345537708 Samuel Tapiro - 133467 Lior Tordjman - 346012065 Daniel Elbaz

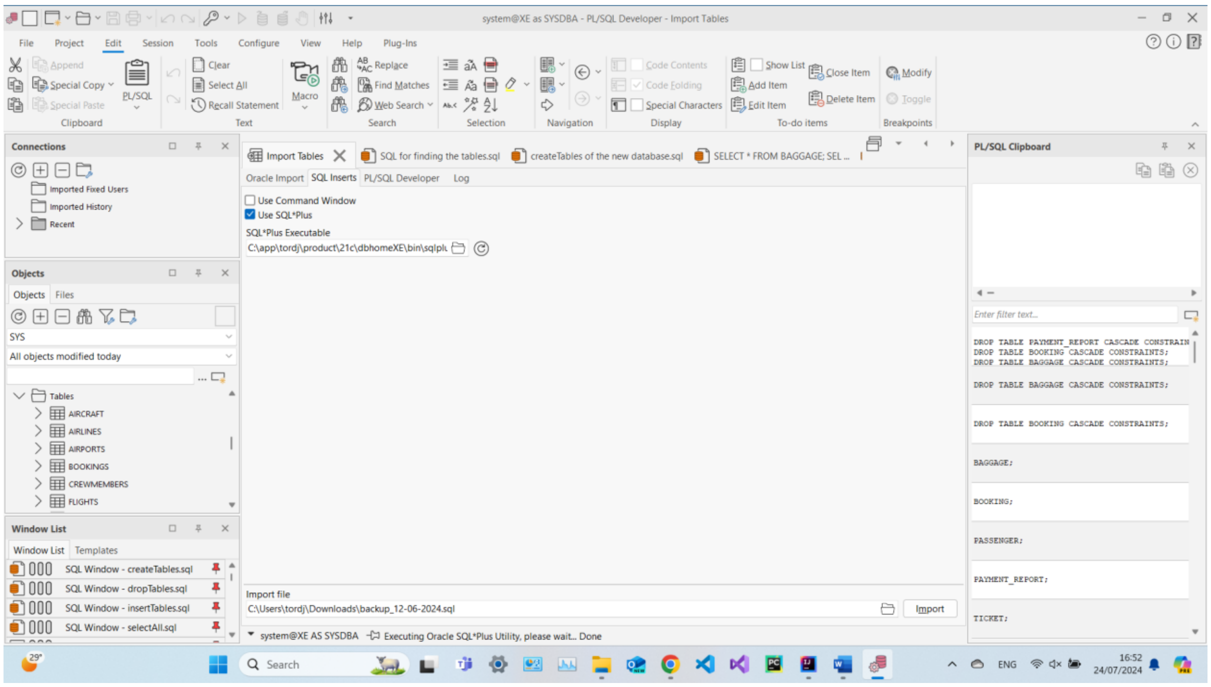
Step D

1. Introduction:

This report documents the final phase of the Database Integration Project, where we integrate two distinct database systems into one cohesive database. The objective of this phase is to combine our existing database with the database received from another project, ensuring seamless integration of the data and structures of both systems. This involves reverse engineering the received database to understand its structure, designing an integrated ERD, and implementing necessary changes to achieve the integration.

2. Backup processing:

We imported the tables from the backup of another database of an airport that we'll integrate on our own duty-free database.


system@XE as SYSDBA - PL/SQL Developer - Import Tables

Import Tables | SQL for finding the tables.sql | createTables of the new database.sql | SELECT * FROM BAGGAGE; SEL ...

Oracle Import | SQL Inserts | PL/SQL Developer | Log

[ ] Use Command Window
[✓] Use SQL*Plus

SQL*Plus Executable: C:\app\tordj\product\21c\dbhomeXE\bin\sqlplu

Import file: C:\Users\tordj\Downloads\backup_12-06-2024.sql

system@XE AS SYSDBA — Executing Oracle SQL*Plus Utility, please wait... Done

Tables: AIRCRAFT, AIRLINES, AIRPORTS, BOOKINGS, CREWMEMBERS, FLIGHTS

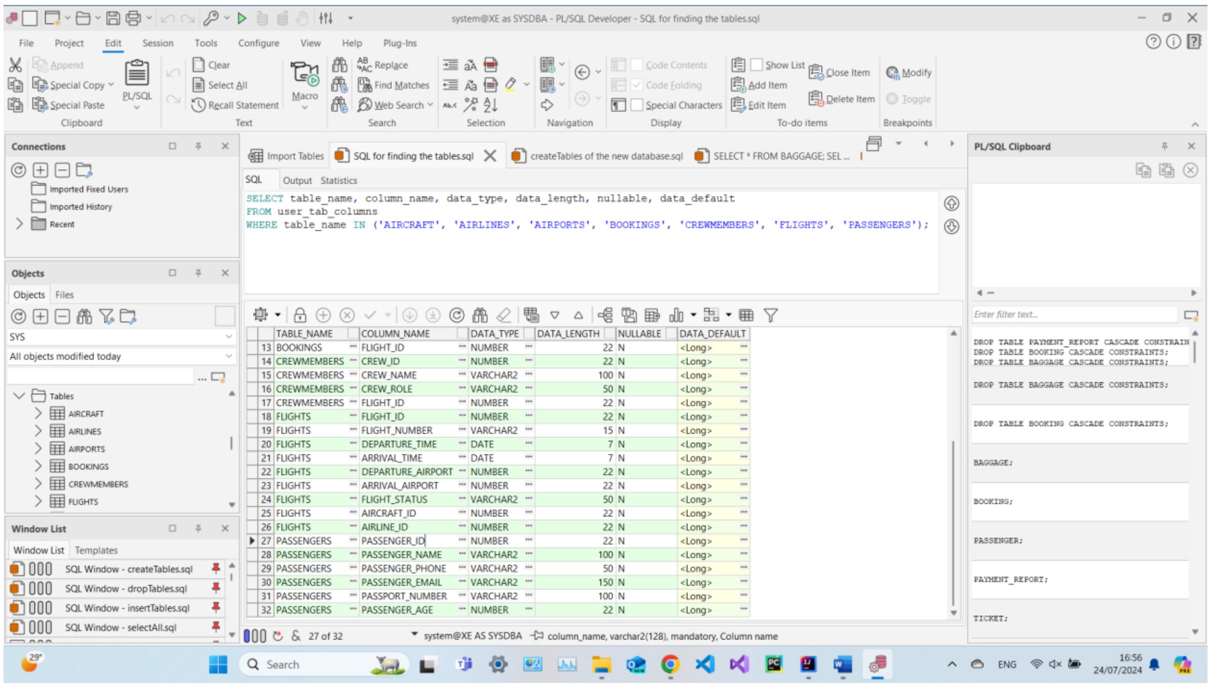
Window List: SQL Window - createTables.sql; SQL Window - dropTables.sql; SQL Window - insertTables.sql; SQL Window - selectAll.sql

PL/SQL Clipboard:

```
DROP TABLE PAYMENT_REPORT CASCADE CONSTRAIN
DROP TABLE BOOKING CASCADE CONSTRAINTS;
DROP TABLE BAGGAGE CASCADE CONSTRAINTS;

DROP TABLE BAGGAGE CASCADE CONSTRAINTS;

DROP TABLE BOOKING CASCADE CONSTRAINTS;

BAGGAGE;

BOOKING;

PASSENGER;

PAYMENT_REPORT;

TICKET;
```

Using the backup file, we could retrieve the create statements so we can start building our DSD of this database.

345537708 Samuel Tapiro - 133467 Lior Tordjman - 346012065 Daniel Elbaz



```sql
SELECT table_name, column_name, data_type, data_length, nullable, data_default
FROM user_tab_columns
WHERE table_name IN ('AIRCRAFT', 'AIRLINES', 'AIRPORTS', 'BOOKINGS', 'CREWMEMBERS', 'FLIGHTS', 'PASSENGERS');
```

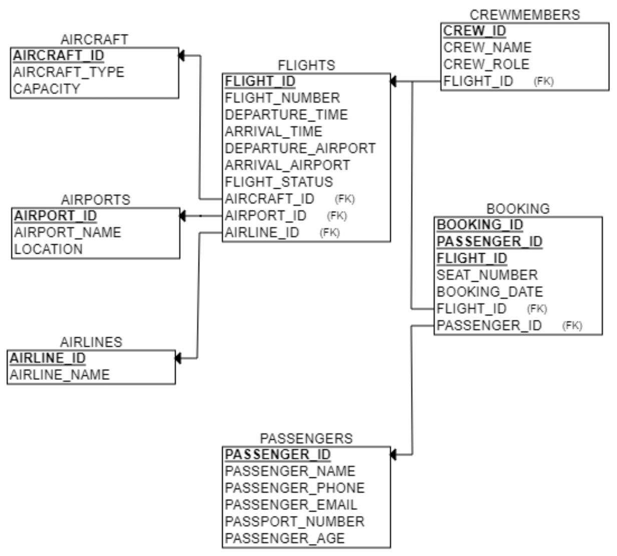
	TABLE_NAME	COLUMN_NAME	DATA_TYPE	DATA_LENGTH	NULLABLE	DATA_DEFAULT
13	BOOKINGS	FLIGHT_ID	NUMBER	22	N	<Long>
14	CREWMEMBERS	CREW_ID	NUMBER	22	N	<Long>
15	CREWMEMBERS	CREW_NAME	VARCHAR2	100	N	<Long>
16	CREWMEMBERS	CREW_ROLE	VARCHAR2	50	N	<Long>
17	CREWMEMBERS	FLIGHT_ID	NUMBER	22	N	<Long>
18	FLIGHTS	FLIGHT_ID	NUMBER	22	N	<Long>
19	FLIGHTS	FLIGHT_NUMBER	VARCHAR2	15	N	<Long>
20	FLIGHTS	DEPARTURE_TIME	DATE	7	N	<Long>
21	FLIGHTS	ARRIVAL_TIME	DATE	7	N	<Long>
22	FLIGHTS	DEPARTURE_AIRPORT	NUMBER	22	N	<Long>
23	FLIGHTS	ARRIVAL_AIRPORT	NUMBER	22	N	<Long>
24	FLIGHTS	FLIGHT_STATUS	VARCHAR2	50	N	<Long>
25	FLIGHTS	AIRCRAFT_ID	NUMBER	22	N	<Long>
26	FLIGHTS	AIRLINE_ID	NUMBER	22	N	<Long>
27	PASSENGERS	PASSENGER_ID	NUMBER	22	N	<Long>
28	PASSENGERS	PASSENGER_NAME	VARCHAR2	100	N	<Long>
29	PASSENGERS	PASSENGER_PHONE	VARCHAR2	50	N	<Long>
30	PASSENGERS	PASSENGER_EMAIL	VARCHAR2	150	N	<Long>
31	PASSENGERS	PASSPORT_NUMBER	VARCHAR2	100	N	<Long>
32	PASSENGERS	PASSENGER_AGE	NUMBER	22	N	<Long>

```
27 of 32    system@XE AS SYSDBA    column_name, varchar2(128), mandatory, Column name
```

```
DROP TABLE PAYMENT_REPORT CASCADE CONSTRAIN
DROP TABLE BOOKING CASCADE CONSTRAINTS;
DROP TABLE BAGGAGE CASCADE CONSTRAINTS;

DROP TABLE BAGGAGE CASCADE CONSTRAINTS;

DROP TABLE BOOKING CASCADE CONSTRAINTS;

BAGGAGE;

BOOKING;

PASSENGER;

PAYMENT_REPORT;

TICKET;
```

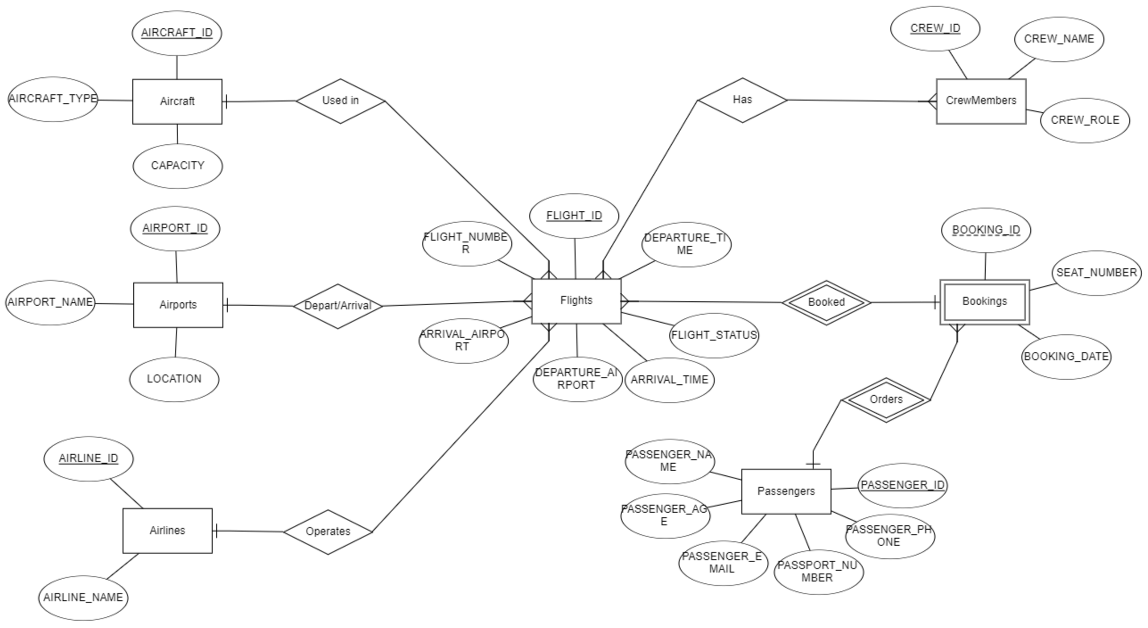
system@XE as SYSDBA - PL/SQL Developer - SQL for finding the tables.sql
File
Project
Edit
Session
Tools
Configure
View
Help
Plug-Ins
Import Tables
SQL for finding the tables.sql
createTables of the new database.sql
SELECT * FROM BAGGAGE; SEL ...
Connections
Objects
Window List
PL/SQL Clipboard
SQL Window - createTables.sql
SQL Window - dropTables.sql
SQL Window - insertTables.sql
SQL Window - selectAll.sql
16:56
24/07/2024

```sql
CREATE TABLE Aircraft
(
  AIRCRAFT_ID INT NOT NULL,
  AIRCRAFT_TYPE VARCHAR2(100) NOT NULL,
  CAPACITY INT NOT NULL,
  PRIMARY KEY (AIRCRAFT_ID)
);

CREATE TABLE Airlines
(
  AIRLINE_ID INT NOT NULL,
  AIRLINE_NAME VARCHAR2(100) NOT NULL,
  PRIMARY KEY (AIRLINE_ID)
);

CREATE TABLE Airports
(
  AIRPORT_ID INT NOT NULL,
  AIRPORT_NAME VARCHAR2(50) NOT NULL,
  LOCATION VARCHAR2(100) NOT NULL,
  PRIMARY KEY (AIRPORT_ID)
);

CREATE TABLE Passengers
(
  PASSENGER_ID INT NOT NULL,
  PASSENGER_NAME VARCHAR2(100) NOT NULL,
  PASSENGER_PHONE VARCHAR2(50) NOT NULL,
  PASSENGER_EMAIL VARCHAR2(150) NOT NULL,
  PASSPORT_NUMBER VARCHAR2(100) NOT NULL,
  PASSENGER_AGE INT NOT NULL,
  PRIMARY KEY (PASSENGER_ID)
);

CREATE TABLE Flights
(
  FLIGHT_ID INT NOT NULL,
  FLIGHT_NUMBER VARCHAR2(15) NOT NULL,
  DEPARTURE_TIME DATE NOT NULL,
  ARRIVAL_TIME DATE NOT NULL,
  DEPARTURE_AIRPORT INT NOT NULL,
  ARRIVAL_AIRPORT INT NOT NULL,
  FLIGHT_STATUS VARCHAR2(50) NOT NULL,
  AIRCRAFT_ID INT NOT NULL,
  AIRLINE_ID INT NOT NULL,
  PRIMARY KEY (FLIGHT_ID),
  FOREIGN KEY (AIRCRAFT_ID) REFERENCES Aircraft(AIRCRAFT_ID),
  FOREIGN KEY (AIRLINE_ID) REFERENCES Airlines(AIRLINE_ID),
  FOREIGN KEY (DEPARTURE_AIRPORT) REFERENCES Airports(AIRPORT_ID),
  FOREIGN KEY (ARRIVAL_AIRPORT) REFERENCES Airports(AIRPORT_ID)
);

CREATE TABLE Bookings
(
  BOOKING_ID INT NOT NULL,
  SEAT_NUMBER VARCHAR2(10) NOT NULL,
  BOOKING_DATE DATE NOT NULL,
  PASSENGER_ID INT NOT NULL,
  FLIGHT_ID INT NOT NULL,
  PRIMARY KEY (BOOKING_ID, PASSENGER_ID, FLIGHT_ID),
  FOREIGN KEY (PASSENGER_ID) REFERENCES Passengers(PASSENGER_ID),
  FOREIGN KEY (FLIGHT_ID) REFERENCES Flights(FLIGHT_ID)
);

CREATE TABLE CrewMembers
(
  CREW_ID INT NOT NULL,
  CREW_NAME VARCHAR2(100) NOT NULL,
  CREW_ROLE VARCHAR2(50) NOT NULL,
  FLIGHT_ID INT NOT NULL,
  PRIMARY KEY (CREW_ID),
  FOREIGN KEY (FLIGHT_ID) REFERENCES Flights(FLIGHT_ID)
);
```



3. <u>Reverse engineering:</u>

Using reverse engineering on the DSD, we could retrieve the ERD of the received airport database.



4. <u>Integrating database into new ERD:</u>

During the integration process, we recognized that the Passengers entity represents individuals in the airport. Consequently, this entity includes passengers who may also be customers of the duty-free shops within the airport. This understanding allows us to track activities and transactions of passengers within the airport environment comprehensively.

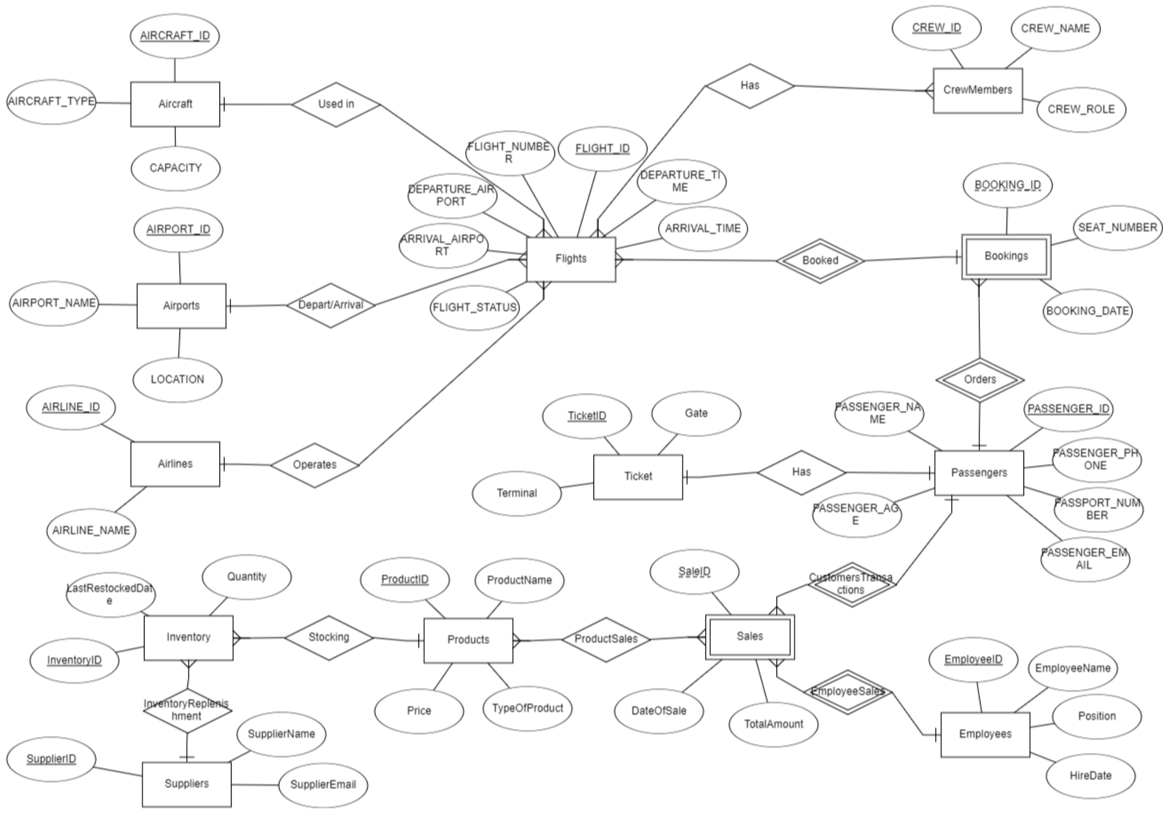
For example, a passenger identified by a specific PassengerID might purchase an electronic device from a duty-free store, thereby acting as a customer. This integration facilitates a unified view of passenger activities, from booking flights to making purchases, thereby enhancing the realism and utility of the database.

Conflicts arose when there were differences in attribute definitions or when entities had similar purposes but different names. For example, our database had an entity named Customers, while the received database had an entity named Passengers. After reviewing the attributes, we decided to merge them into a single Passengers entity, retaining all unique attributes.

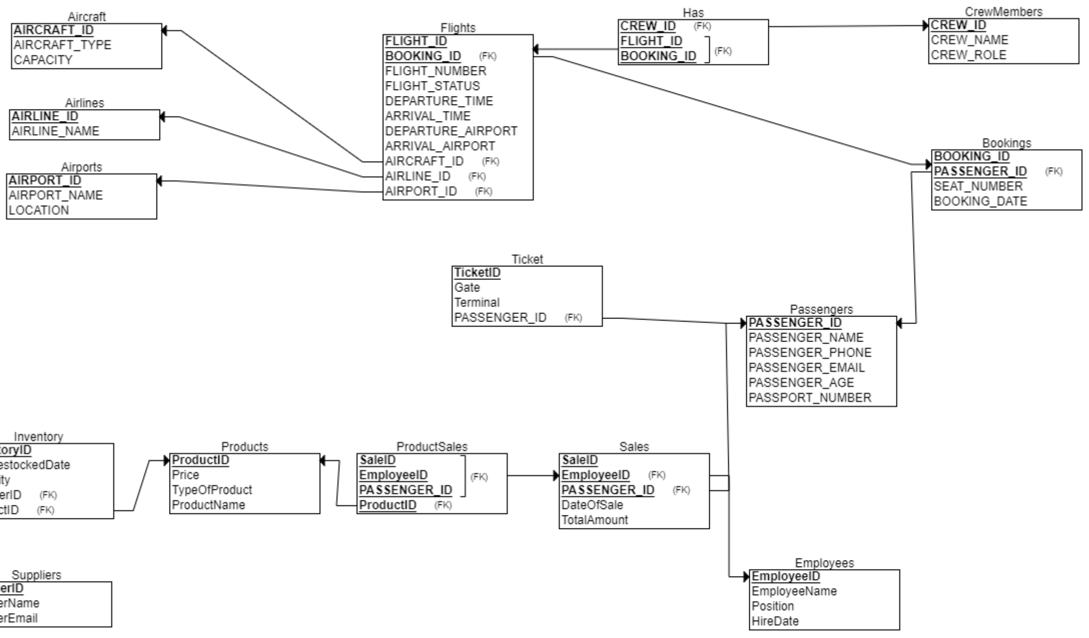
To better represent the reality of the integrated system, we added a new entity, Ticket. This entity captures additional details related to flight tickets, such as Gate and Terminal, which were not covered by existing entities.

345537708 Samuel Tapiro - 133467 Lior Tordjman - 346012065 Daniel Elbaz

Screenshot of the new integrated ERD



Screenshot of the new integrated DSD



5. **Integrate.sql:**

We added the "Ticket" entity to adapt more to reality.

```
CREATE TABLE TICKET
(
  TICKET_ID INT NOT NULL,
  GATE VARCHAR2(10) NOT NULL,
  TERMINAL INT NOT NULL,
  PRIMARY KEY (TICKET_ID)
);
```

6. **Views.sql:**

The Views.sql file is designed to create views that illustrate the interaction between the duty-free database and the airport database, showcasing the relationship between passengers, their purchases at duty-free shops, and their flight details. This helps provide a comprehensive view of passenger activities within the airport environment.

View 1: PassengerPurchasesView1

The PassengerPurchasesView1 view combines data from the Passengers, Sales, Products, and Employees tables to provide detailed information on purchases made by passengers at the duty-free shops.

This view allows us to analyze passenger spending behavior and understand the role of employees in facilitating these sales.

```
-- View 1: PassengerPurchases
-- This view shows the purchases made by passengers at the duty-free shops.
CREATE VIEW PassengerPurchasesView1 AS
SELECT
    p.PASSENGER_ID,
    p.PASSENGER_NAME,
    p.PASSENGER_EMAIL,
    s.SALEID,
    s.DATEOFSALE,
    s.TOTALAMOUNT,
    pr.PRODUCTNAME,
    pr.PRICE,
    e.EMPLOYEENAME
FROM
    Passengers p
JOIN
    Sales s ON p.PASSENGER_ID = s.CustomerID
JOIN
    Products pr ON s.SaleID = pr.ProductID
JOIN
    Employees e ON s.EmployeeID = e.EmployeeID;
```

View 2: FlightPassengerDetailsView2

The FlightPassengerDetailsView2 view combines data from the Flights, Bookings, and Passengers tables to provide detailed information on flight details and the passengers who have booked those flights. This view captures the following details:

This view allows us to analyze passenger travel patterns and manage flight bookings effectively.

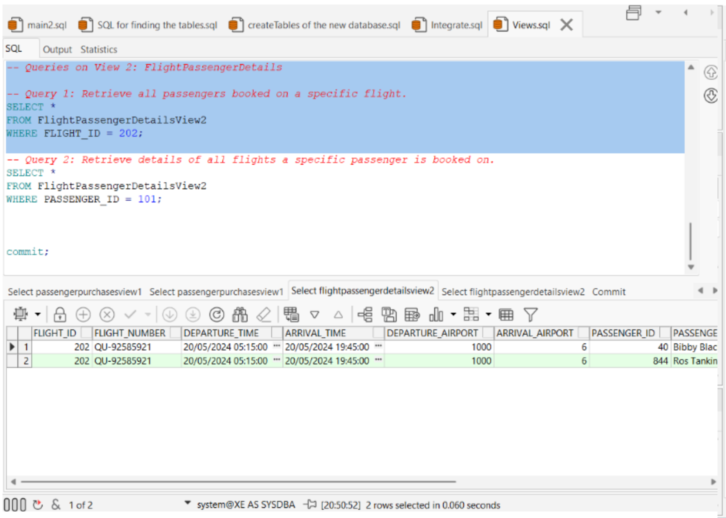
```
-- View 2: FlightPassengerDetails
-- This view shows flight details along with the passengers who have booked
the flights.
CREATE VIEW FlightPassengerDetailsView2 AS
SELECT
    f.FLIGHT_ID,
    f.FLIGHT_NUMBER,
    f.DEPARTURE_TIME,
    f.ARRIVAL_TIME,
    f.DEPARTURE_AIRPORT,
    f.ARRIVAL_AIRPORT,
    p.PASSENGER_ID,
    p.PASSENGER_NAME,
    p.PASSENGER_EMAIL,
    b.BOOKING_ID,
    b.SEAT_NUMBER
FROM
    Flights f
JOIN
    Bookings b ON f.FLIGHT_ID = b.FLIGHT_ID
JOIN
    Passengers p ON b.PASSENGER_ID = p.PASSENGER_ID;
```

Queries:

```
-- Queries on Views

-- Queries on View 1: PassengerPurchases

-- Query 1: Retrieve all purchases made by a specific passenger.
SELECT *
FROM PassengerPurchasesView1
WHERE PASSENGER_ID = 103;

-- Query 2: Retrieve total sales amount made by all passengers.
SELECT PASSENGER_ID, PASSENGER_NAME, SUM(TOTALAMOUNT) AS TOTAL_SPENT
FROM PassengerPurchasesView1
GROUP BY PASSENGER_ID, PASSENGER_NAME;

-- Queries on View 2: FlightPassengerDetails

-- Query 1: Retrieve all passengers booked on a specific flight.
SELECT *
FROM FlightPassengerDetailsView2
WHERE FLIGHT_ID = 202;

-- Query 2: Retrieve details of all flights a specific passenger is booked
on.
```

345537708 Samuel Tapiro - 133467 Lior Tordjman - 346012065 Daniel Elbaz

```sql
SELECT *
FROM FlightPassengerDetailsView2
WHERE PASSENGER_ID = 101;



commit;
```



main2.sql | SQL for finding the tables.sql | createTables of the new database.sql | Integrate.sql | Views.sql

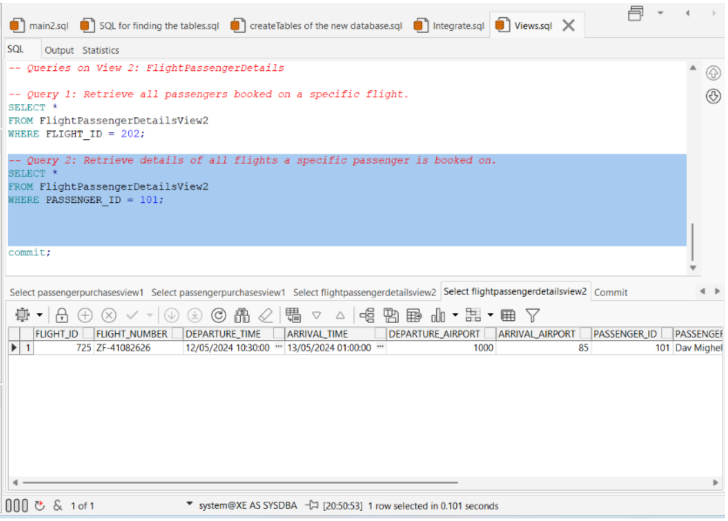
SQL | Output | Statistics

```sql
-- Queries on View 2: FlightPassengerDetails

-- Query 1: Retrieve all passengers booked on a specific flight.
SELECT *
FROM FlightPassengerDetailsView2
WHERE FLIGHT_ID = 202;

-- Query 2: Retrieve details of all flights a specific passenger is booked on.
SELECT *
FROM FlightPassengerDetailsView2
WHERE PASSENGER_ID = 101;


commit;
```

Select passengerpurchasesview1 | Select passengerpurchasesview1 | Select flightpassengerdetailsview2 | Select flightpassengerdetailsview2 | Commit

	FLIGHT_ID	FLIGHT_NUMBER	DEPARTURE_TIME	ARRIVAL_TIME	DEPARTURE_AIRPORT	ARRIVAL_AIRPORT	PASSENGER_ID	PASSENGE
1	202	QU-92585921	20/05/2024 05:15:00	20/05/2024 19:45:00	1000	6	40	Bibby Blac
2	202	QU-92585921	20/05/2024 05:15:00	20/05/2024 19:45:00	1000	6	844	Ros Tankin

1 of 2 system@XE AS SYSDBA [20:50:52] 2 rows selected in 0.060 seconds



main2.sql | SQL for finding the tables.sql | createTables of the new database.sql | Integrate.sql | Views.sql

SQL | Output | Statistics

```sql
-- Queries on View 2: FlightPassengerDetails

-- Query 1: Retrieve all passengers booked on a specific flight.
SELECT *
FROM FlightPassengerDetailsView2
WHERE FLIGHT_ID = 202;

-- Query 2: Retrieve details of all flights a specific passenger is booked on.
SELECT *
FROM FlightPassengerDetailsView2
WHERE PASSENGER_ID = 101;


commit;
```

Select passengerpurchasesview1 | Select passengerpurchasesview1 | Select flightpassengerdetailsview2 | Select flightpassengerdetailsview2 | Commit

	FLIGHT_ID	FLIGHT_NUMBER	DEPARTURE_TIME	ARRIVAL_TIME	DEPARTURE_AIRPORT	ARRIVAL_AIRPORT	PASSENGER_ID	PASSENGEI
1	725	ZF-41082626	12/05/2024 10:30:00	13/05/2024 01:00:00	1000	85	101	Dav Mighel

1 of 1 system@XE AS SYSDBA [20:50:53] 1 row selected in 0.101 seconds