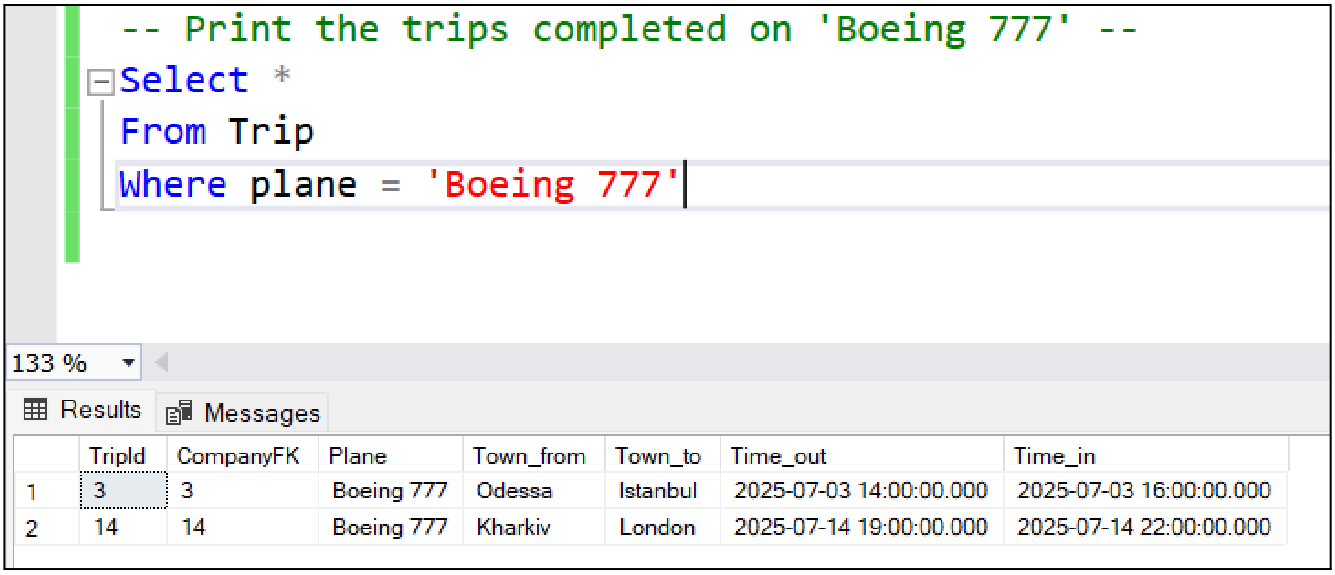
Sample SQL Tasks for Airline Management System

Task 1: Print the trips completed on Boeing 777

Used:

1. Operators: SELECT, WHERE, =
2. Selecting all columns
3. Filtering rows by exact match



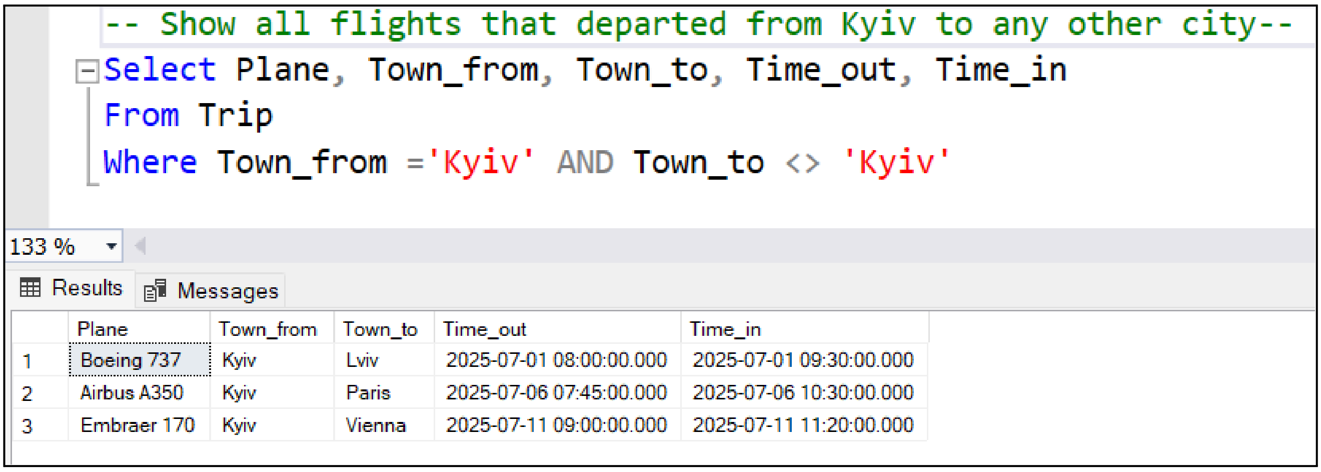
```
-- Print the trips completed on 'Boeing 777' --
Select *
From Trip
Where plane = 'Boeing 777'
```

	TripId	CompanyFK	Plane	Town_from	Town_to	Time_out	Time_in
1	3	3	Boeing 777	Odessa	Istanbul	2025-07-03 14:00:00.000	2025-07-03 16:00:00.000
2	14	14	Boeing 777	Kharkiv	London	2025-07-14 19:00:00.000	2025-07-14 22:00:00.000

Task 2: Show all flights that departed from Kyiv to any other city

Used:

1. Operators SELECT, FROM, WHERE, AND, <>
2. Inequality comparison (<>)
3. Filtering rows by multiple conditions



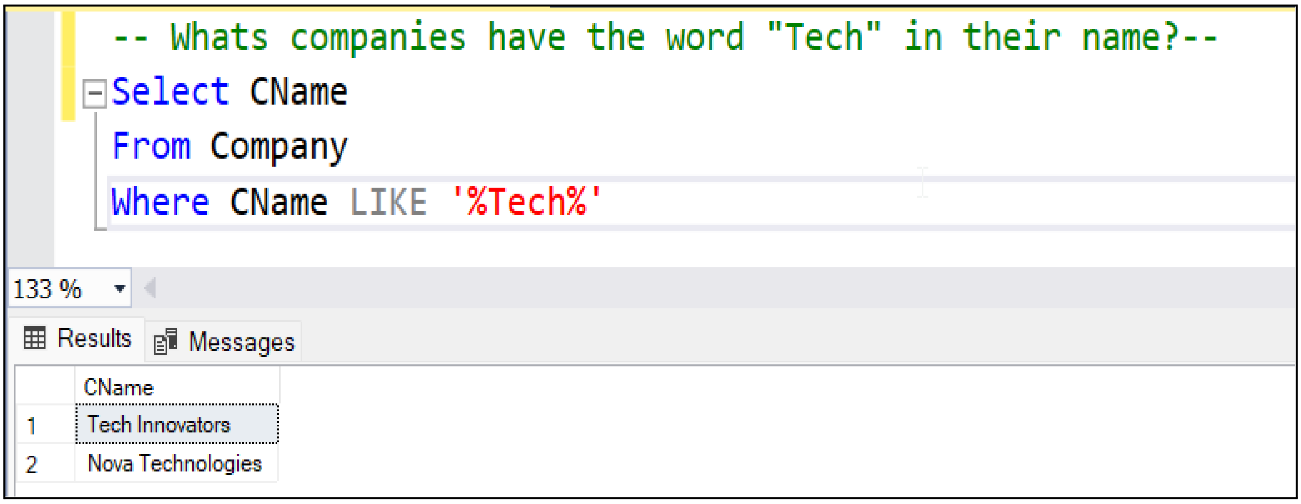
```
-- Show all flights that departed from Kyiv to any other city--
Select Plane, Town_from, Town_to, Time_out, Time_in
From Trip
Where Town_from ='Kyiv' AND Town_to <> 'Kyiv'
```

	Plane	Town_from	Town_to	Time_out	Time_in
1	Boeing 737	Kyiv	Lviv	2025-07-01 08:00:00.000	2025-07-01 09:30:00.000
2	Airbus A350	Kyiv	Paris	2025-07-06 07:45:00.000	2025-07-06 10:30:00.000
3	Embraer 170	Kyiv	Vienna	2025-07-11 09:00:00.000	2025-07-11 11:20:00.000

Task 3. What companies have the word "Tech" in their name?

Used:

1. Operators: SELECT, FROM, WHERE, LIKE
2. Text pattern matching (LIKE '%Tech%')



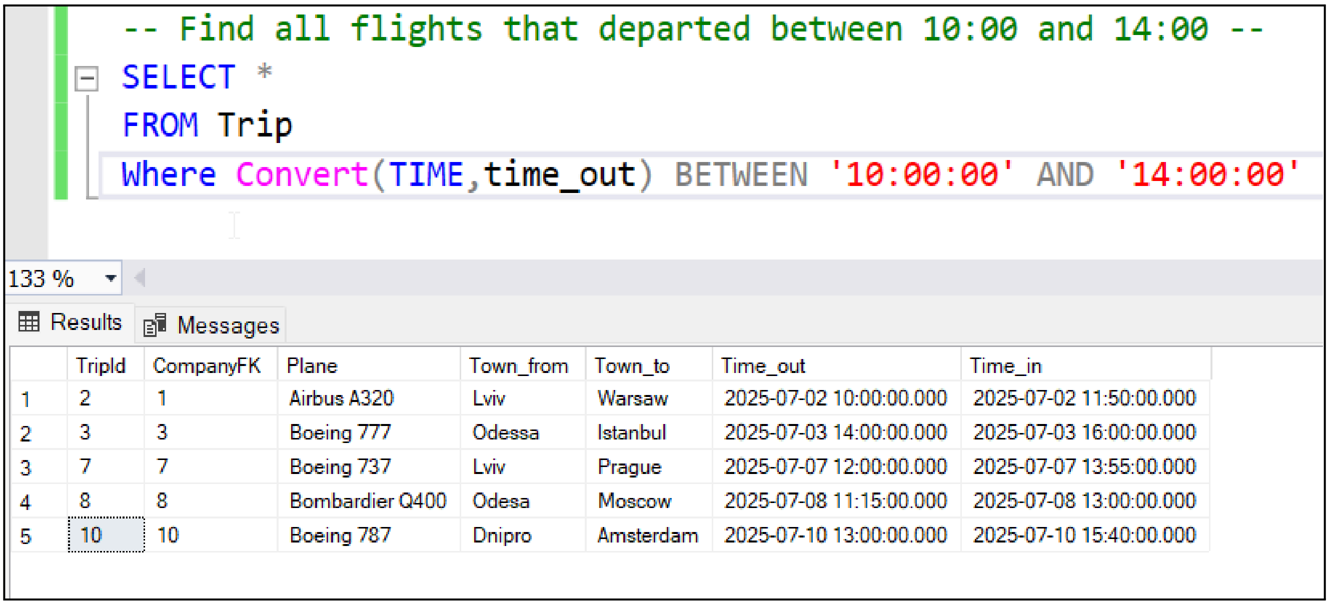
```
-- Whats companies have the word "Tech" in their name?--
Select CName
From Company
Where CName LIKE '%Tech%'
```

	CName
1	Tech Innovators
2	Nova Technologies

Task 4: Find all flights that departed between 10:00 and 14:00

Used:

1. Operators: SELECT, FROM, WHERE, BETWEEN
2. Data type conversion using CONVERT function (casting time_out to TIME)
3. Filtering rows by a range of values (BETWEEN '10:00:00' AND '14:00:00')



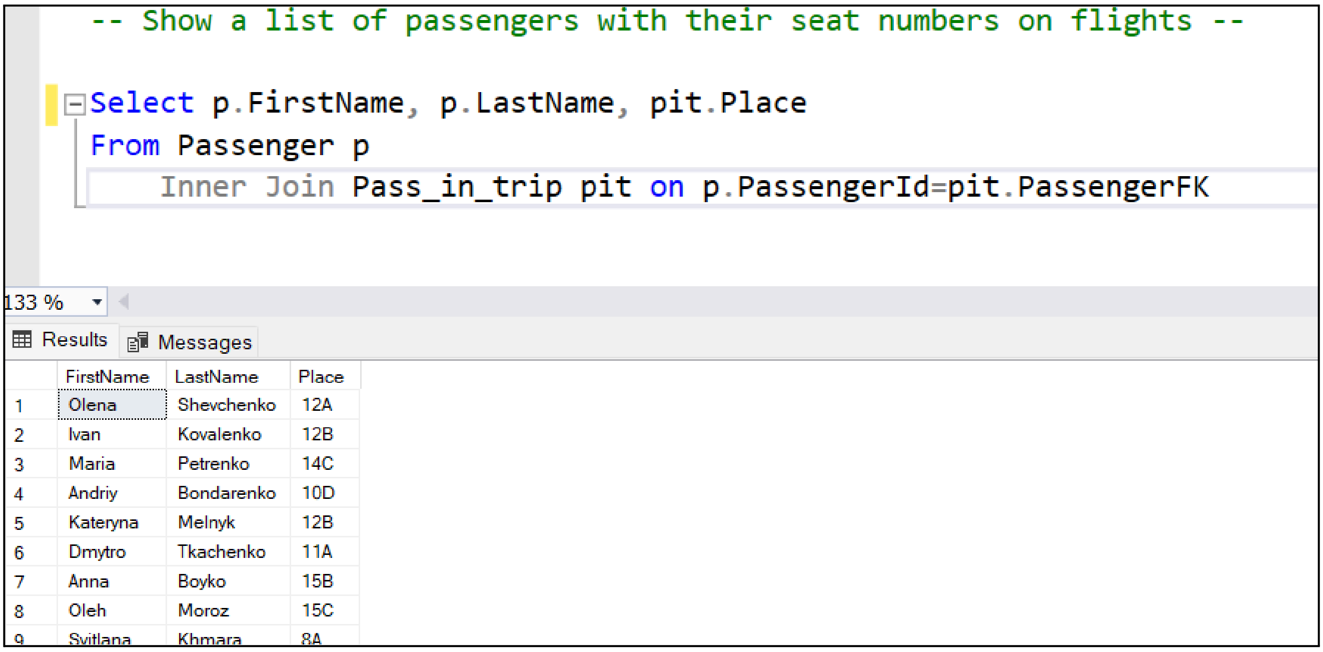
```
-- Find all flights that departed between 10:00 and 14:00 --
SELECT *
FROM Trip
Where Convert(TIME,time_out) BETWEEN '10:00:00' AND '14:00:00'
```

	TripId	CompanyFK	Plane	Town_from	Town_to	Time_out	Time_in
1	2	1	Airbus A320	Lviv	Warsaw	2025-07-02 10:00:00.000	2025-07-02 11:50:00.000
2	3	3	Boeing 777	Odessa	Istanbul	2025-07-03 14:00:00.000	2025-07-03 16:00:00.000
3	7	7	Boeing 737	Lviv	Prague	2025-07-07 12:00:00.000	2025-07-07 13:55:00.000
4	8	8	Bombardier Q400	Odesa	Moscow	2025-07-08 11:15:00.000	2025-07-08 13:00:00.000
5	10	10	Boeing 787	Dnipro	Amsterdam	2025-07-10 13:00:00.000	2025-07-10 15:40:00.000

Task 5. Show a list of passengers with their seat numbers on flights

Used:

1. Operators: SELECT, FROM, INNER JOIN
2. Joining two tables (Passenger and Pass_in_trip) using a foreign key



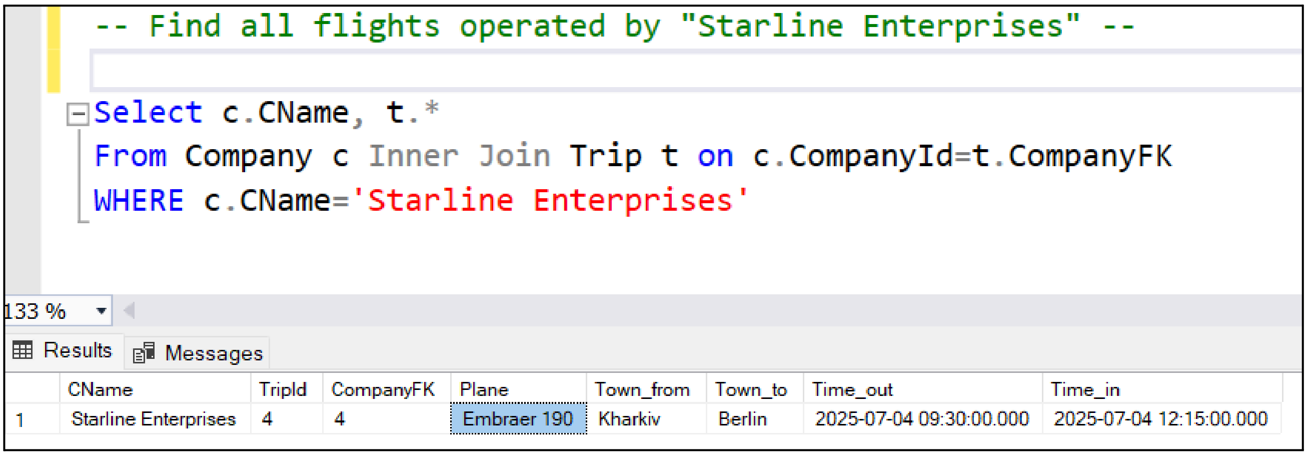
```sql
-- Show a list of passengers with their seat numbers on flights --

Select p.FirstName, p.LastName, pit.Place
From Passenger p
    Inner Join Pass_in_trip pit on p.PassengerId=pit.PassengerFK
```

	FirstName	LastName	Place
1	Olena	Shevchenko	12A
2	Ivan	Kovalenko	12B
3	Maria	Petrenko	14C
4	Andriy	Bondarenko	10D
5	Kateryna	Melnyk	12B
6	Dmytro	Tkachenko	11A
7	Anna	Boyko	15B
8	Oleh	Moroz	15C
9	Svitlana	Khmara	8A

Task 6. Find all flights operated by "Starline Enterprises"

Used:

1. Operators: SELECT, FROM, WHERE, INNER JOIN
2. Joining two tables (Company and Trip) based on a foreign key
3. Filtering rows by exact text match (c.CName = 'Starline Enterprises')



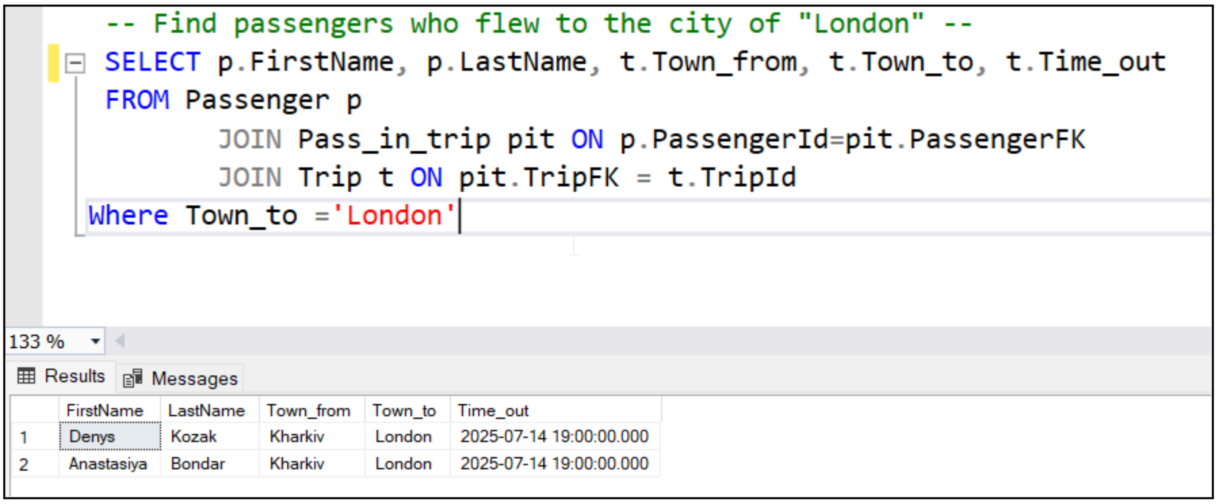
```sql
-- Find all flights operated by "Starline Enterprises" --

Select c.CName, t.*
From Company c Inner Join Trip t on c.CompanyId=t.CompanyFK
WHERE c.CName='Starline Enterprises'
```

	CName	TripId	CompanyFK	Plane	Town_from	Town_to	Time_out	Time_in
1	Starline Enterprises	4	4	Embraer 190	Kharkiv	Berlin	2025-07-04 09:30:00.000	2025-07-04 12:15:00.000

Task 7. Find passengers who flew to the city of "London"

Used:

1. Operators: SELECT, FROM, WHERE, INNER JOIN
2. Multiple table joins (Passenger, Pass_in_trip, Trip) using foreign keys
3. Filtering by exact match (Town_to = 'London')
4. Selecting specific fields across joined tables



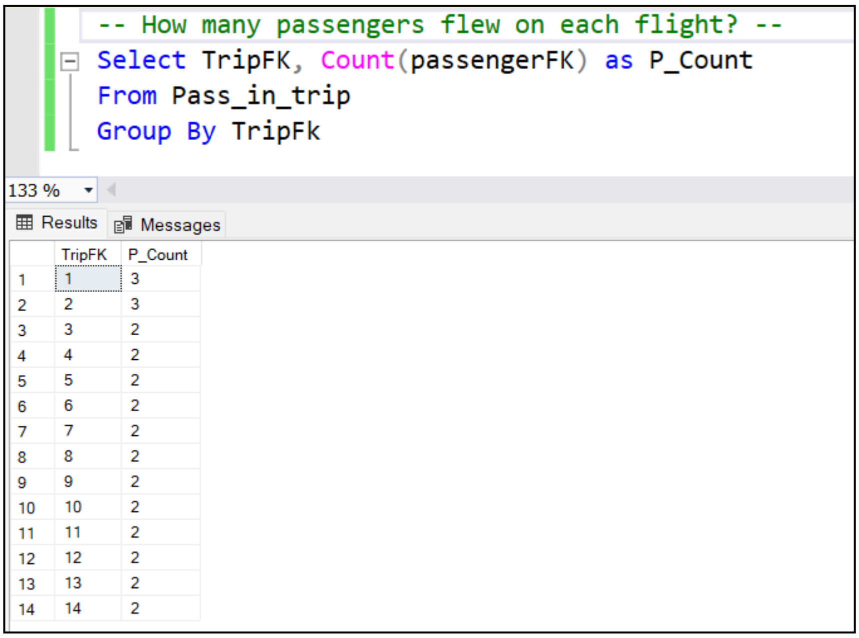
```
-- Find passengers who flew to the city of "London" --
SELECT p.FirstName, p.LastName, t.Town_from, t.Town_to, t.Time_out
FROM Passenger p
        JOIN Pass_in_trip pit ON p.PassengerId=pit.PassengerFK
        JOIN Trip t ON pit.TripFK = t.TripId
Where Town_to ='London'
```

	FirstName	LastName	Town_from	Town_to	Time_out
1	Denys	Kozak	Kharkiv	London	2025-07-14 19:00:00.000
2	Anastasiya	Bondar	Kharkiv	London	2025-07-14 19:00:00.000

Task 8: How many passengers flew on each flight?

Used:

1. Operators: SELECT, FROM, GROUP BY
2. Aggregate function: COUNT() — counts passengers per flight (TripFK)
3. Grouping rows by flight ID to calculate counts per group (GROUP BY TripFK)
4. Aliasing column (AS P_Count) for readability



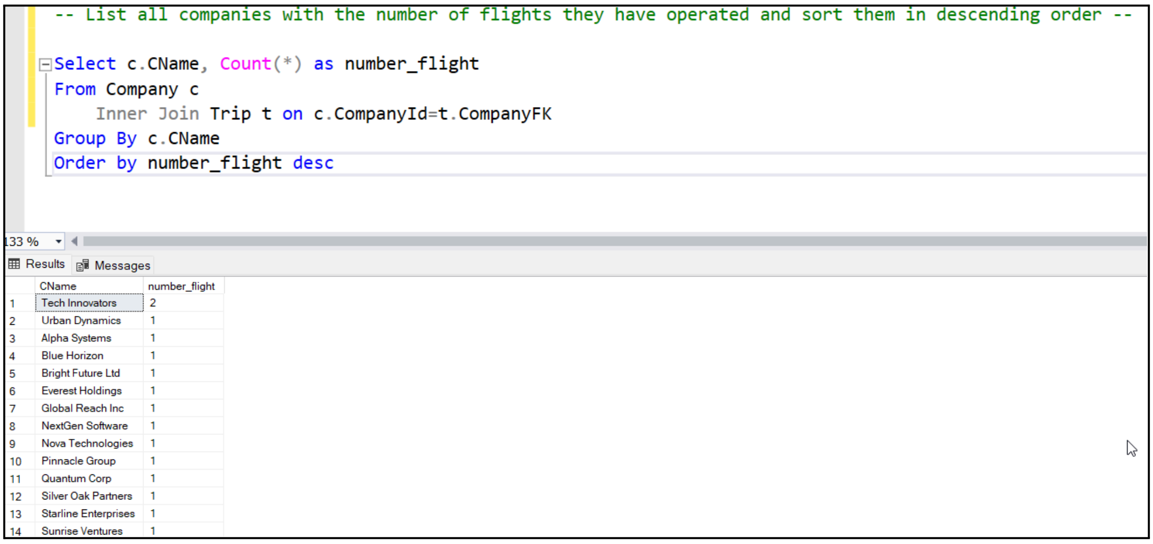
```
-- How many passengers flew on each flight? --
Select TripFK, Count(passengerFK) as P_Count
From Pass_in_trip
Group By TripFk
```

	TripFK	P_Count
1	1	3
2	2	3
3	3	2
4	4	2
5	5	2
6	6	2
7	7	2
8	8	2
9	9	2
10	10	2
11	11	2
12	12	2
13	13	2
14	14	2

Task 9. List all companies with the number of flights they have operated and sort them in descending order

Used:

1. Operators: SELECT, FROM, INNER JOIN, GROUP BY, ORDER BY
2. Aggregate function: COUNT(*) — counts the number of flights per company
3. Joining Company and Trip tables via foreign key
4. Grouping by company name to get aggregated flight count
5. Sorting results in descending order by number of flight



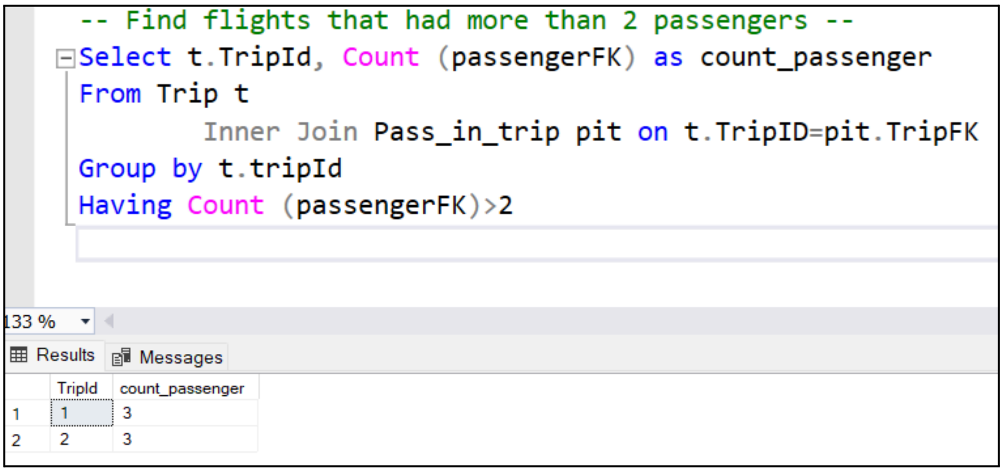
```sql
-- List all companies with the number of flights they have operated and sort them in descending order --

Select c.CName, Count(*) as number_flight
From Company c
    Inner Join Trip t on c.CompanyId=t.CompanyFK
Group By c.CName
Order by number_flight desc
```

	CName	number_flight
1	Tech Innovators	2
2	Urban Dynamics	1
3	Alpha Systems	1
4	Blue Horizon	1
5	Bright Future Ltd	1
6	Everest Holdings	1
7	Global Reach Inc	1
8	NextGen Software	1
9	Nova Technologies	1
10	Pinnacle Group	1
11	Quantum Corp	1
12	Silver Oak Partners	1
13	Starline Enterprises	1
14	Sunrise Ventures	1

Task 10: Find flights that had more than 2 passengers

Used:

1. Operators: SELECT, FROM, INNER JOIN, GROUP BY, HAVING
2. Aggregate function: COUNT() — counts passengers per flight
3. Filtering aggregated results using HAVING (only flights with more than 2 passengers)
4. Grouping data by TripId to apply aggregation



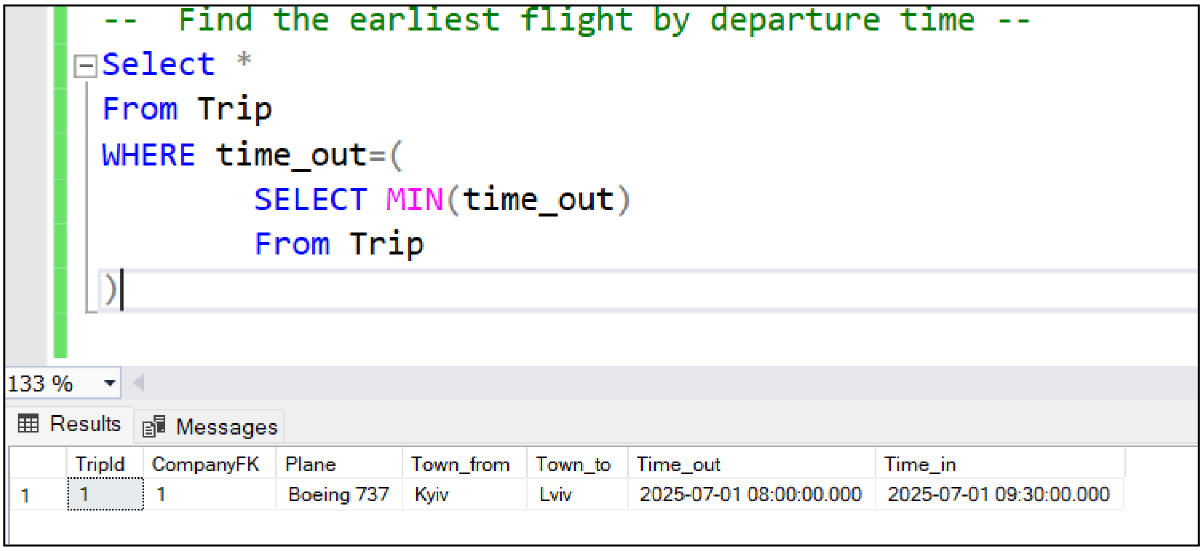
```sql
-- Find flights that had more than 2 passengers --
Select t.TripId, Count (passengerFK) as count_passenger
From Trip t
        Inner Join Pass_in_trip pit on t.TripID=pit.TripFK
Group by t.tripId
Having Count (passengerFK)>2
```

	TripId	count_passenger
1	1	3
2	2	3

Task 11. Find the earliest flight by departure time

Used:

1. Operators: SELECT, FROM, WHERE, =
2. Aggregate function: MIN() — finds the earliest time_out value
3. Subquery — used inside WHERE to compare against a calculated value



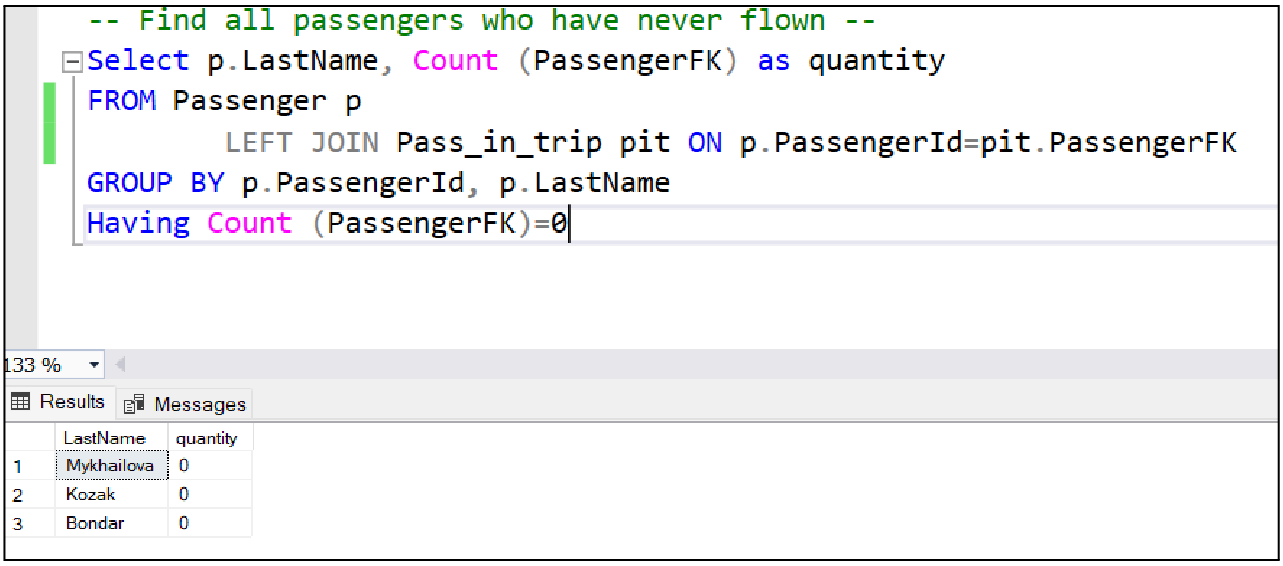
```
--  Find the earliest flight by departure time --
Select *
From Trip
WHERE time_out=(
        SELECT MIN(time_out)
        From Trip
)
```

	TripId	CompanyFK	Plane	Town_from	Town_to	Time_out	Time_in
1	1	1	Boeing 737	Kyiv	Lviv	2025-07-01 08:00:00.000	2025-07-01 09:30:00.000

Task 12: Find all passengers who have never flown

Used:

1. Operators: SELECT, FROM, LEFT JOIN, COUNT, GROUP BY, HAVING
2. Aggregate function: COUNT() — counts how many times a passenger appears in trips
3. LEFT JOIN — includes all passengers, even if they have no trips
4. HAVING — filters grouped results to show only those with zero flights
5. Identifies passengers who have never flown



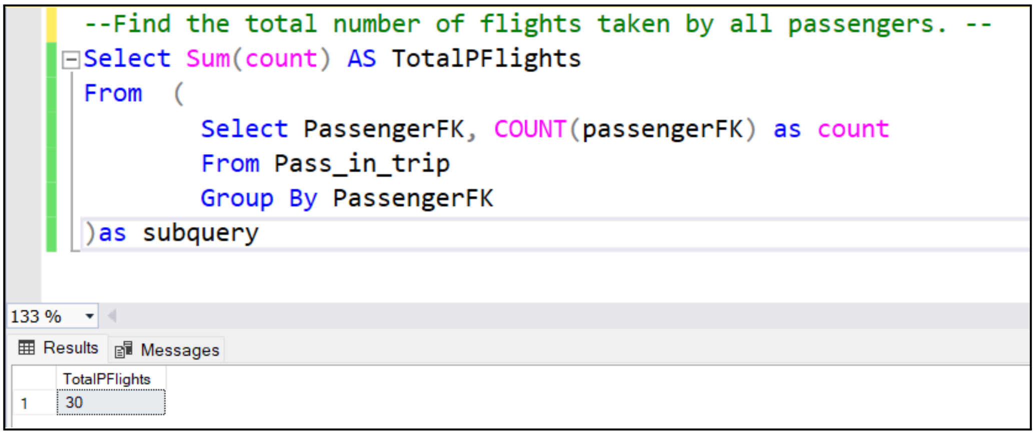
```
-- Find all passengers who have never flown --
Select p.LastName, Count (PassengerFK) as quantity
FROM Passenger p
        LEFT JOIN Pass_in_trip pit ON p.PassengerId=pit.PassengerFK
GROUP BY p.PassengerId, p.LastName
Having Count (PassengerFK)=0
```

	LastName	quantity
1	Mykhailova	0
2	Kozak	0
3	Bondar	0

Task 13. Find the total number of flights taken by all passengers

Used:

1. Operators: SELECT, FROM, GROUP BY, SUM()
2. Aggregate functions:
   – COUNT() — counts flights per passenger
   – SUM() — totals all individual passenger flight counts
3. Subquery — calculates flight count per passenger and passes results to the outer query
4. Aliasing: AS count (inner), AS TotalPFlights (outer) for column naming clarity



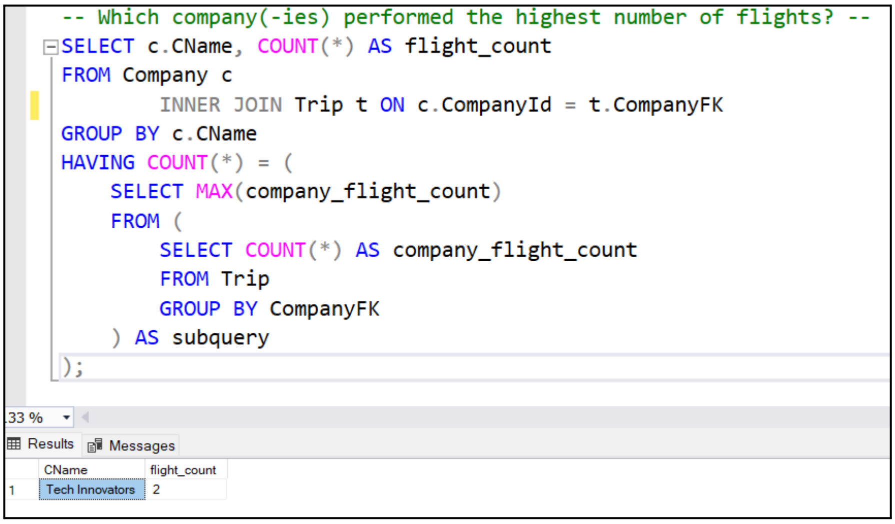
```
--Find the total number of flights taken by all passengers. --
Select Sum(count) AS TotalPFlights
From  (
        Select PassengerFK, COUNT(passengerFK) as count
        From Pass_in_trip
        Group By PassengerFK
)as subquery
```

```
   TotalPFlights
1  30
```

Task 14. Which company(-ies) performed the highest number of flights?

Used:

1. Operators: SELECT, FROM, INNER JOIN, GROUP BY, COUNT, HAVING
2. Aggregate functions:
   – COUNT(*) — counts the number of flights per company
   – MAX() — gets the maximum number of flights among companies
3. Nested subqueries — to calculate the max flight count across all companies
4. Filtering with HAVING to return only companies matching the max count



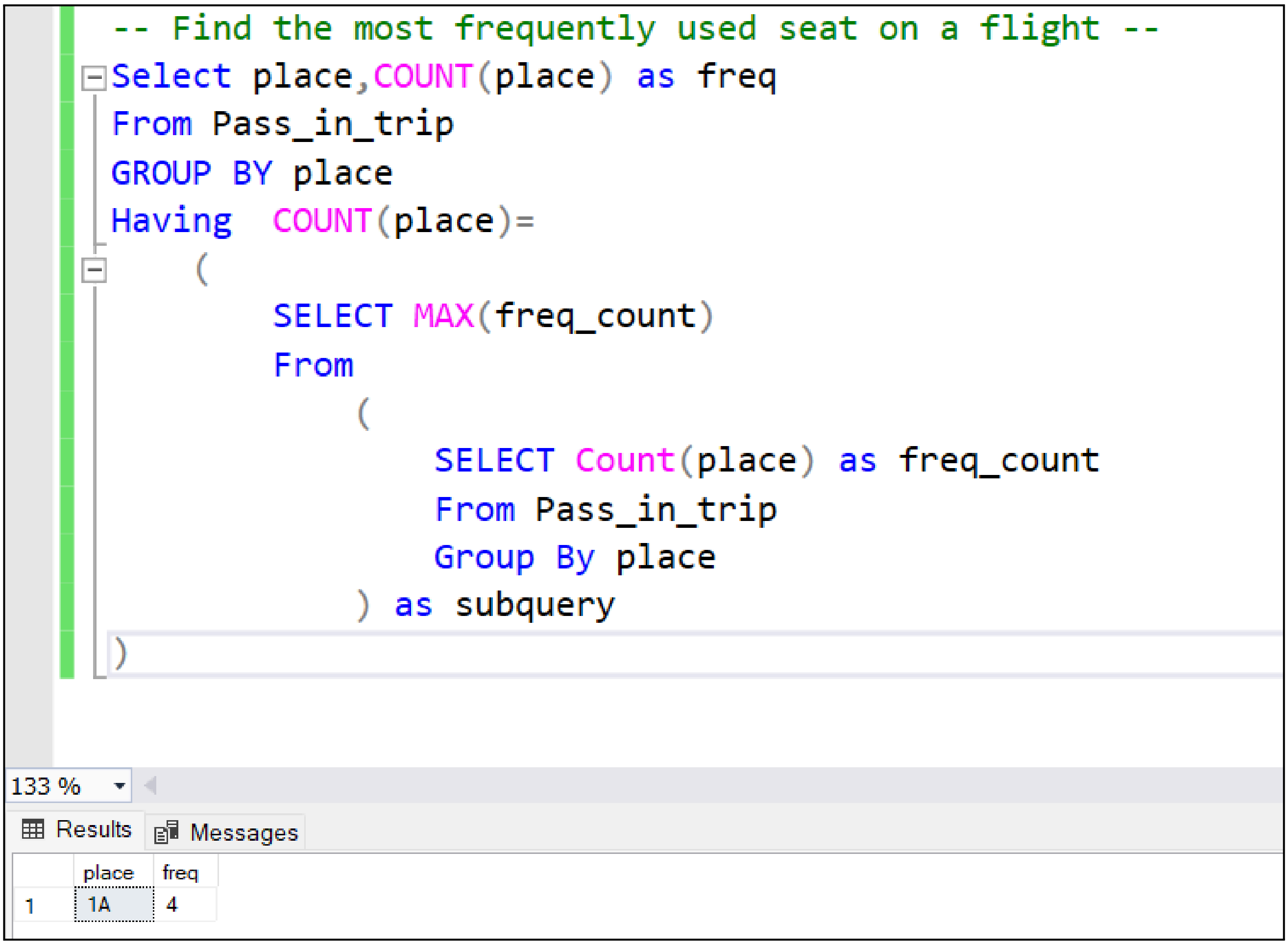
```
-- Which company(-ies) performed the highest number of flights? --
SELECT c.CName, COUNT(*) AS flight_count
FROM Company c
        INNER JOIN Trip t ON c.CompanyId = t.CompanyFK
GROUP BY c.CName
HAVING COUNT(*) = (
    SELECT MAX(company_flight_count)
    FROM (
        SELECT COUNT(*) AS company_flight_count
        FROM Trip
        GROUP BY CompanyFK
    ) AS subquery
);
```

	CName	flight_count
1	Tech Innovators	2

Task 15 Find the most frequently used seat on a flight

Used:

1. Operators: SELECT, FROM, GROUP BY, HAVING
2. Aggregate functions:
   – COUNT() — counts how many times each seat/place was used
   – MAX() — identifies the highest usage count
3. Subquery — calculates the frequency of each seat, then finds the max
4. HAVING — filters to return only the seat(s) with the highest usage



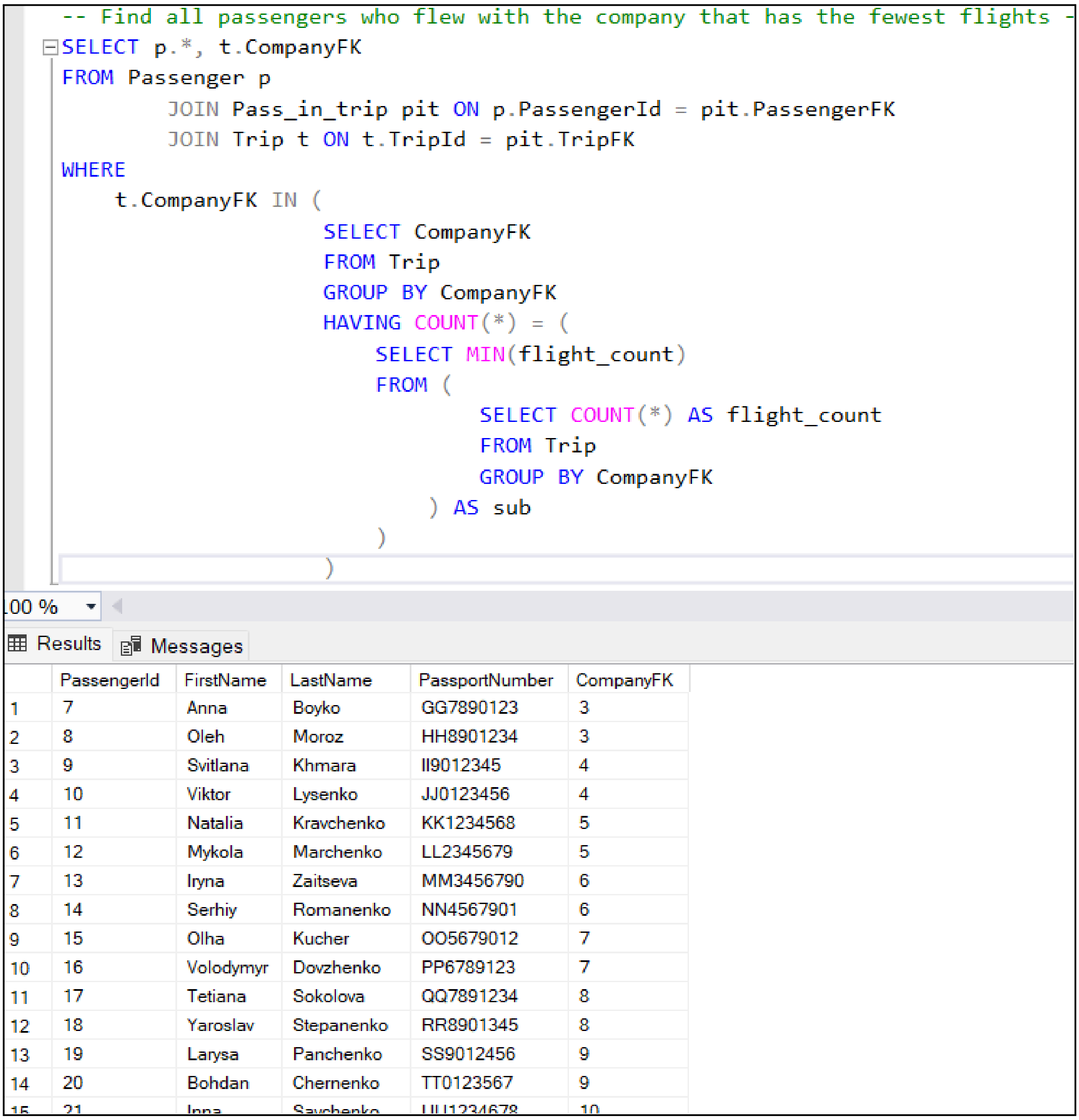
```
-- Find the most frequently used seat on a flight --
Select place,COUNT(place) as freq
From Pass_in_trip
GROUP BY place
Having  COUNT(place)=
    (
        SELECT MAX(freq_count)
        From
            (
                SELECT Count(place) as freq_count
                From Pass_in_trip
                Group By place
            ) as subquery
)
```

	place	freq
1	1A	4

Task 16. Find all passengers who flew with the company that has the fewest flights

Used:

1. Operators: SELECT, FROM, JOIN, WHERE, IN, GROUP BY, HAVING
2. Aggregate functions:
3. – COUNT() — counts the number of flights per company
4. – MIN() — identifies the lowest number of flights operated
5. Nested subqueries — used to find the company (or companies) with the fewest flights
6. IN — filters passengers who flew with those companies
7. Multiple joins across Passenger, Pass_in_trip, and Trip to retrieve relevant data



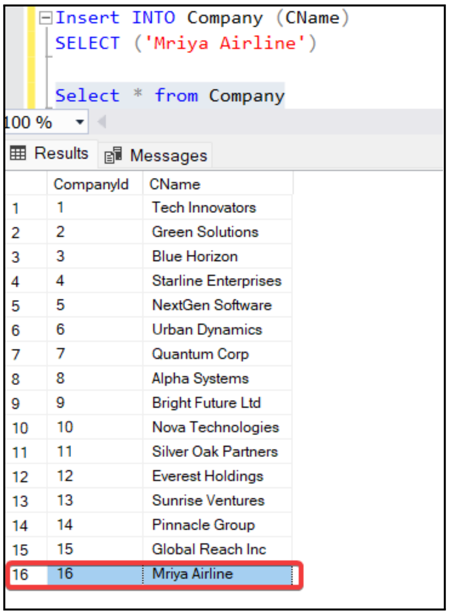
```
-- Find all passengers who flew with the company that has the fewest flights -
SELECT p.*, t.CompanyFK
FROM Passenger p
        JOIN Pass_in_trip pit ON p.PassengerId = pit.PassengerFK
        JOIN Trip t ON t.TripId = pit.TripFK
WHERE
    t.CompanyFK IN (
                    SELECT CompanyFK
                    FROM Trip
                    GROUP BY CompanyFK
                    HAVING COUNT(*) = (
                        SELECT MIN(flight_count)
                        FROM (
                                SELECT COUNT(*) AS flight_count
                                FROM Trip
                                GROUP BY CompanyFK
                            ) AS sub
                        )
                )
```

	PassengerId	FirstName	LastName	PassportNumber	CompanyFK
1	7	Anna	Boyko	GG7890123	3
2	8	Oleh	Moroz	HH8901234	3
3	9	Svitlana	Khmara	II9012345	4
4	10	Viktor	Lysenko	JJ0123456	4
5	11	Natalia	Kravchenko	KK1234568	5
6	12	Mykola	Marchenko	LL2345679	5
7	13	Iryna	Zaitseva	MM3456790	6
8	14	Serhiy	Romanenko	NN4567901	6
9	15	Olha	Kucher	OO5679012	7
10	16	Volodymyr	Dovzhenko	PP6789123	7
11	17	Tetiana	Sokolova	QQ7891234	8
12	18	Yaroslav	Stepanenko	RR8901345	8
13	19	Larysa	Panchenko	SS9012456	9
14	20	Bohdan	Chernenko	TT0123567	9
15	21	Inna	Savchenko	UU1234678	10

Task 17. Insert a new airline company into the Company table.

Used:

1. Operators: INSERT INTO, SELECT, FROM
2. INSERT INTO ... SELECT — inserts a new row using a SELECT statement
3. SELECT * — used afterward to verify that the new record was added



```sql
Insert INTO Company (CName)
SELECT ('Mriya Airline')

Select * from Company
```

100 %

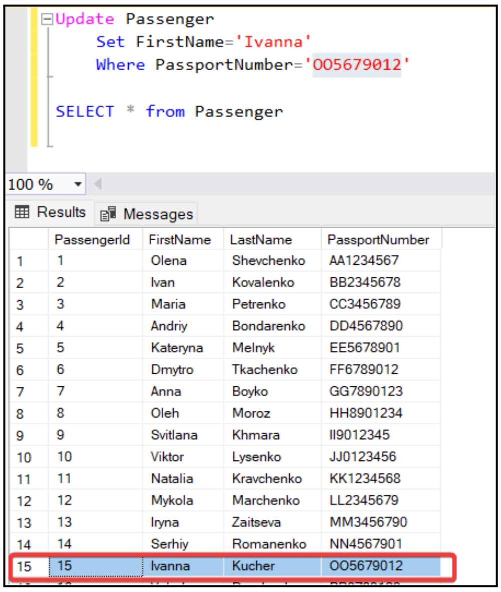
Results | Messages

	CompanyId	CName
1	1	Tech Innovators
2	2	Green Solutions
3	3	Blue Horizon
4	4	Starline Enterprises
5	5	NextGen Software
6	6	Urban Dynamics
7	7	Quantum Corp
8	8	Alpha Systems
9	9	Bright Future Ltd
10	10	Nova Technologies
11	11	Silver Oak Partners
12	12	Everest Holdings
13	13	Sunrise Ventures
14	14	Pinnacle Group
15	15	Global Reach Inc
16	16	Mriya Airline

Task 18. Update the first name of a passenger based on their passport number.

Used:

1. Operators: UPDATE, SET, WHERE, SELECT *
2. UPDATE ... SET — modifies the value of a specific column
3. WHERE — filters the row to update based on a condition (PassportNumber = 'OO5679012')
4. SELECT * — retrieves all rows to verify the update operation



```sql
Update Passenger
    Set FirstName='Ivanna'
    Where PassportNumber='OO5679012'

SELECT * from Passenger
```

100 %

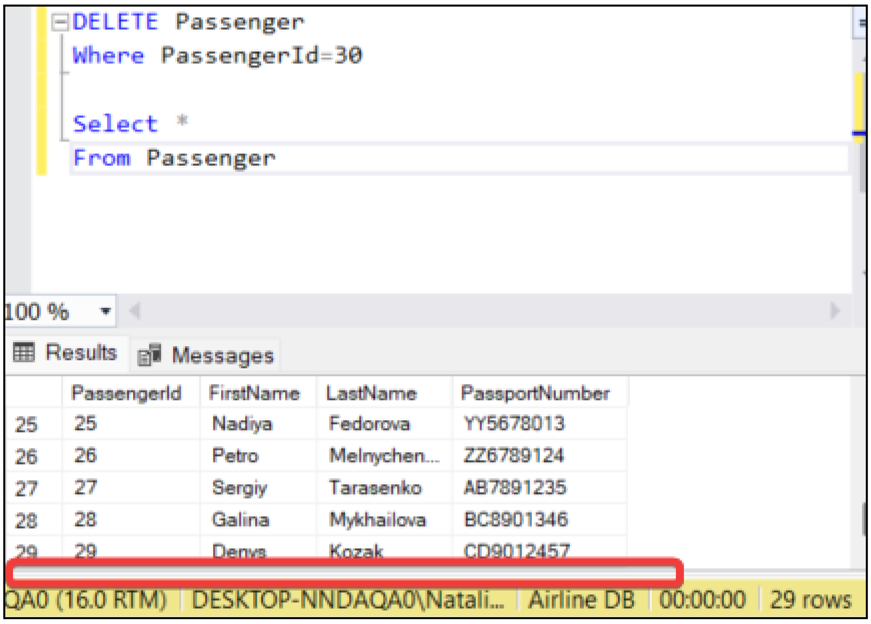
Results | Messages

	PassengerId	FirstName	LastName	PassportNumber
1	1	Olena	Shevchenko	AA1234567
2	2	Ivan	Kovalenko	BB2345678
3	3	Maria	Petrenko	CC3456789
4	4	Andriy	Bondarenko	DD4567890
5	5	Kateryna	Melnyk	EE5678901
6	6	Dmytro	Tkachenko	FF6789012
7	7	Anna	Boyko	GG7890123
8	8	Oleh	Moroz	HH8901234
9	9	Svitlana	Khmara	II9012345
10	10	Viktor	Lysenko	JJ0123456
11	11	Natalia	Kravchenko	KK1234568
12	12	Mykola	Marchenko	LL2345679
13	13	Iryna	Zaitseva	MM3456790
14	14	Serhiy	Romanenko	NN4567901
15	15	Ivanna	Kucher	OO5679012

Task 19: Delete a passenger by ID and verify the result.

Used:

1. Operators: DELETE, WHERE, SELECT *
2. DELETE ... WHERE — removes a specific row from the Passenger table based on ID
3. WHERE — ensures only the targeted row is deleted (PassengerId = 30)
4. SELECT * — used afterward to verify that the row has been successfully removed



```
DELETE Passenger
Where PassengerId=30

Select *
From Passenger
```

	PassengerId	FirstName	LastName	PassportNumber
25	25	Nadiya	Fedorova	YY5678013
26	26	Petro	Melnychen...	ZZ6789124
27	27	Sergiy	Tarasenko	AB7891235
28	28	Galina	Mykhailova	BC8901346
29	29	Denys	Kozak	CD9012457

QA0 (16.0 RTM) | DESKTOP-NNDAQA0\Natali... | Airline DB | 00:00:00 | 29 rows