

DML & DQL task on Airline database

DQL

1. Display all flight leg records.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including the 'Airline' database. The central query window contains the following SQL code:

```
INSERT INTO fare (amount, flightID) VALUES
(120.00, 100),
(350.50, 101),
(180.00, 102),
(550.75, 103),
(210.00, 104);

SELECT * FROM airport;
SELECT * FROM flight;
SELECT * FROM flight_leg;
SELECT * FROM airplane_type;
SELECT * FROM airplane;
SELECT * FROM leg_instance;
SELECT * FROM seat_reservation;
SELECT * FROM fare;
```

The Results pane at the bottom displays the output of the query, showing a table with columns: legno, scheduled_dep_time, scheduled_arr_time, departairport, arriveairport, flightID, and airportID. The data is as follows:

legno	scheduled_dep_time	scheduled_arr_time	departairport	arriveairport	flightID	airportID
1	08:00:00.0000000	10:00:00.0000000	Muscat	Duba	100	NULL
2	11:00:00.0000000	13:00:00.0000000	Duba	London	101	NULL
3	09:00:00.0000000	12:00:00.0000000	Riyadh	Muscat	102	NULL
4	14:00:00.0000000	18:00:00.0000000	London	New York	103	NULL
5	16:00:00.0000000	18:30:00.0000000	Muscat	Riyadh	104	NULL

2. Display each flight leg ID, scheduled departure time, and arrival time.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including the 'Airline' database. The central query window contains the following SQL code:

```
INSERT INTO fare (amount, flightID) VALUES
(120.00, 100),
(350.50, 101),
(180.00, 102),
(550.75, 103),
(210.00, 104);

SELECT * FROM airport;
SELECT * FROM flight;
SELECT * FROM flight_leg;
SELECT * FROM airplane_type;
SELECT * FROM airplane;
SELECT * FROM leg_instance;
SELECT * FROM seat_reservation;
SELECT * FROM fare;
```

The Results pane at the bottom displays the output of the query, showing a table with columns: legno, scheduled_dep_time, and scheduled_arr_time. The data is as follows:

legno	scheduled_dep_time	scheduled_arr_time
1	08:00:00.0000000	10:00:00.0000000
2	11:00:00.0000000	13:00:00.0000000
3	09:00:00.0000000	12:00:00.0000000
4	14:00:00.0000000	18:00:00.0000000
5	16:00:00.0000000	18:30:00.0000000

3. Display each airplane's ID, type, and seat capacity.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'Airline' database selected. The right pane shows a query window with the following SQL code:

```

INSERT INTO fare (amount, flightID) VALUES
(120.00, 100),
(150.00, 101),
(180.00, 102),
(150.75, 103),
(210.00, 104);

SELECT * FROM airport;
SELECT * FROM flight;
SELECT * FROM flight_leg;
SELECT * FROM airplane_type;
SELECT * FROM airplane;
SELECT * FROM leg_instance;
SELECT * FROM seat_reservation;
SELECT * FROM fare;

SELECT legno, scheduled_dep_time, scheduled_arr_time
FROM flight_leg;

SELECT airplaneID, typename, total_no_seats
FROM airplane;
  
```

The 'Results' pane shows the output of the last query, displaying a table with 5 rows and 3 columns: airplaneID, typename, and total_no_seats.

airplaneID	typename	total_no_seats
1	A320	150
2	A330	250
3	B737	140
4	B787	290
5	A350	320

The status bar at the bottom indicates 'Query executed successfully.' and '5 rows'.

4. Display each flight leg's ID and available seats as AvailableSeats.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'Airline' database selected. The right pane shows a query window with the following SQL code:

```

SELECT * FROM airport;
SELECT * FROM flight;
SELECT * FROM flight_leg;
SELECT * FROM airplane_type;
SELECT * FROM airplane;
SELECT * FROM leg_instance;
SELECT * FROM seat_reservation;
SELECT * FROM fare;

SELECT legno, scheduled_dep_time, scheduled_arr_time
FROM flight_leg;

SELECT airplaneID, typename, total_no_seats
FROM airplane;

SELECT leg_instance_id AS flightID, number_of_available_seats AS AvailableSeats
FROM leg_instance;
  
```

The 'Results' pane shows the output of the last query, displaying a table with 5 rows and 2 columns: flightID and AvailableSeats.

flightID	AvailableSeats
1	150
2	200
3	140
4	270
5	200

The status bar at the bottom indicates 'Query executed successfully.' and '5 rows'.

5. List flight leg IDs with available seats greater than 100.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'Airline' database selected. The right pane shows a SQL query window with the following code:

```
SELECT leg_instance_id FROM leg_instance WHERE number_of_available_seats > 100
```

The 'Results' pane at the bottom shows 5 rows of data:

leg_instance_id
1
2
3
4
5

The status bar at the bottom indicates 'Query executed successfully' and '5 rows'.

6. List airplane IDs with seat capacity above 300.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'Airline' database selected. The right pane shows a SQL query window with the following code:

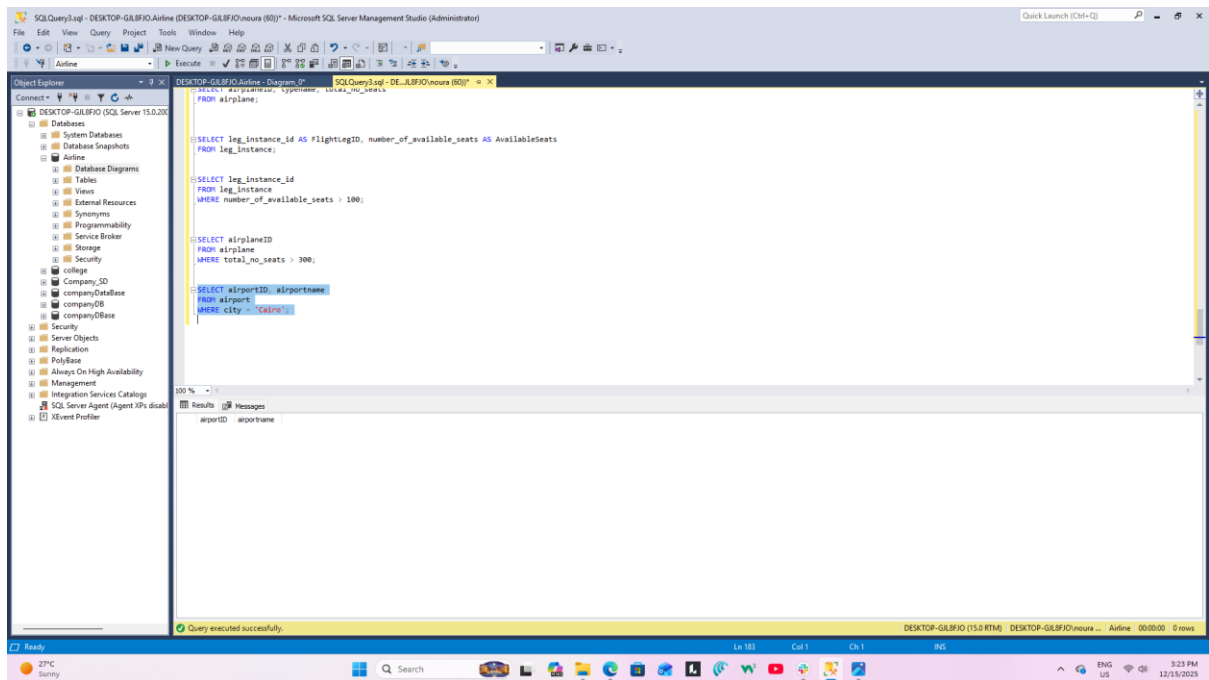
```
SELECT airplaneID FROM airplane WHERE total_no_seats > 300
```

The 'Results' pane at the bottom shows 1 row of data:

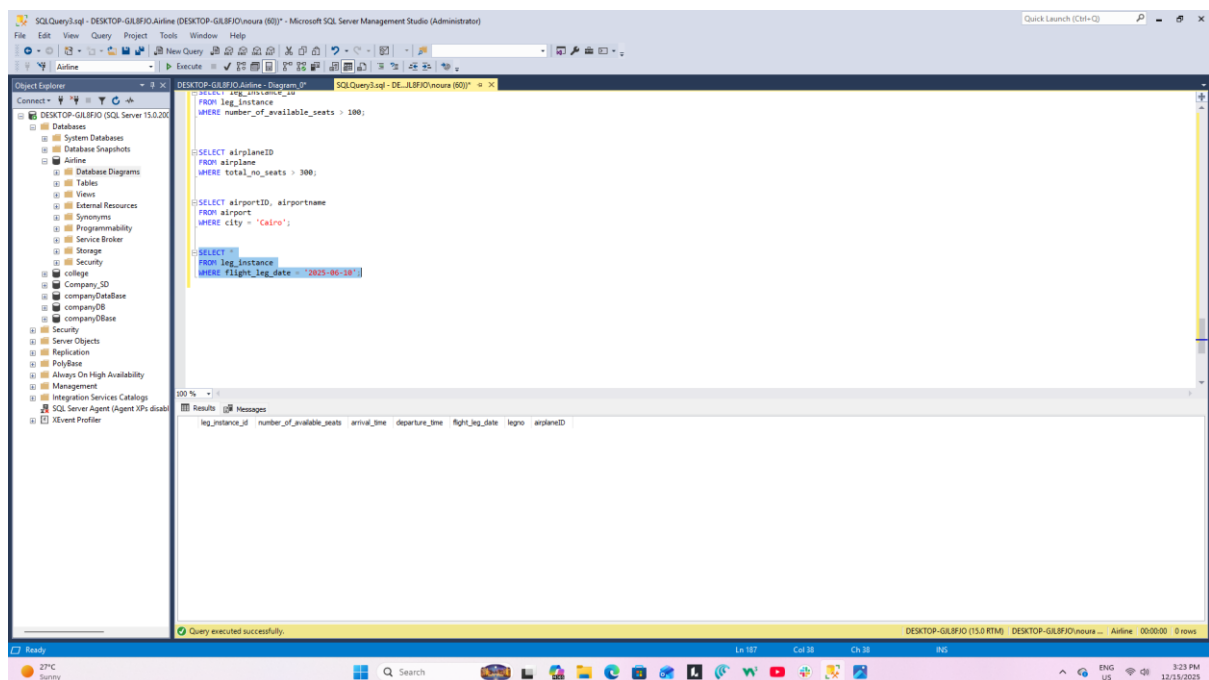
airplaneID
5

The status bar at the bottom indicates 'Query executed successfully' and '1 rows'.

7. Display airport codes and names where city = 'Cairo'.



8. Display flight legs scheduled on 2025-06-10.



9. Display flight legs ordered by departure time.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'Airline' database selected. The right pane shows a SQL query window with the following query:

```

SELECT leg_instance_id
FROM leg_instance
WHERE number_of_available_seats > 100;

SELECT airplaneID
FROM airplane
WHERE total_no_seats > 300;

SELECT airportID, airportname
FROM airport
WHERE city = 'Cairo';

SELECT *
FROM leg_instance
WHERE flight_leg_date = '2025-06-18';

SELECT *
FROM leg_instance
ORDER BY departure_time;

```

The 'Results' pane shows the output of the query, displaying 5 rows of flight leg data:

leg_instance_id	number_of_available_seats	arrival_time	departure_time	flight_leg_date	legno	airplaneID
1	150	10:05:00.0000000	08:00:00.0000000	2025-01-10	1	1
3	140	12:15:00.0000000	09:00:00.0000000	2025-01-12	3	3
3	200	12:10:00.0000000	11:00:00.0000000	2025-01-11	2	2
4	270	18:20:00.0000000	14:00:00.0000000	2025-01-13	4	4
5	300	18:40:00.0000000	16:00:00.0000000	2025-01-14	5	5

The status bar at the bottom indicates 'Query executed successfully' and '5 rows'.

10. Display the maximum, minimum, and average available seats.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with the 'Airline' database selected. The right pane shows a SQL query window with the following query:

```

SELECT airplaneID
FROM airplane
WHERE total_no_seats > 300;

SELECT airportID, airportname
FROM airport
WHERE city = 'Cairo';

SELECT *
FROM leg_instance
WHERE flight_leg_date = '2025-06-18';

SELECT *
FROM leg_instance
ORDER BY departure_time;

SELECT
MAX(number_of_available_seats) AS MaxSeats,
MIN(number_of_available_seats) AS MinSeats,
AVG(number_of_available_seats) AS AvgSeats
FROM leg_instance;

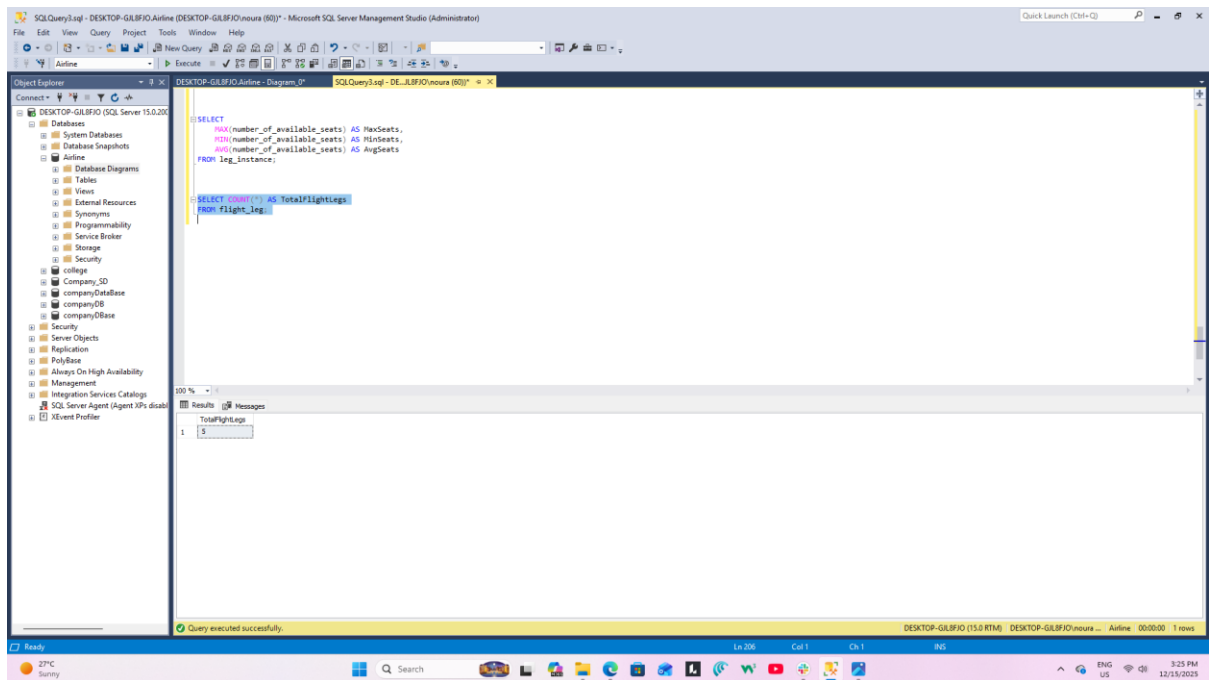
```

The 'Results' pane shows the output of the query, displaying 1 row of summary data:

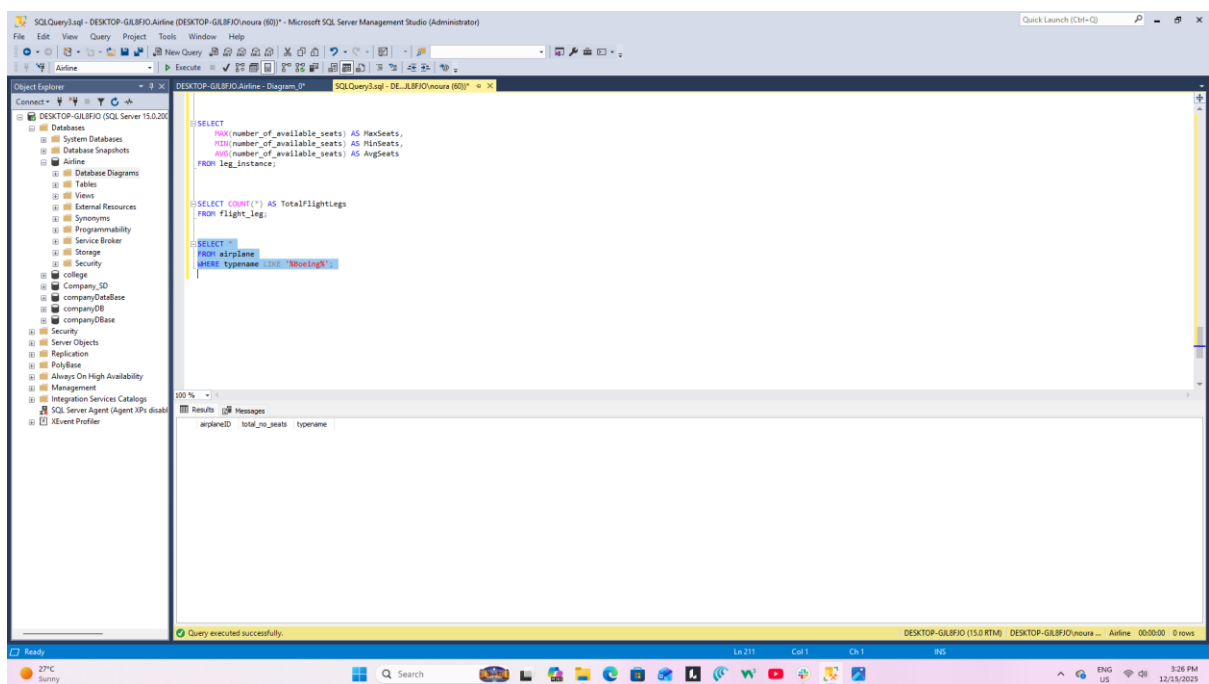
MaxSeats	MinSeats	AvgSeats
300	140	212

The status bar at the bottom indicates 'Query executed successfully' and '1 rows'.

11. Display total number of flight legs.

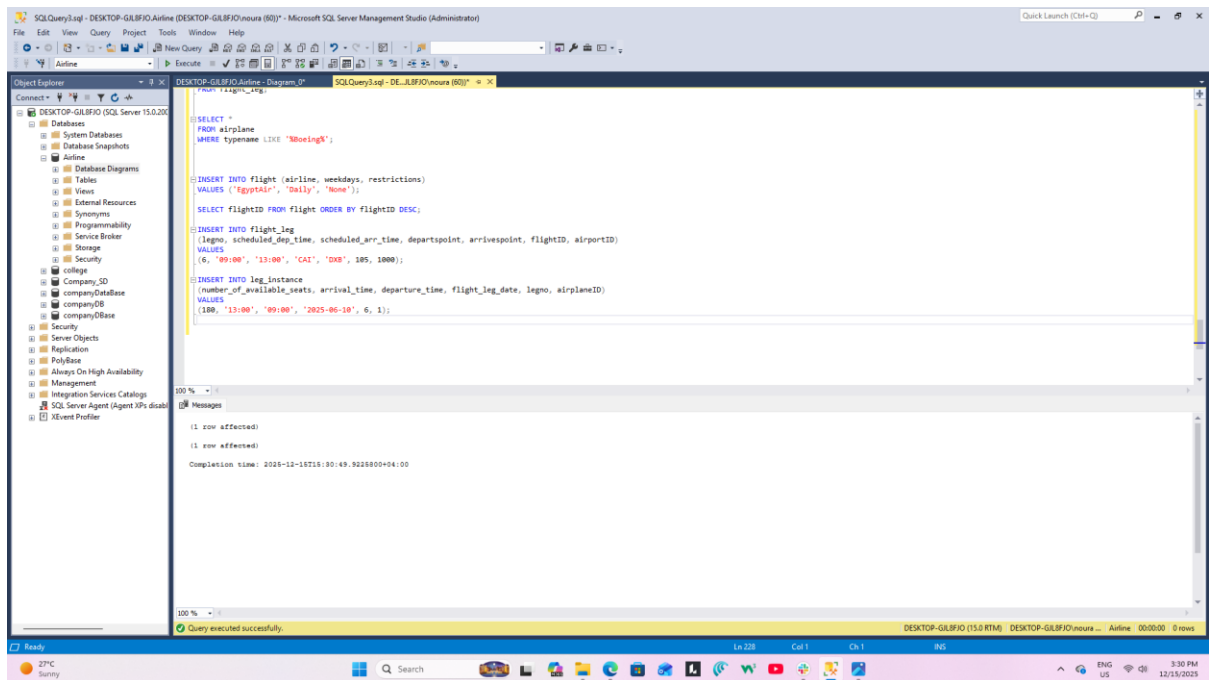


12. Display airplanes whose type contains 'Boeing'.



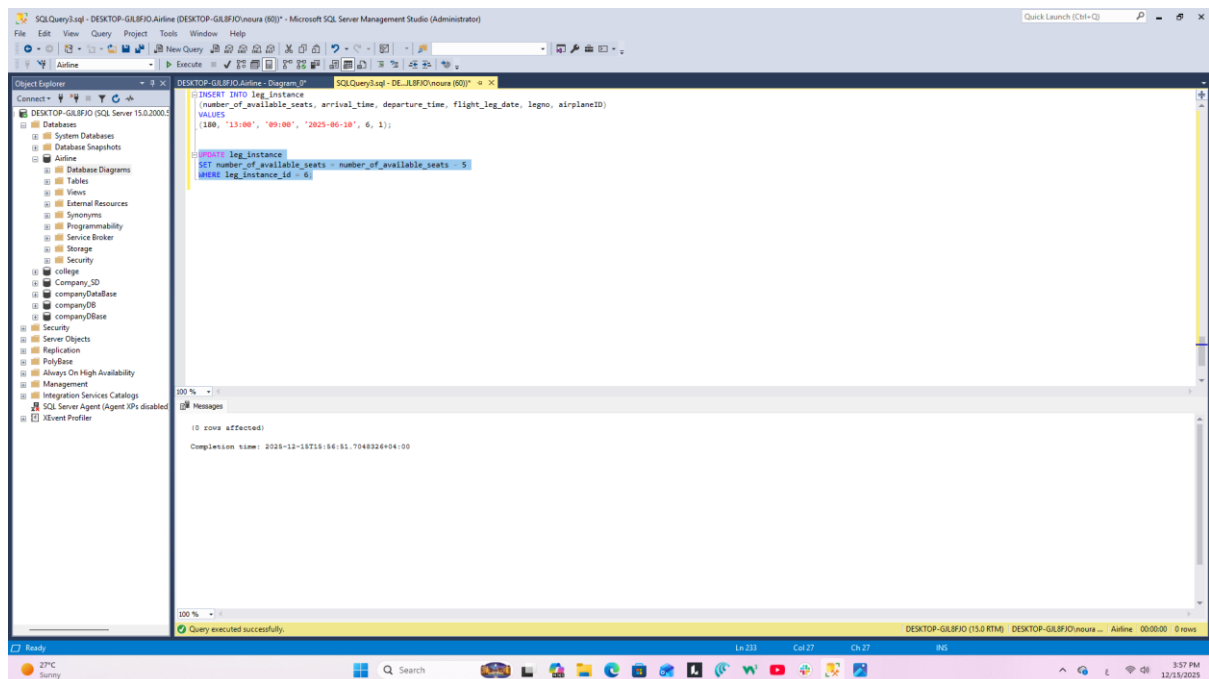
DML

13. Insert a new flight leg departing from 'CAI' to 'DXB' on 2025-06-10.

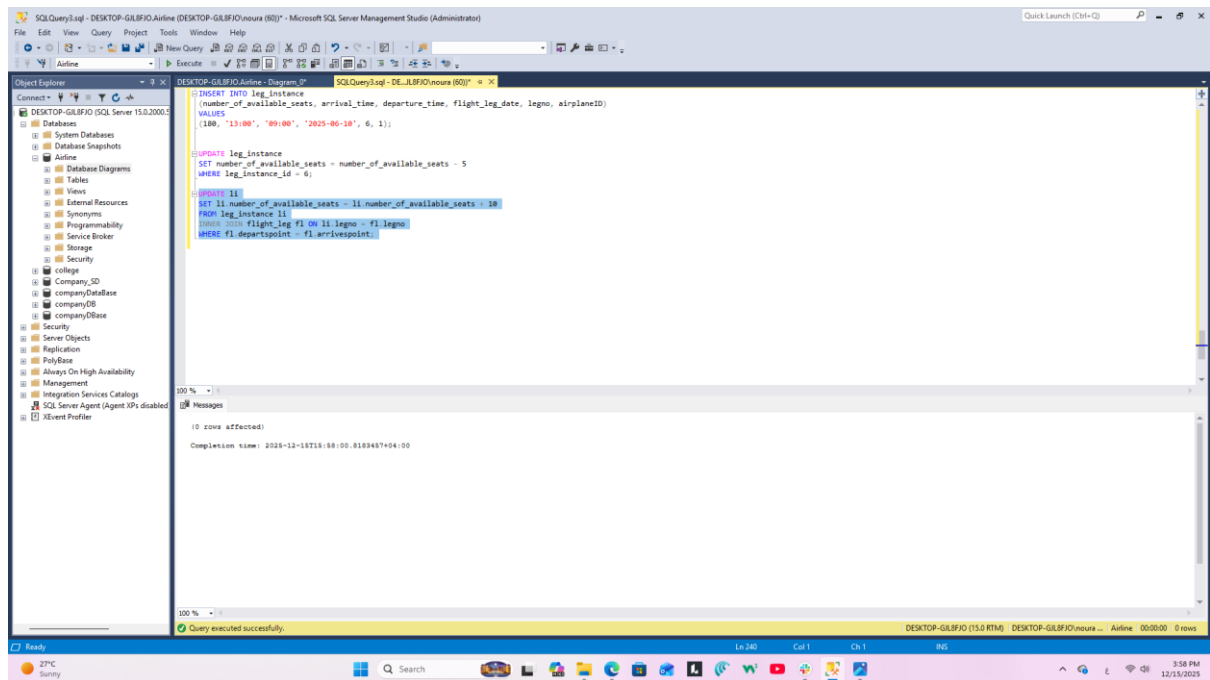


14. Insert a customer with NULL contact number.

15. Reduce available seats of your inserted flight leg by 5.



16. Increase available seats by 10 for all domestic flights.



17. Update airplane seat capacity by +20 where capacity < 150.

