vendors and the meals they provide, along with the class they are available for

```
SELECT V.VendorName, M.Name AS MealName, CP.ClassName
FROM Vendor V
JOIN Meal M ON V.VendorID = M.VendorID
JOIN ClassPolicy CP ON M.ClassAvailable = CP.ClassID
ORDER BY V.VendorName, M.Name;
```



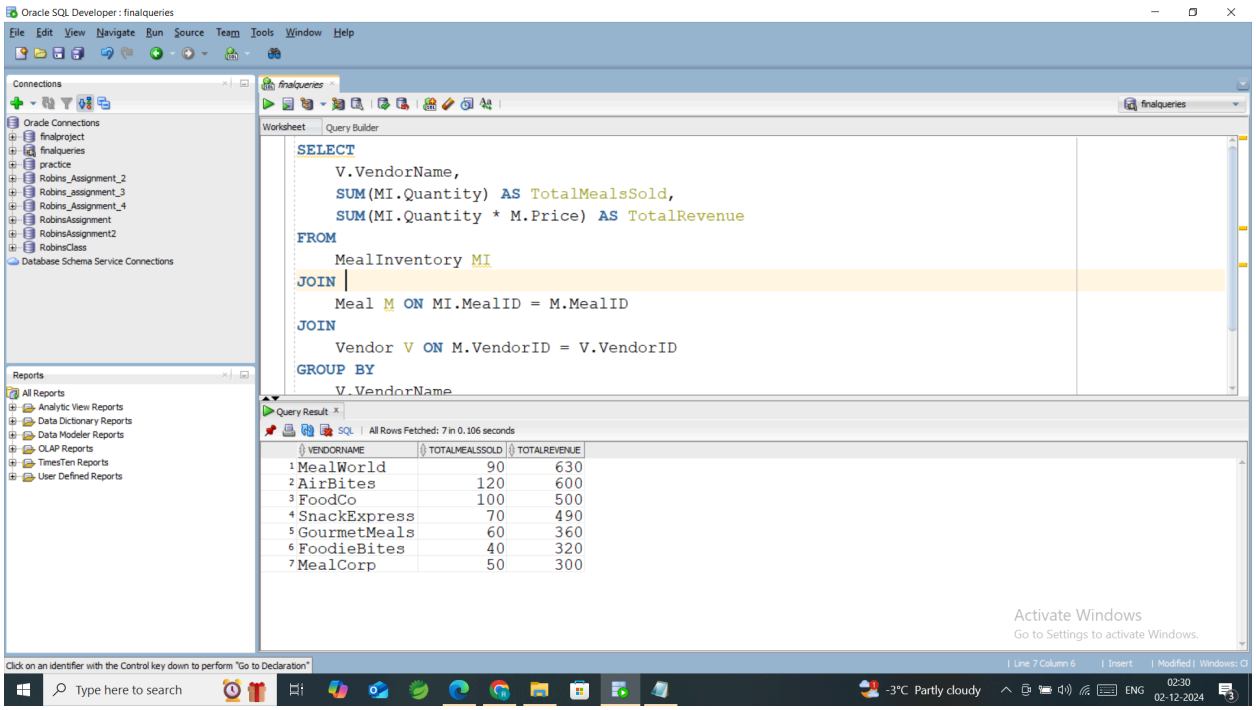
10)Find the highest baggage fee paid by passengers based on weight limits:

```
SELECT P.Name, BF.FeeAmount, B.Weight
FROM Baggage B
JOIN Passenger P ON B.PassengerID = P.PassengerID
JOIN BaggageFees BF ON B.ClassID = BF.ClassID
WHERE B.Weight > BF.WeightLimit
ORDER BY BF.FeeAmount DESC;
```



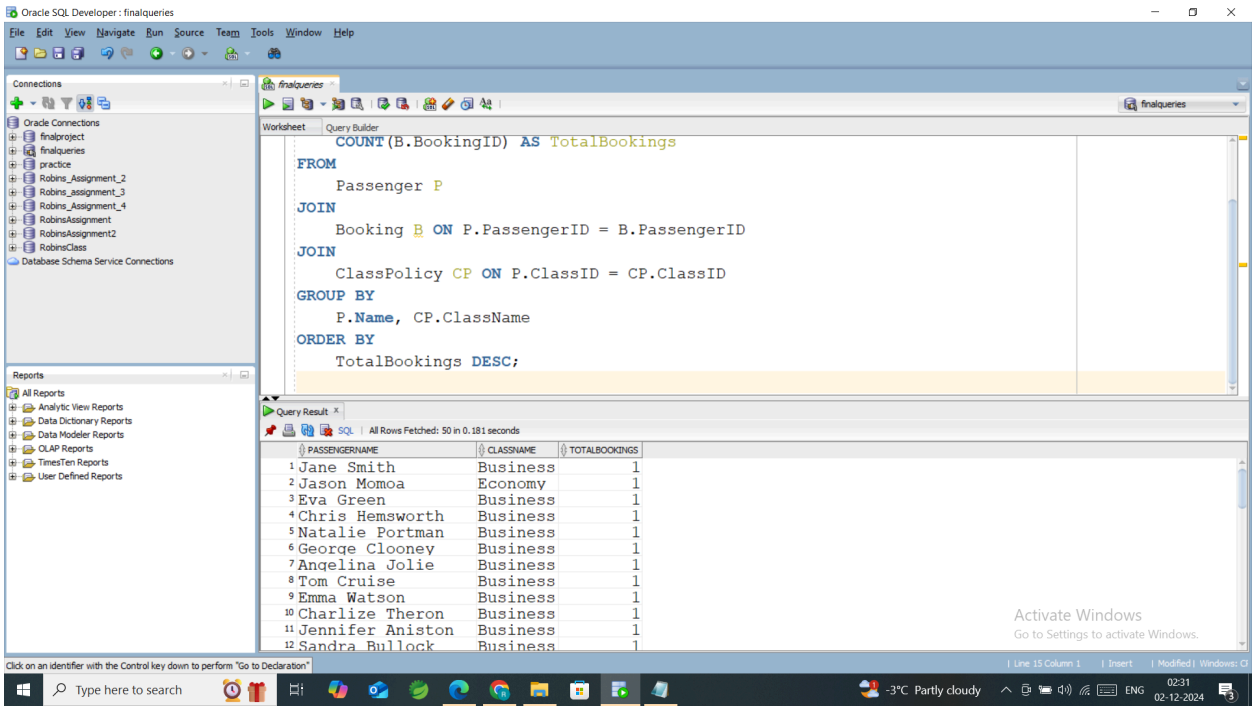
APPLICATIONS/ANALYSIS OF BCMS

1. **Top Performing Vendors.**



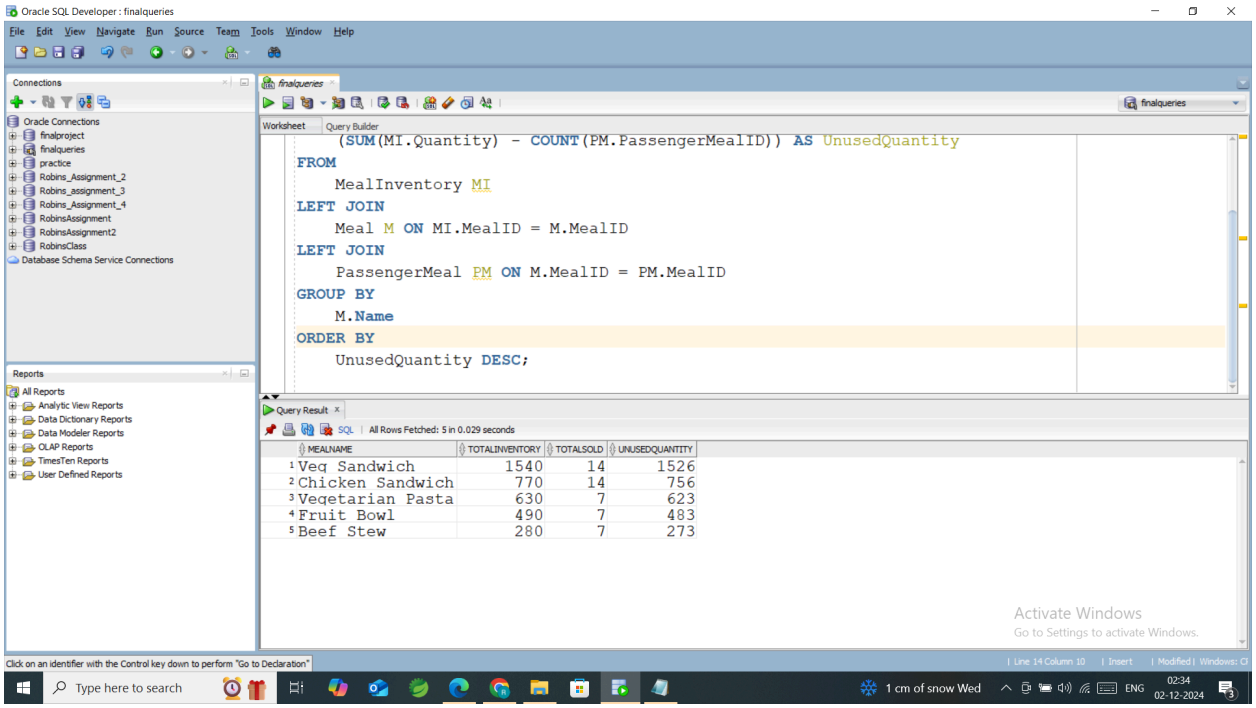
2. Frequent Flyers by Class

Identify passengers who have made the most bookings, categorized by class.



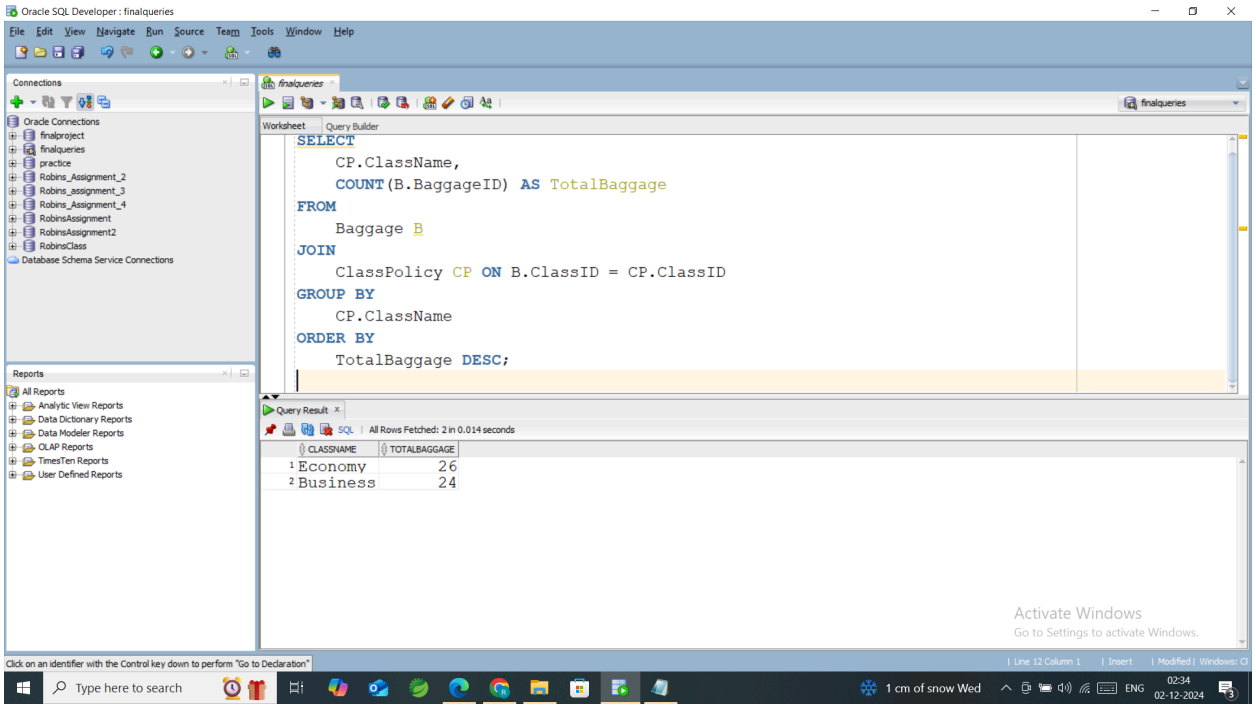
3. Unutilized Meals

Find meals that are underutilized (i.e., low quantity sold compared to available inventory).



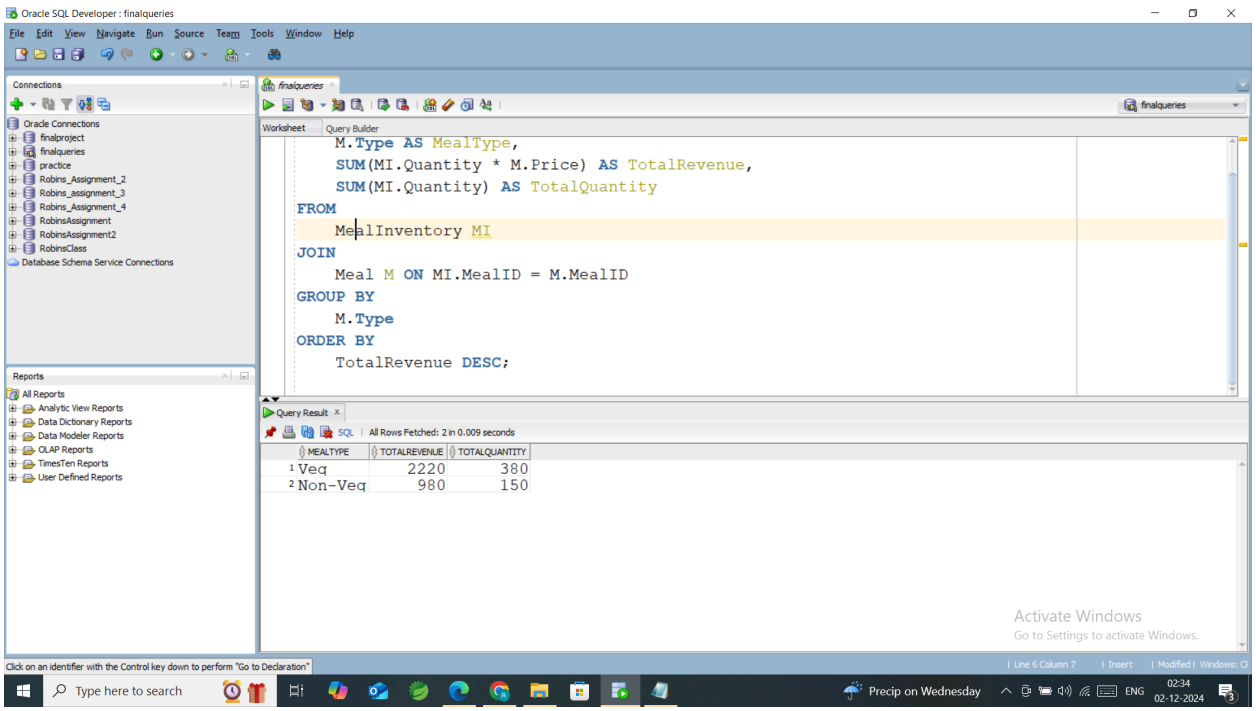
4- Popular Baggage Classes

Find which class of passengers utilizes the baggage service the most.

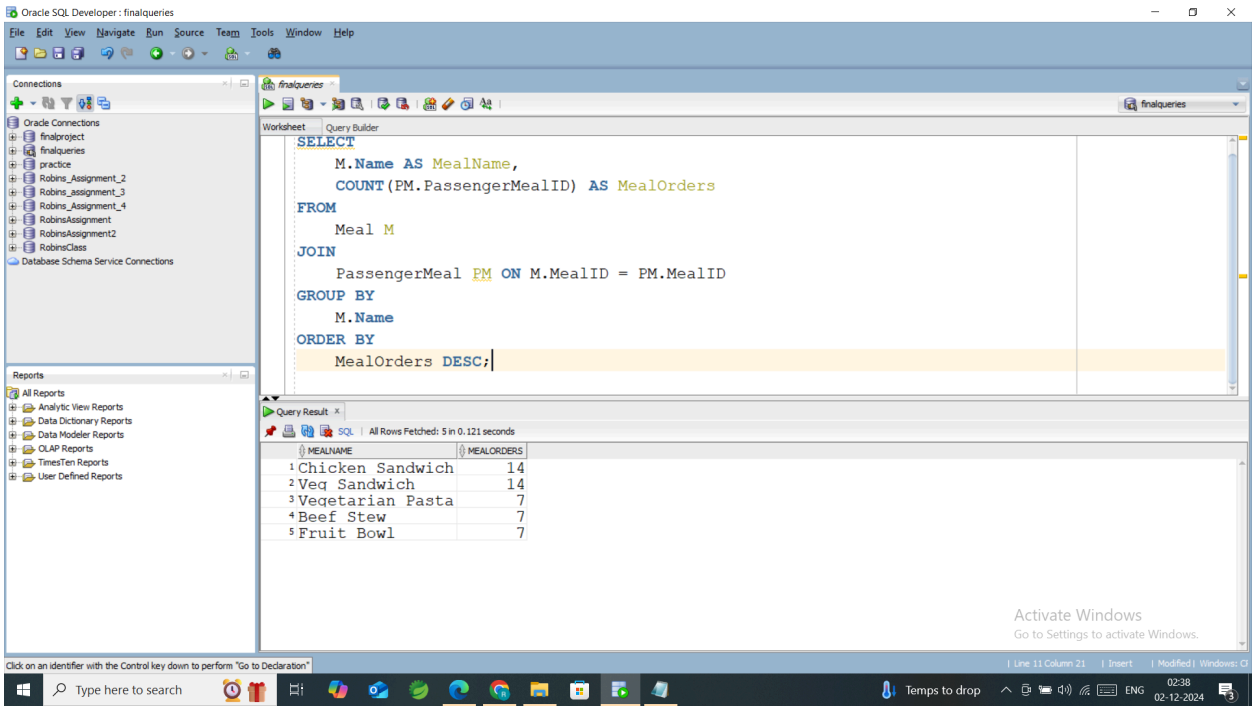


5. Profitability by Meal Type

Compare the profitability of vegetarian and non-vegetarian meals.

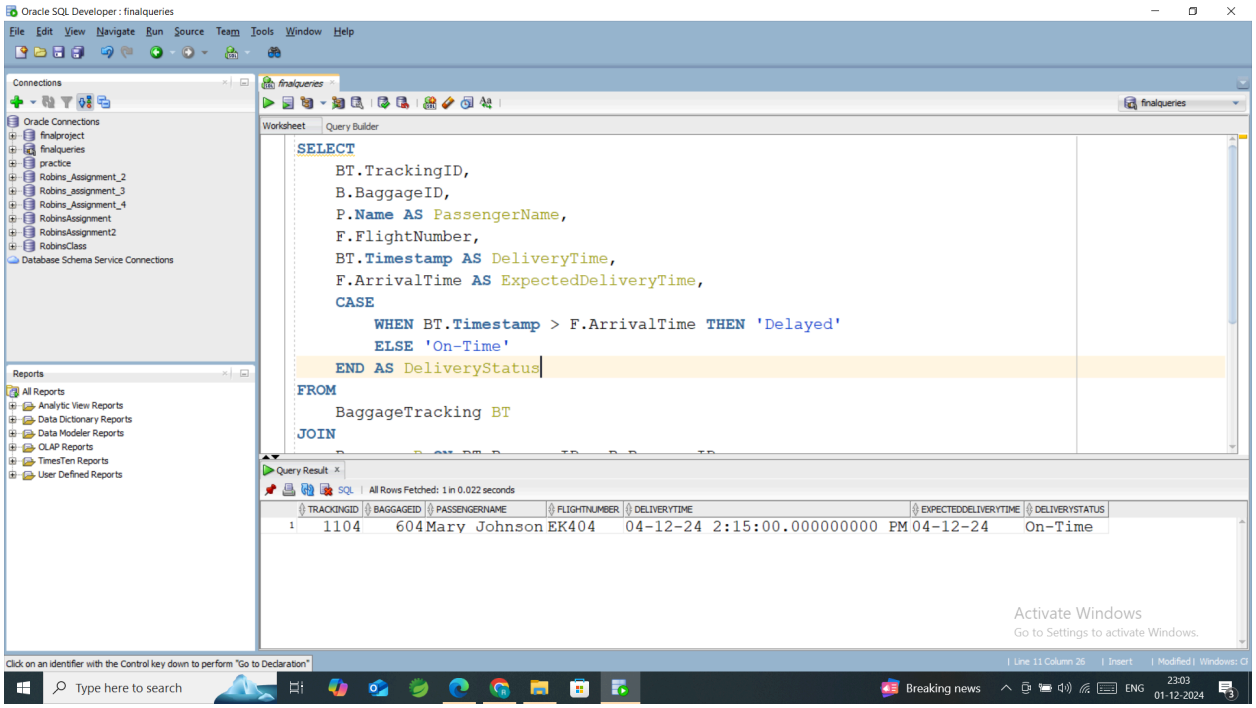


6. Identify the most popular meals based on the number of passenger meal orders.



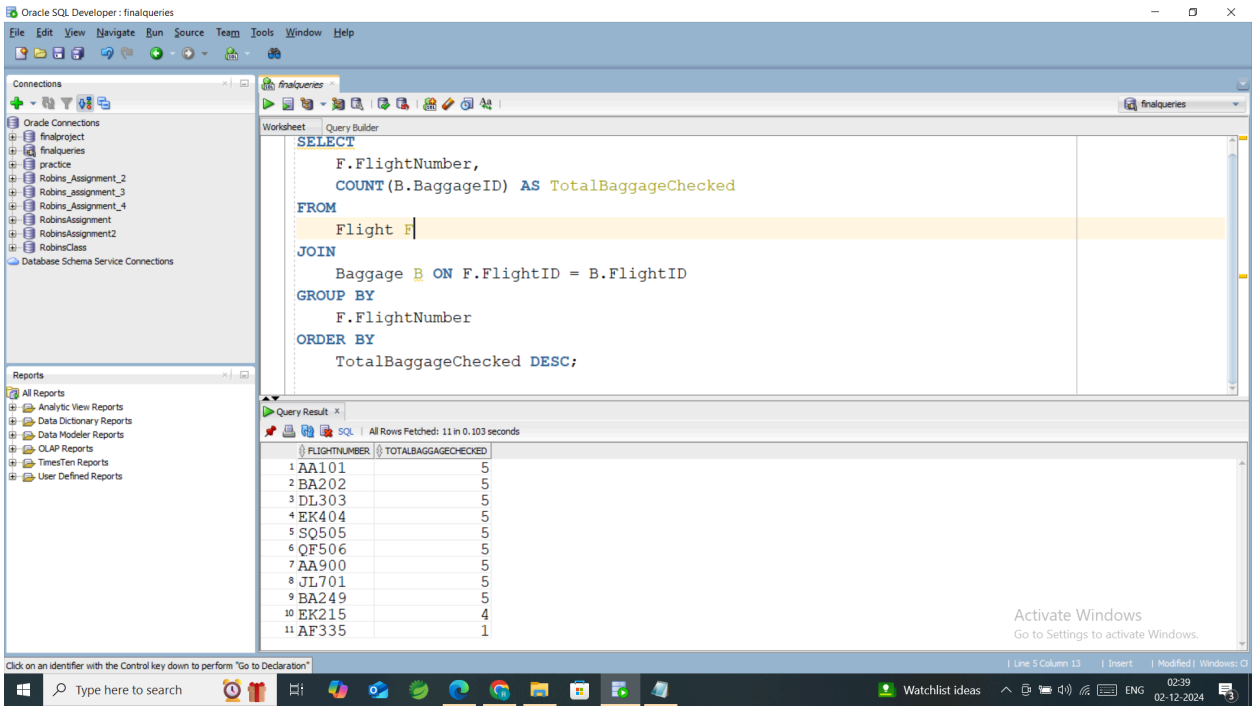
7. Delayed Baggage Tracking

Find baggage that has been delayed or not delivered within the expected time frame.



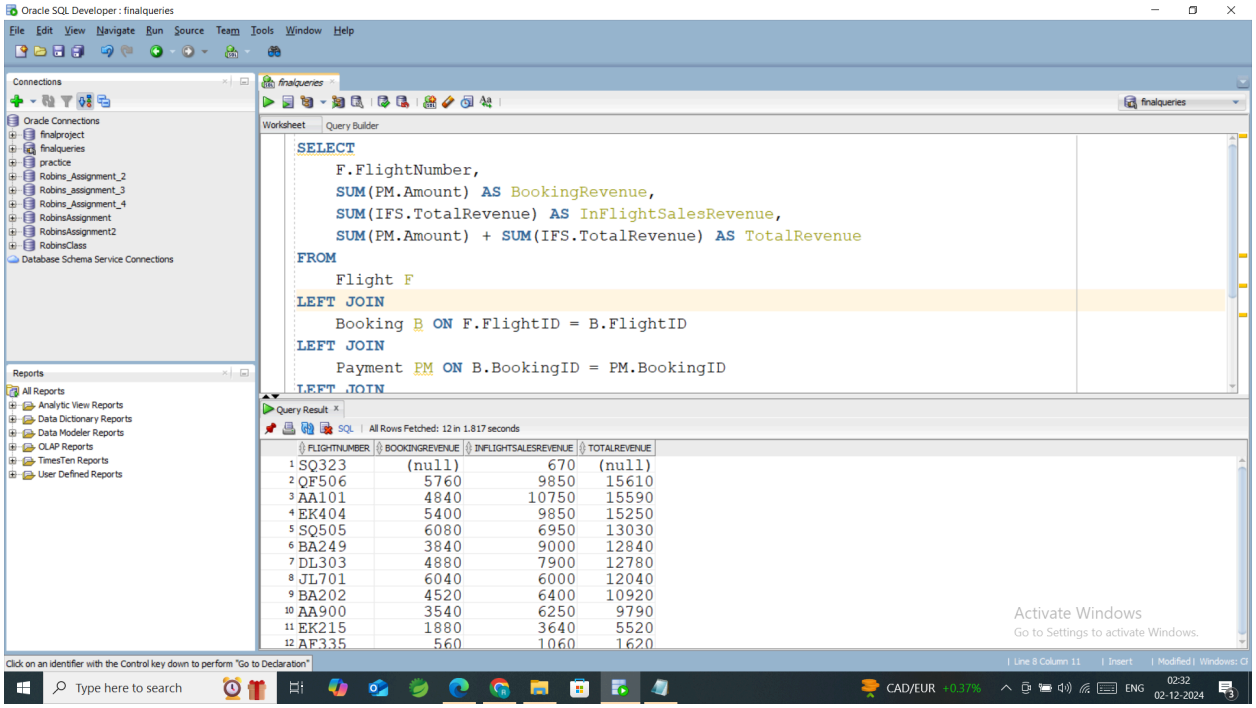
8 Baggage Volume by Flight

Identify the number of baggage items checked in on each flight.



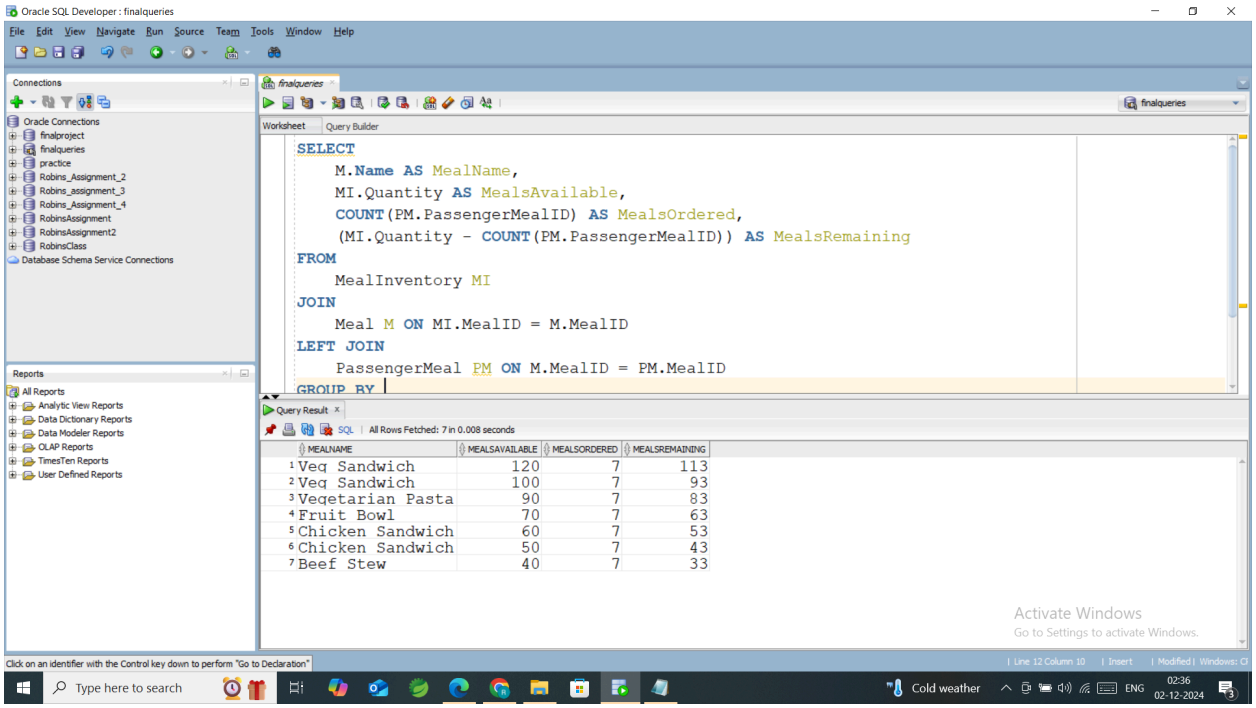
9-Revenue by flights

Calculate the total revenue generated by each flight (including booking payments and in-flight sales).



10- Catering Inventory Status

Check the current status of catering inventory, including the number of meals available versus meals ordered.



LEARNING OUTCOMES

1. Understanding System Requirements:

The project emphasizes the ability to effectively gather, analyze, and define detailed requirements for the Baggage and Catering Management System (BCMS). By focusing on real-world challenges such as baggage tracking and catering services, the system was designed to meet both the immediate needs of airport operations and future scalability. It involves a thorough exploration of user needs, performance expectations, and potential system integrations to ensure the solution is adaptable, practical, and aligned with operational goals.

2. Database Architecture Design:

Through the project, the importance of designing a comprehensive and efficient database architecture is demonstrated. Using Entity-Relationship (ER) diagrams, a clear structure for data entities like Baggage, Passengers, and Meals was developed. The database design also considers data flow, storage optimization, and

retrieval methods, ensuring smooth and effective operations for managing large datasets and real-time information.

3. Normalization and Data Integrity:

In the project, normalization was applied to organize the database effectively and reduce redundancy. The database was designed to meet First Normal Form (1NF) by ensuring all data entries are atomic and by eliminating repeating groups. This provided a structured layout for the data without unnecessary duplication.

4. Optimized Data Relationships:

The project illustrates how to design and manage complex data relationships in the system, such as linking Baggage, Meals, Flight Schedules, and Passengers. By utilizing appropriate foreign keys and relationships, the system is optimized for accurate and streamlined data retrieval. This approach helps ensure that all related data is seamlessly integrated, minimizing errors and delays in retrieving information about baggage status, meal preferences, or customer details.

5. Clear Definition of System Processes:

The system processes, such as baggage handling and meal orders, were clearly defined with specific business rules. These rules govern how data flows between the entities and ensure operational consistency. By detailing these processes, the project guarantees that the system remains efficient, transparent, and capable of handling all tasks related to baggage management and catering services without redundancy or confusion.

6. Informed Design Decision Making:

The project reflects a strategic approach to database design decisions. Key design choices, including the use of foreign key relationships, the creation of indexed tables for faster data retrieval, and the implementation of constraint checks to enforce data accuracy, were made to ensure the system's robustness and scalability. These decisions ensure that the system performs well under heavy data loads while remaining flexible for future upgrades.

7. Practical Data Modeling:

The use of ER diagrams was central to translating complex business processes into structured, understandable visual representations. These diagrams provide a clear map of the relationships between data entities, ensuring that both technical and non-technical stakeholders can easily understand how the system operates. By illustrating data relationships through effective modeling, the project enhances communication and understanding across all project phases.

8. Ensuring System Efficiency:

The project focuses on designing a highly efficient system that can handle large volumes of real-time data while maintaining optimal performance. Whether tracking baggage through the airport or managing meal preferences, the database was designed to efficiently store, retrieve, and process data in a way that ensures minimal delays and errors.

CONCLUSION

The Baggage and Catering Management System (BCMS) effectively addresses airports' key operational challenges in managing baggage and catering services. Outdated systems often lead to inefficiencies and errors in tracking baggage, managing meal inventories, and coordinating operations across various airport functions. By developing a modern system that integrates real-time data tracking and dynamic resource allocation, BCMS optimizes airport operations, enhances data accuracy, and improves overall efficiency.

Through careful requirement gathering, database design, and normalization, we created a system that is both scalable and reliable. The system ensures seamless integration across different airport services, such as baggage handling and meal management, while providing valuable insights to improve decision-making. By focusing on key areas like data integrity, user-friendly design, and scalability, BCMS not only improves operational workflows but also enhances the passenger experience.

In conclusion, the BCMS offers a robust solution to the pressing challenges in airport operations, paving the way for more efficient and interconnected airport services. With the foundation laid for future advancements like real-time data updates and predictive analytics, the system ensures long-term growth and adaptability in the evolving landscape of airport management**.**

REFERENCES

1. Codd, E. F. (1970). "A Relational Model of Data for Large Shared Data Banks." Communications of the ACM. This seminal paper laid the foundation for relational database design, which was a cornerstone of your project in implementing a robust relational database system.
2. Elmasri, R., & Navathe, S. B. (2020). Fundamentals of Database Systems. Pearson Education. This book provided critical insights into database design, normalization, and entity-relationship modeling, directly aligning with the design and normalization processes in your report.
3. Oracle Corporation. (2024). Oracle SQL Documentation. Retrieved from https://www.oracle.com/database/technologies/. This resource was instrumental in guiding the use of SQL DDL, DML, and complex queries as outlined in your project.
4. Mullins, C. S. (2018). Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design. Addison-Wesley Professional. This book provided practical approaches to designing a well-structured database, which directly supports the goals and methodologies described in your project.
5. Airline Operations Management Systems. (2023). "Integrating Airport Baggage and Catering Systems: A Case Study." Retrieved from https://www.airlinemanagementsystems.com/. This case study addressed real-world challenges in airport operations, directly correlating with the problem statement and business problems addressed in your project.

APPENDIX

- **Team Charter**(Team Charter – Team1.pdf)
- **Database Topic and Project Objective** (Baggage and Catering Management System.pdf)
- **Design and Initial ERD** (Initial ERD.pdf)
- **DDL Scripts DML Script**(DDL AND DML SCRIPT.PDF)
- **Complex Queries and Joins** (Complex Queries and Joins.sql)