Airline Reservation Database Project

System Overview

- **Objective**
  To design a comprehensive and well-structured database schema for managing airline operations, including flight scheduling, fleet inventory, and passenger reservations.

- **Scope**
  The system supports the complete operational workflow of an airline, from defining flight routes and ensuring airport compatibility to managing aircraft assignments and individual seat reservations.

Why this matters?

- **Operational Safety**
  By enforcing constraints such as **CAN_LAND**, the system guarantees that safety protocols are never violated (e.g., preventing large aircraft from being assigned to runways that cannot accommodate them).
- **Cost Efficiency**
  Optimized scheduling through **LEG_INSTANCE** minimizes data duplication and reduces storage costs. Accurate tracking of **ASSIGNED** aircraft also enables better fleet utilization and more effective maintenance planning.
- **Customer Satisfaction**
  A reliable backend ensures smooth and secure **RESERVATION** processes by preventing double bookings and accurately managing **SEAT** assignments.

Benefits of proposed schema

- **Normalization & Scalability**
  The clear separation between **FLIGHT** (route definition) and **LEG_INSTANCE** (date-specific execution) follows database normalization principles.
  Benefit: Flight routes can be scheduled for many years in advance without re-entering route data, allowing the database to scale efficiently.
- **Robust Data Integrity**
  The use of **strong and weak entities** (e.g., **FLIGHT** and **FLIGHT_LEG**) ensures that incomplete or orphan data cannot exist—for example, a flight leg cannot exist without its parent flight.
- **Flexible Relationship Modeling**
  The schema supports complex real-world airline scenarios, such as **multi-leg flights (stopovers)** and **mixed fleet types**, through the effective use of **1:N** and **M:N** relationships.

Database entities

- **AIRPORT**
  Stores airport-related information such as airport code, city, and state.
- **AIRPLANE**
  Maintains aircraft inventory, including total seat capacity.
- **AIRPLANE_TYPE**
  Contains technical aircraft details such as model, capacity, and manufacturer.
- **FLIGHT**
  Stores flight numbers and associated airline information.

Database entities

- **FLIGHT_LEG**
Represents a **nonstop segment** of a flight.
Example: A flight from Houston to New York with an intermediate stop in Atlanta
consists of multiple legs (stopovers).
- **LEG_INSTANCE**
Represents a specific occurrence of a flight leg on a particular date.
- **FARE**
Manages pricing rules, fare amounts, and restrictions.
- **SEAT**
Handles seat allocation and assignment for reservations.

Key relationships

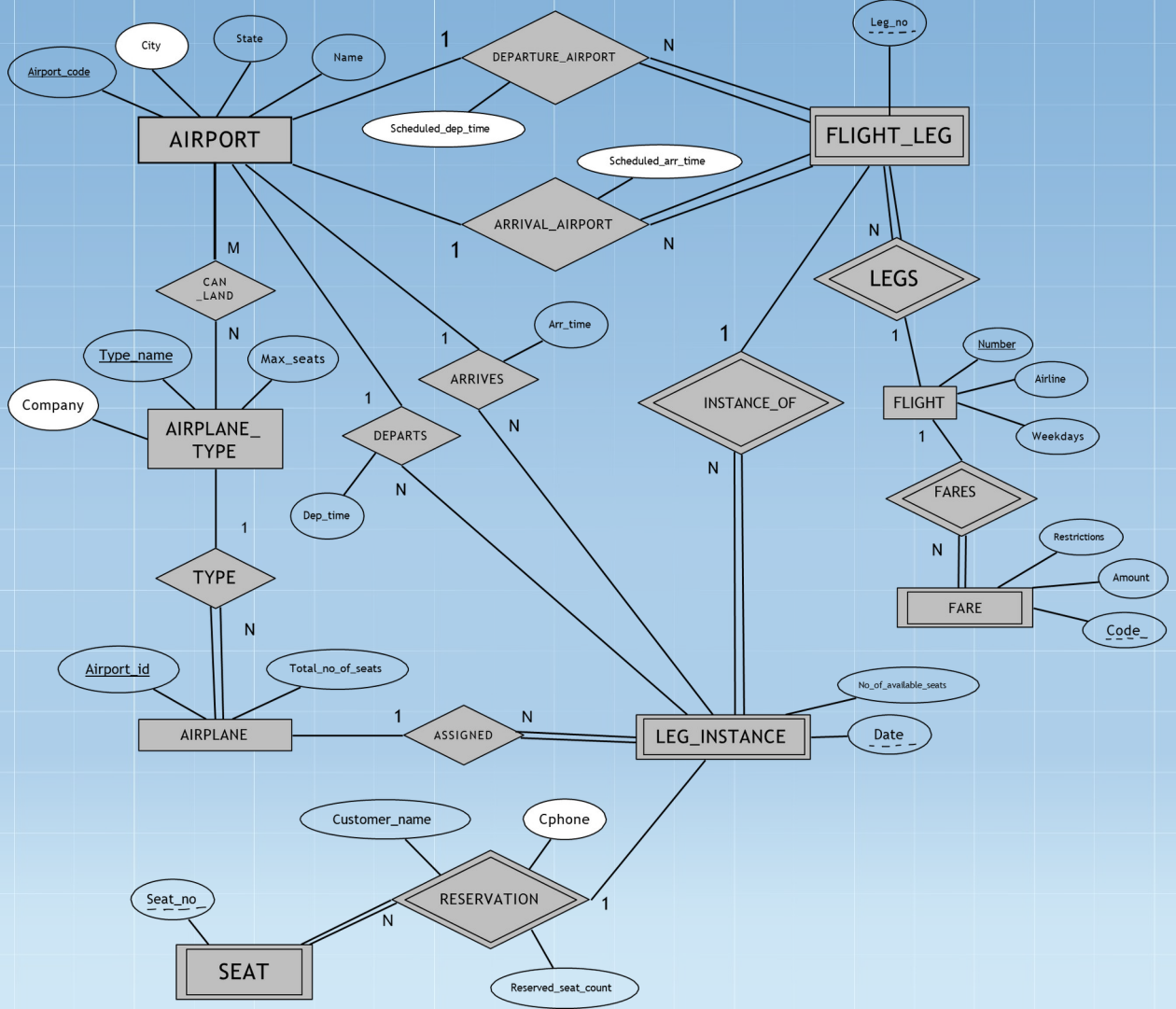
- **CAN_LAND**
  Ensures airport–aircraft compatibility
  to guarantee safe landing operations.
- **INSTANCE_OF**
  Links a flight leg to a specific date,
  enabling accurate schedule
  management.
- **ASSIGNED**
  Assigns a physical aircraft to a flight
  instance
  for efficient fleet tracking and usage.

Key relationships

- **LEGS**
  Connects a main flight to its legs,
  modeling stopovers and weak entities.
- **FARES**
  Associates pricing rules and restrictions
  with flights for revenue management.
- **DEPARTURE_AIRPORT**
  Defines the origin airport
  where a flight leg starts.

Key relationships

- **ARRIVAL_AIRPORT**
  Defines the destination airport where a flight leg ends.
- **RESERVATION**
  Links passenger, seat, and flight instance as a complete booking record.


AIRPORT
Airport_code
City
State
Name
DEPARTURE_AIRPORT
Scheduled_dep_time
FLIGHT_LEG
Leg_no
Scheduled_arr_time
ARRIVAL_AIRPORT
CAN_LAND
AIRPLANE_TYPE
Type_name
Company
Max_seats
ARRIVES
Arr_time
DEPARTS
Dep_time
INSTANCE_OF
LEGS
FLIGHT
Number
Airline
Weekdays
FARES
FARE
Restrictions
Amount
Code_
TYPE
AIRPLANE
Airport_id
Total_no_of_seats
ASSIGNED
LEG_INSTANCE
No_of_available_seats
Date_
Customer_name
Cphone
RESERVATION
Seat_no_
SEAT
Reserved_seat_count
1
N
M

Business rules

1. **FLIGHT SCHEDULING RULE: (FLIGHT_LEG Integrity)**

- **Rule:** The departure airport and the arrival airport for a single flight leg must be logically distinct (they cannot be the same airport).
- **Applies to:** FLIGHT_LEG entity, Departure_Airport and Arrival_Airport relationships).
- **Technical Implementation Note:** Can be enforced using a CHECK constraint during table creation or an INSERT/UPDATE Trigger.

Business rules

2. Reservation Capacity Constraint (SEAT and RESERVATION)

- **Rule:** A reservation can only be confirmed if the number of seats requested does not exceed the total available seat capacity for that specific leg instance (LEG_INSTANCE). Furthermore, the same physical seat cannot be assigned to multiple unique reservations for the same flight instance.
- **Applies to:** RESERVATION, SEAT, and LEG_INSTANCE entities.
- **Technical Implementation Note:** This should be controlled by a Trigger that is fired when a record is added to the RESERVATION table.

Business rules

3. SEAT CAPACITY RULE: (LEG_INSTANCE and RESERVATION)

- **Rule:** The number of available seats (No_of_available_seats) in a flight leg instance (LEG_INSTANCE) must always be greater than or equal to the total number of reservations for that flight (counted using the RESERVATION table). **Applies To:** LEG_INSTANCE entity,No_of_available_seats attribute ,RESERVATION relationship and COUNT attribute.

- **Technical Implementation Note:** When a reservation is added (INSERT to RESERVATION), a Trigger is used that checks this rule and updates the No_of_available_seats value.

Business rules

4. **AIRCRAFT LANDING COMPABILITY RULE: (CAN_LAND Relationship)**

- **Rule:** An aircraft (AIRPLANE) to which an aircraft type (AIRPLANE_TYPE) is assigned (TYPE relation) cannot take off or land from an airport (AIRPORT) that is not allowed to land for that aircraft type (i.e. it cannot use an AIRPORT that is not on the M side in the CAN_LAND relation).
- **Applies To:** AIRPLANE_TYPE, AIRPORT and CAN_LAND relationship between them.

Business rules

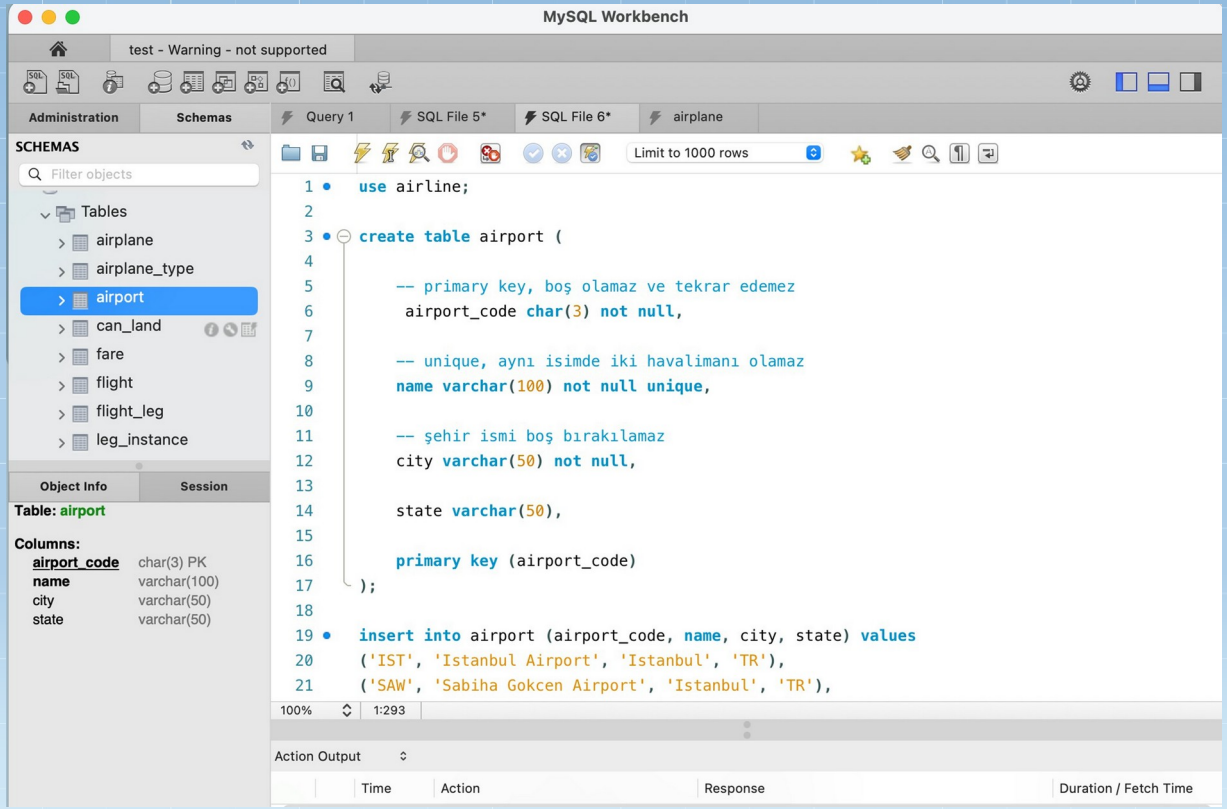
5. **POSITIVE FARE VALUE RULE: (FARE)**

- **Rule:** The amount (Amount) of a fee (FARE) cannot be a negative value. Additionally, every fee must have a restriction (Restrictions).
- **Applies To:** FARE entity and Amount,Restrictions attributes
- **Technical Implementation Note:** Implemented in DDL with CHECK(Amount >= 0) on the Amount column and NOT NULL constraint on the Restrictions column.

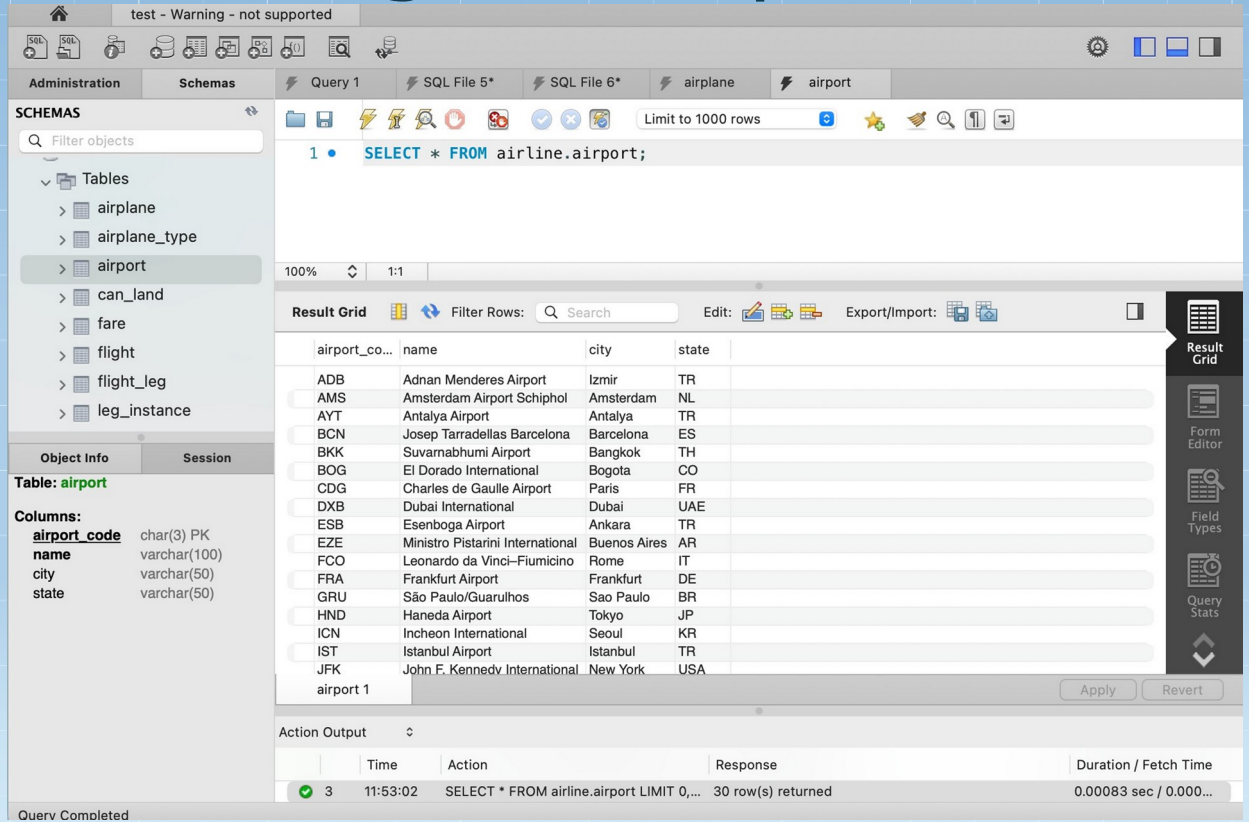
normalization

- First Normal Form (1NF) is satisfied, as all attributes contain atomic (indivisible) values, and there are no repeating groups within any table.
- Second Normal Form (2NF) is satisfied because all non-key attributes in every table with a composite key are fully dependent on the entire primary key.
- Third Normal Form (3NF) is satisfied because there are no transitive dependencies; meaning, all non-key attributes are directly dependent on the primary key, and no non- key attribute is dependent on another non-key attribute.

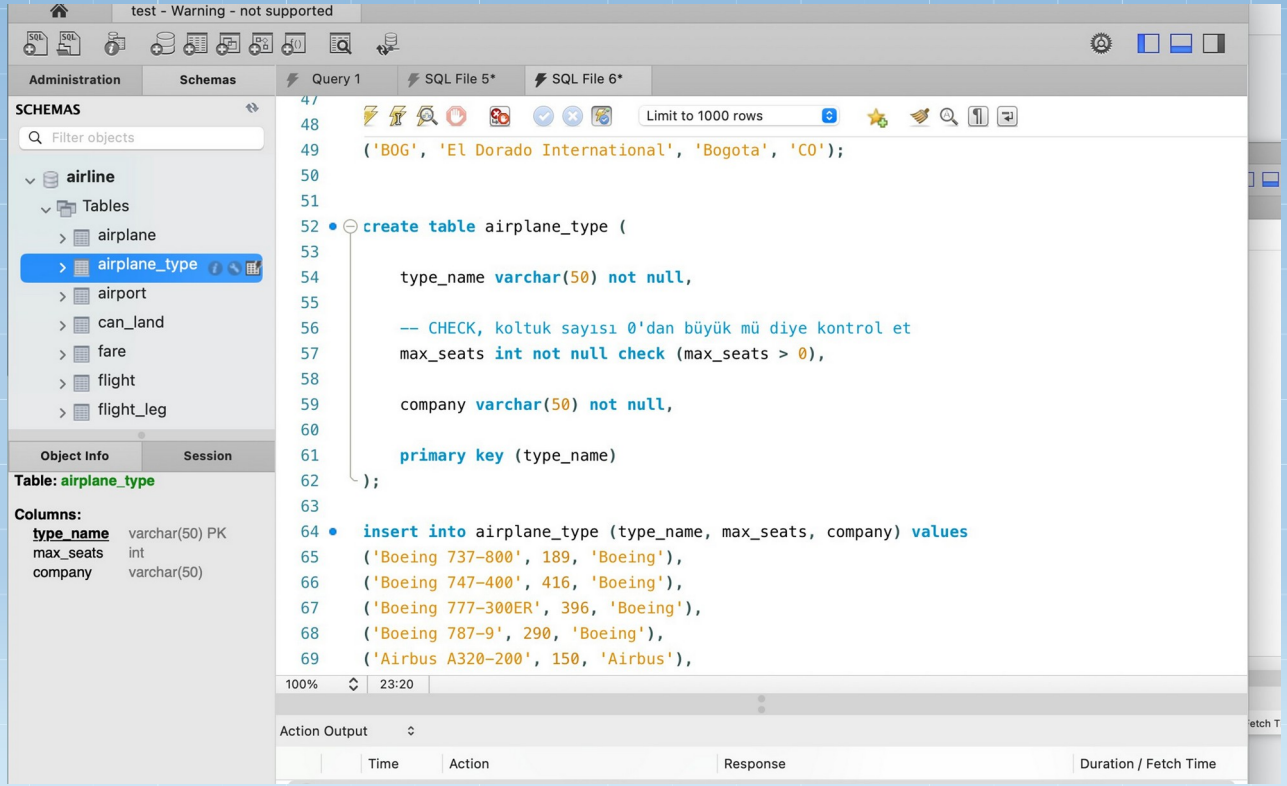
Creating the Airport Table



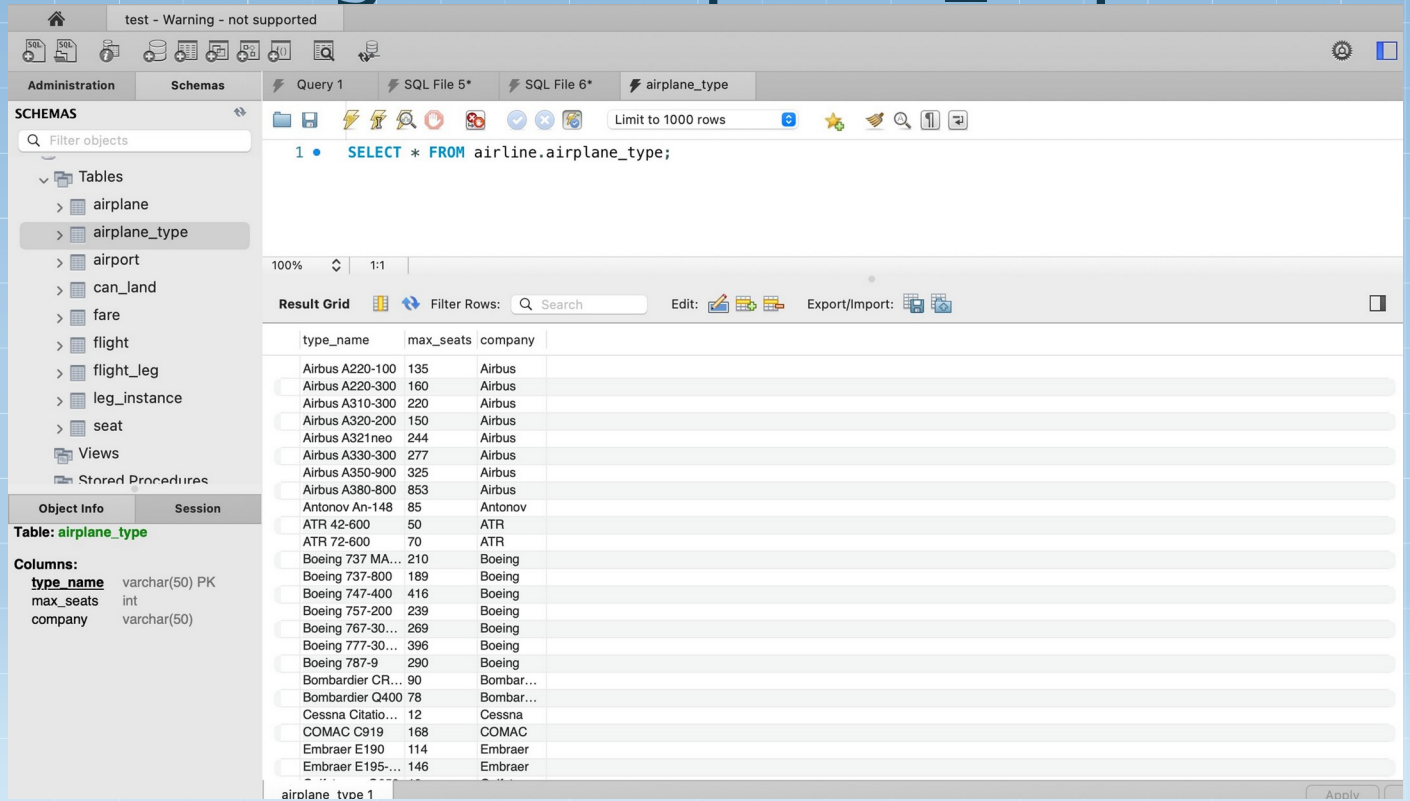
Creating the Airport Table



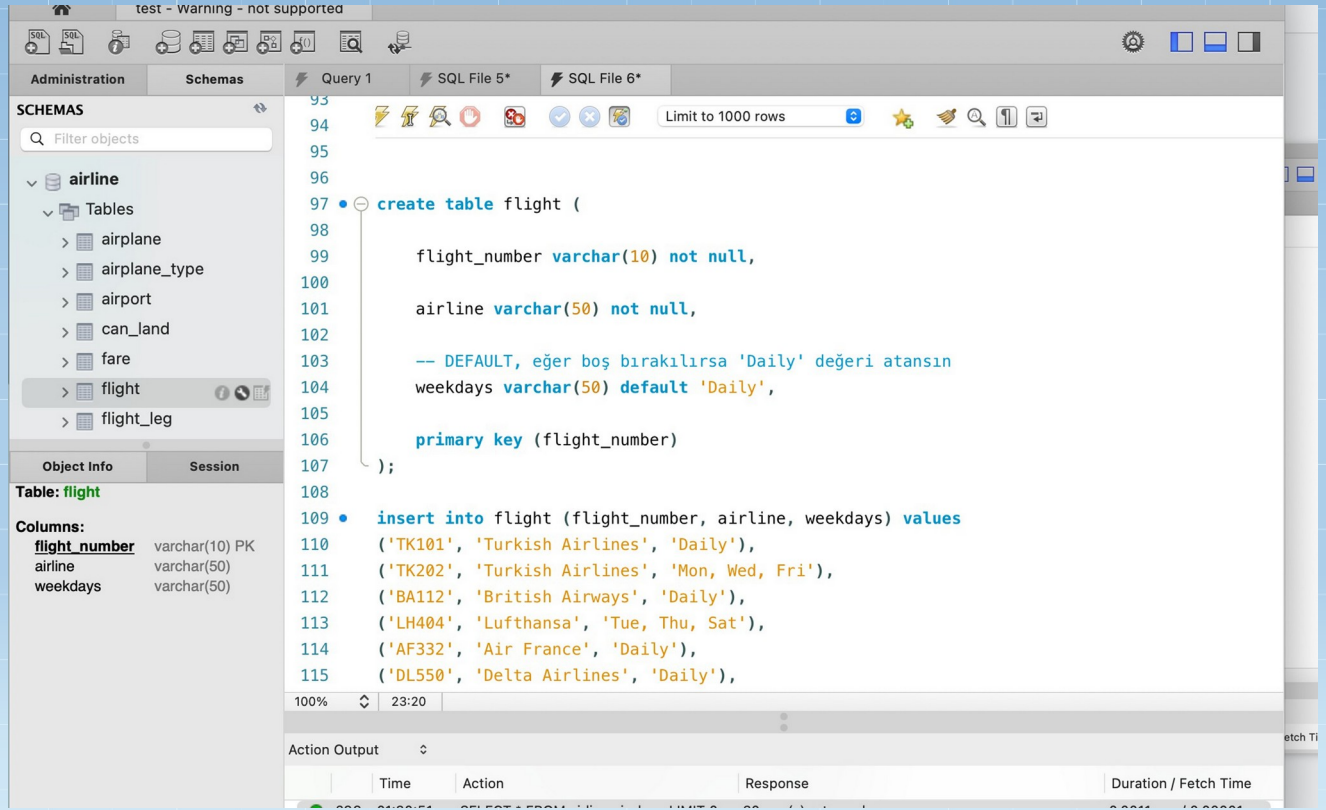
Creating the Airplane_TYpe Table



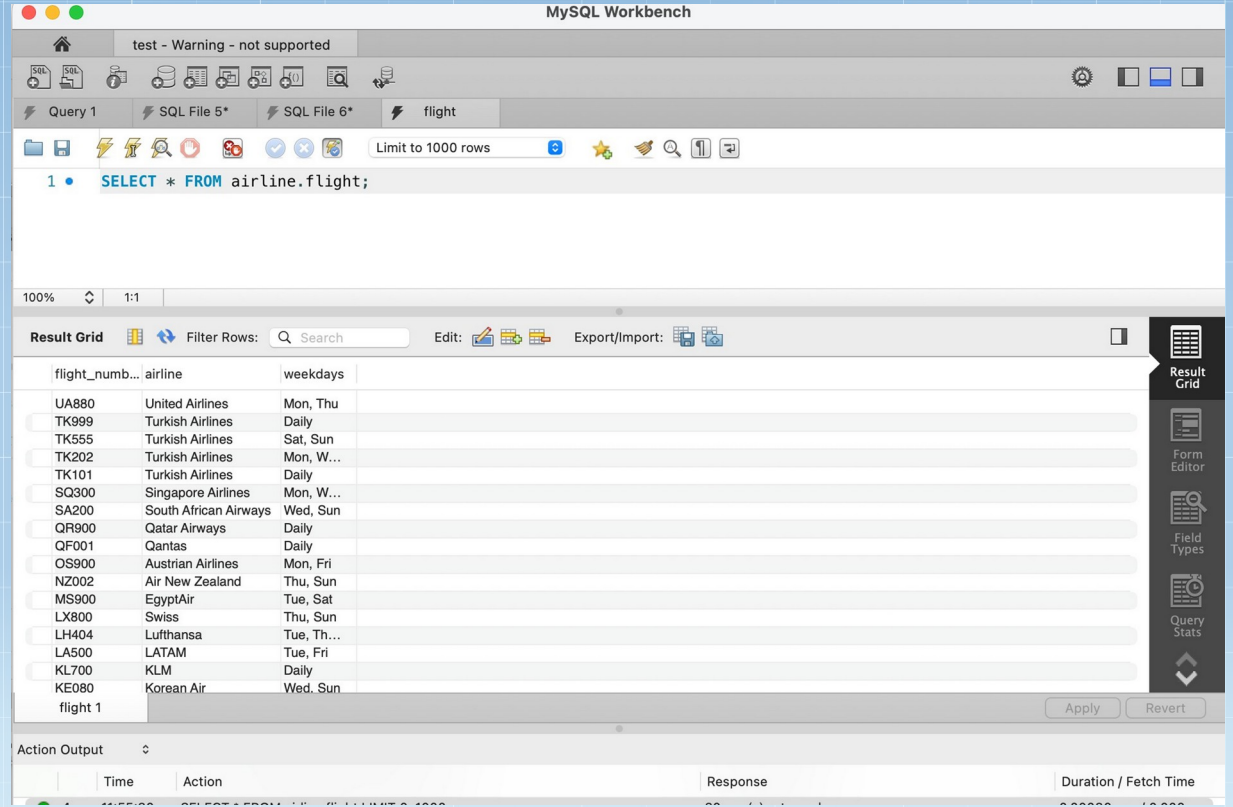
Creating the Airplane_TYpe Table



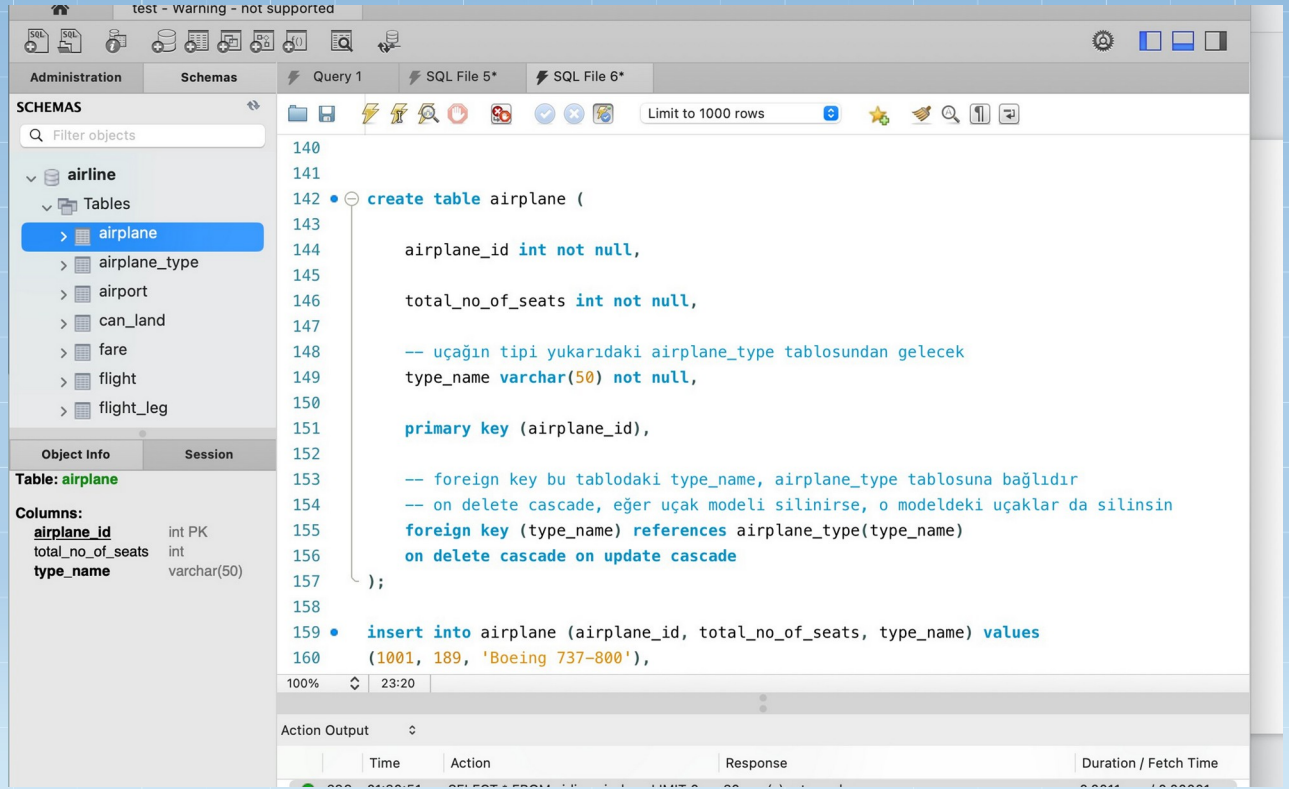
Creating the flight Table



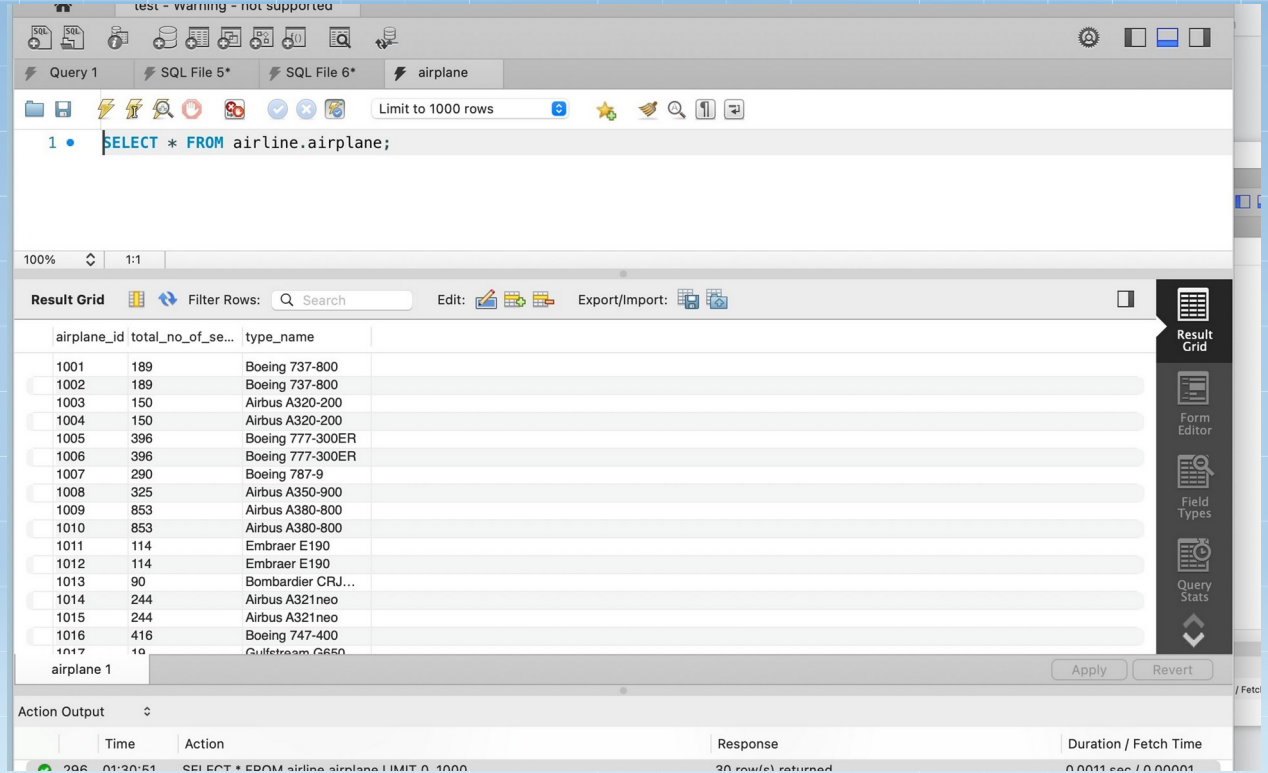
Creating the flight Table



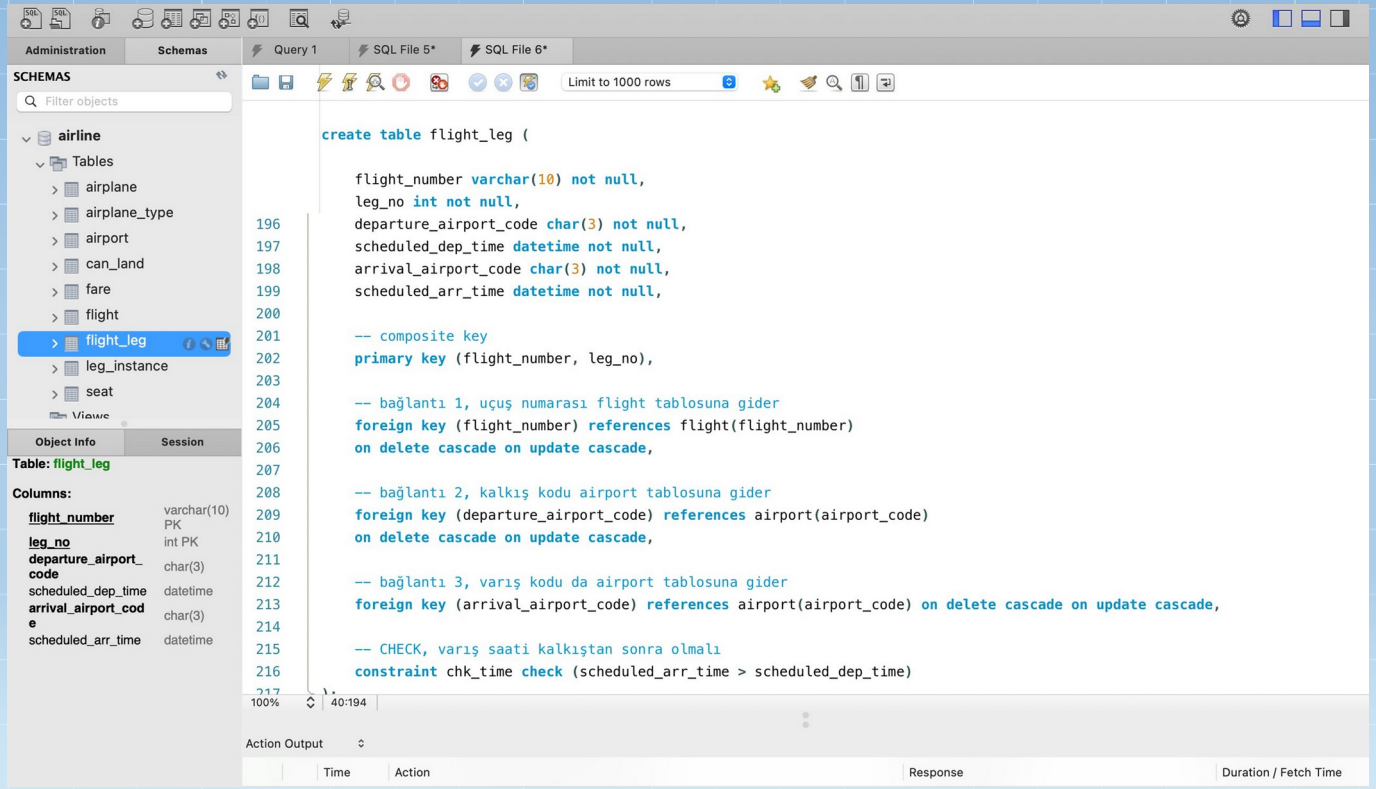
Creating the airplane Table



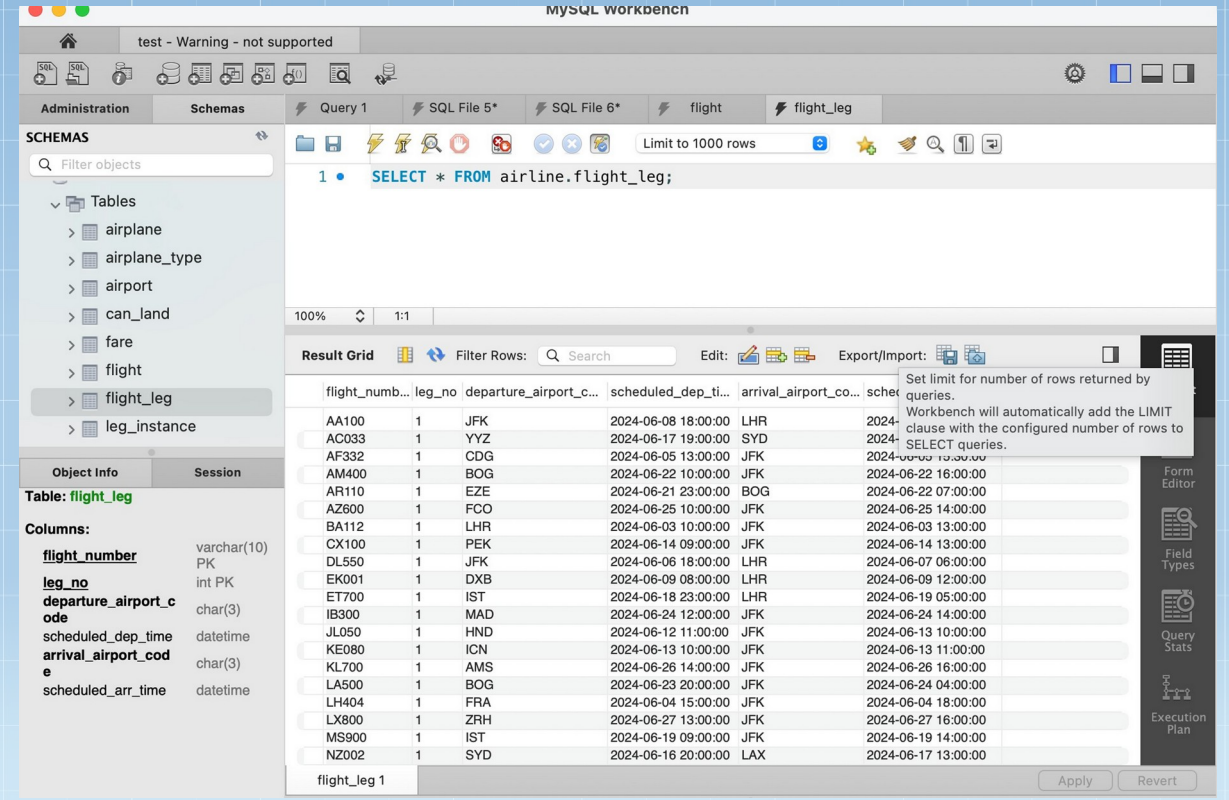
Creating the airplane Table



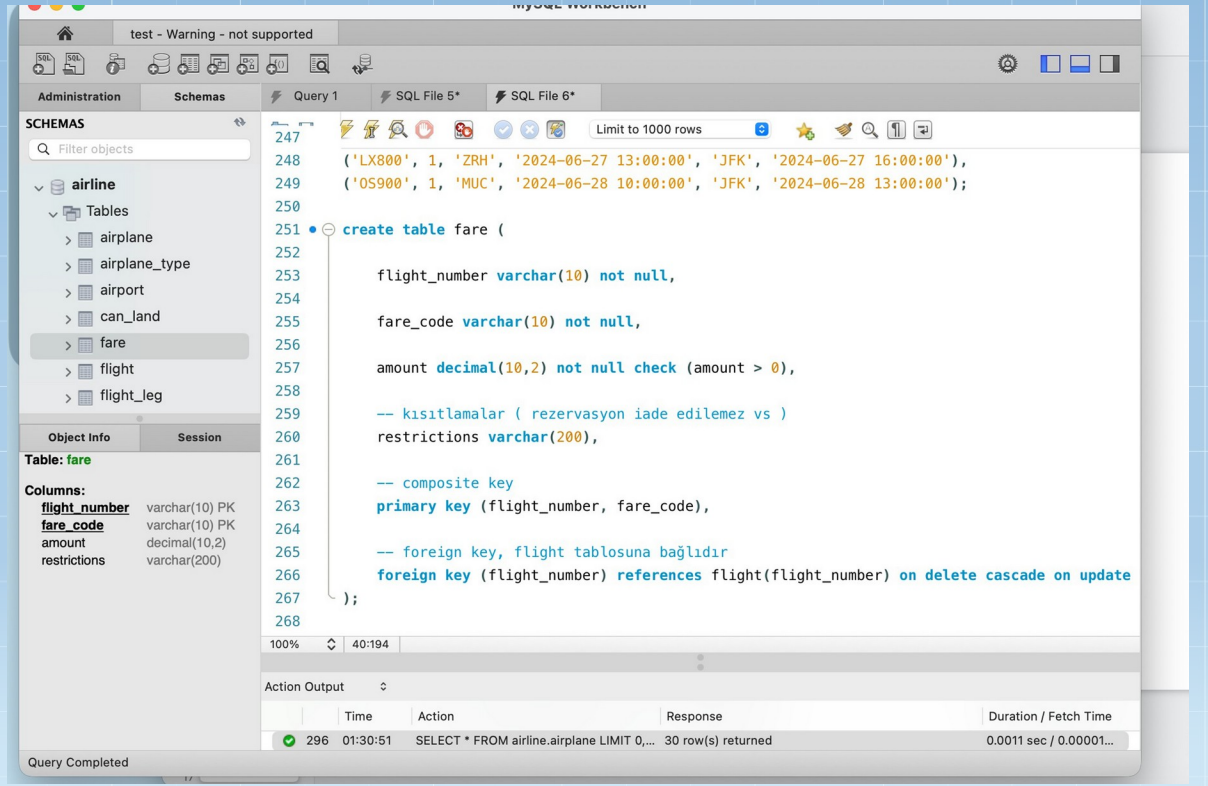
Creating the flight_leg Table



Creating the flight_leg Table



Creating the fare Table



Creating the fare Table

```
1 •  SELECT * FROM airline.fare;
2
```

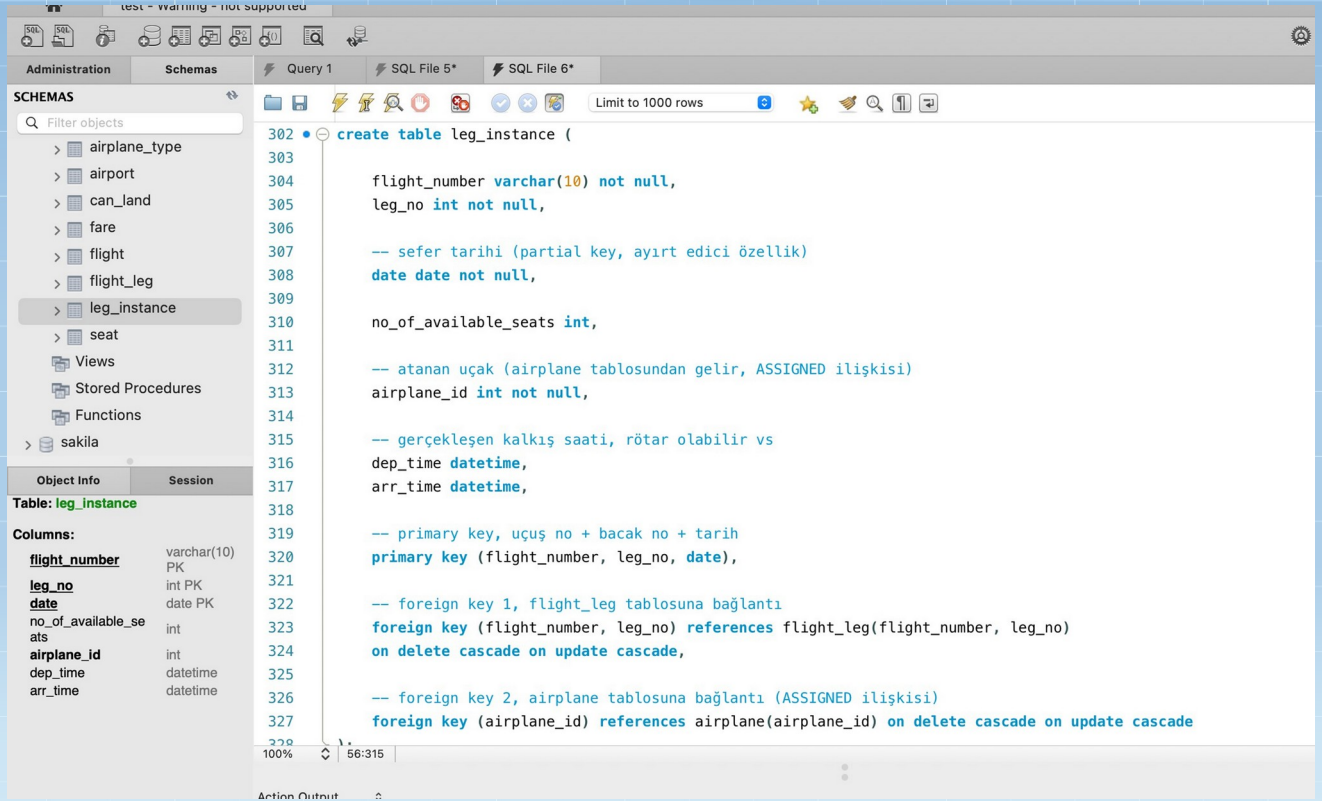
100% 28:1

Result Grid | Filter Rows: Search | Edit: | Export/Impo

flight_number	fare_code	amount	restrictions
AA100	ECO	650.00	No meals
AC033	ECO	600.00	Standard
AF332	ECO	520.00	Non-refundable
AM400	ECO	450.00	Light
AR110	ECO	900.00	Standard
AZ600	ECO	480.00	Standard
BA112	ECO	600.00	Standard
BA112	FIRST	2500.00	All inclusive
CX100	ECO	880.00	Standard
DL550	ECO	700.00	1 Bag included
EK001	BUS	2000.00	Lounge access
EK001	ECO	800.00	Standard
ET700	ECO	400.00	2 Bags included
IB300	ECO	500.00	Standard
JL050	BUS	2200.00	Refundable
KE080	ECO	850.00	Standard
KL700	BUS	1100.00	Lounge access
LA500	BUS	1300.00	Refundable

fare 1

Creating the leg_ınstance Table



Creating the leg_instance Table

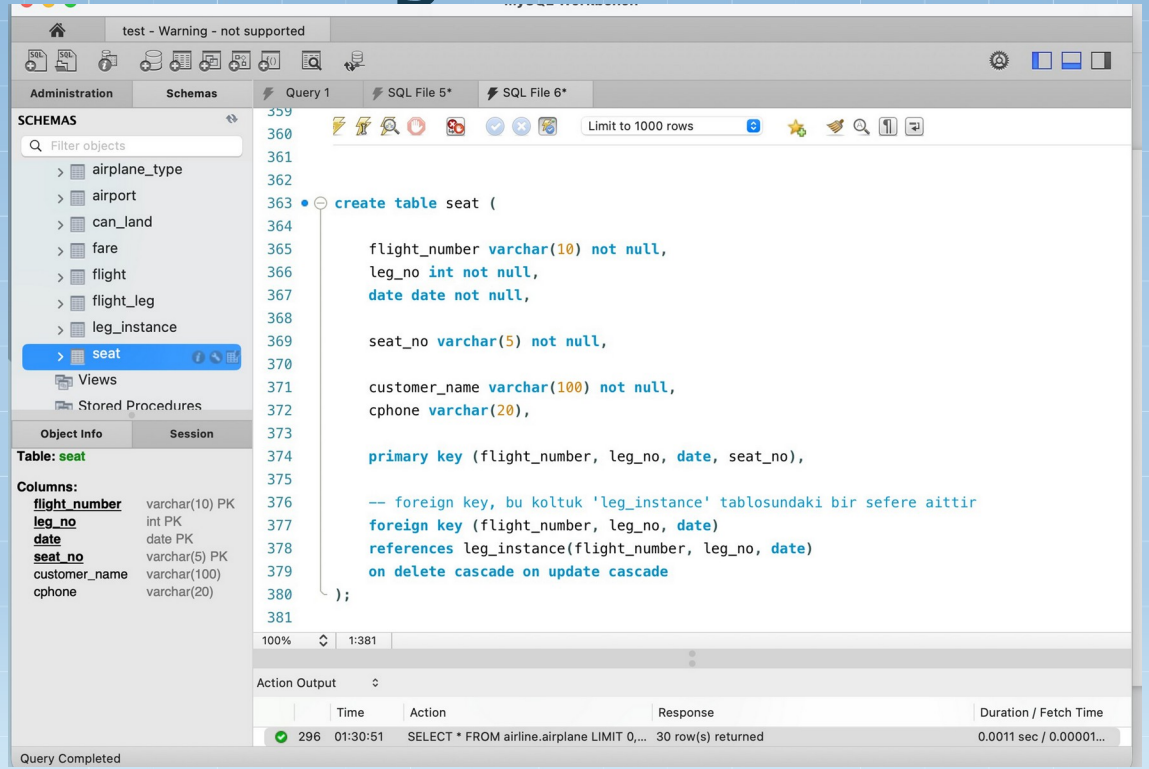
```
1 •  SELECT * FROM airline.leg_instance;
2
```

100% 36:1

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_number	leg_no	date	no_of_available_seats	airplane_id	dep_time	arr_time
AA100	1	2024-06-08	25	1004	2024-06-08 18:00:00	2024-06-09 06:00:00
AC033	1	2024-06-17	90	1012	2024-06-17 19:00:00	2024-06-19 06:00:00
AF332	1	2024-06-05	150	1008	2024-06-05 13:00:00	2024-06-05 15:30:00
AM400	1	2024-06-22	9	1021	2024-06-22 10:00:00	2024-06-22 16:00:00
AR110	1	2024-06-21	11	1030	2024-06-21 23:00:00	2024-06-22 07:00:00
AZ600	1	2024-06-25	44	1026	2024-06-25 10:00:00	2024-06-25 14:00:00
BA112	1	2024-06-03	100	1005	2024-06-03 10:00:00	2024-06-03 13:00:00
CX100	1	2024-06-14	15	1024	2024-06-14 09:00:00	2024-06-14 13:00:00
DL550	1	5.MVC_Core_DataTransfer (1).pptx		1006	2024-06-06 18:00:00	2024-06-07 06:00:00
EK001	1	2024-06-09	200	1009	2024-06-09 08:00:00	2024-06-09 12:00:00
ET700	1	2024-06-18	45	1015	2024-06-18 23:00:00	2024-06-19 05:00:00
IB300	1	2024-06-24	66	1025	2024-06-24 12:00:00	2024-06-24 14:00:00
JL050	1	2024-06-12	60	1016	2024-06-12 11:00:00	2024-06-13 10:00:00
KE080	1	2024-06-13	80	1023	2024-06-13 10:00:00	2024-06-13 11:00:00
KL700	1	2024-06-26	10	1027	2024-06-26 14:00:00	2024-06-26 16:00:00
LA500	1	2024-06-23	88	1022	2024-06-23 20:00:00	2024-06-24 04:00:00
LH404	1	2024-06-04	5	1007	2024-06-04 15:10:00	2024-06-04 18:15:00
LX800	1	2024-06-27	5	1028	2024-06-27 13:00:00	2024-06-27 16:00:00

Creating the seat Table



Creating the seat Table

```
1 •  SELECT * FROM airline.seat;
2
```

100% 28:1

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_numb...	leg_no	date	seat_no	customer_name	cphone
AA100	1	2024-06-08	15F	Tom Hanks	555-0112
AC033	1	2024-06-17	9F	Ryan Reynolds	555-0121
AF332	1	2024-06-05	4D	Jean Dupont	555-0109
AM400	1	2024-06-22	20E	Frida Kahlo	555-0126
AR110	1	2024-06-21	18D	Lionel Messi	555-0125
AZ600	1	2024-06-25	35B	Monica Bellucci	555-0129
BA112	1	2024-06-03	12A	John Smith	555-0106
BA112	1	2024-06-03	12B	Jane Doe	555-0107
CX100	1	2024-06-14	6C	Bruce Lee	555-0118
DL550	1	2024-06-06	1A	Michael Jordan	555-0110
EK001	1	2024-06-09	1A	Ali Veli	555-0113
ET700	1	2024-06-18	11A	Abebe Bikila	555-0122
IB300	1	2024-06-24	30A	Rafael Nadal	555-0128
JL050	1	2024-06-12	44H	Tanaka Sato	555-0116
KE080	1	2024-06-13	5A	Kim Min	555-0117
KL700	1	2024-06-26	40C	Vincent van Gogh	555-0130
LA500	1	2024-06-23	25F	Pedro Pascal	555-0127
LH404	1	2024-06-04	3C	Hans Müller	555-0108

Creating the fare Table

```
1 •  SELECT * FROM airline.fare;
2
```

100% 28:1

Result Grid | Filter Rows: Search | Edit: | Export/Impo

flight_number	fare_code	amount	restrictions
AA100	ECO	650.00	No meals
AC033	ECO	600.00	Standard
AF332	ECO	520.00	Non-refundable
AM400	ECO	450.00	Light
AR110	ECO	900.00	Standard
AZ600	ECO	480.00	Standard
BA112	ECO	600.00	Standard
BA112	FIRST	2500.00	All inclusive
CX100	ECO	880.00	Standard
DL550	ECO	700.00	1 Bag included
EK001	BUS	2000.00	Lounge access
EK001	ECO	800.00	Standard
ET700	ECO	400.00	2 Bags included
IB300	ECO	500.00	Standard
JL050	BUS	2200.00	Refundable
KE080	ECO	850.00	Standard
KL700	BUS	1100.00	Lounge access
LA500	BUS	1300.00	Refundable

fare 1

Creating the leg_instance Table

```
1 •  SELECT * FROM airline.leg_instance;
2
```

100% 36:1

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_number	leg_no	date	no_of_available_seats	airplane_id	dep_time	arr_time
AA100	1	2024-06-08	25	1004	2024-06-08 18:00:00	2024-06-09 06:00:00
AC033	1	2024-06-17	90	1012	2024-06-17 19:00:00	2024-06-19 06:00:00
AF332	1	2024-06-05	150	1008	2024-06-05 13:00:00	2024-06-05 15:30:00
AM400	1	2024-06-22	9	1021	2024-06-22 10:00:00	2024-06-22 16:00:00
AR110	1	2024-06-21	11	1030	2024-06-21 23:00:00	2024-06-22 07:00:00
AZ600	1	2024-06-25	44	1026	2024-06-25 10:00:00	2024-06-25 14:00:00
BA112	1	2024-06-03	100	1005	2024-06-03 10:00:00	2024-06-03 13:00:00
CX100	1	2024-06-14	15	1024	2024-06-14 09:00:00	2024-06-14 13:00:00
DL550	1	5.MVC_Core_DataTransfer (1).pptx		1006	2024-06-06 18:00:00	2024-06-07 06:00:00
EK001	1	2024-06-09	200	1009	2024-06-09 08:00:00	2024-06-09 12:00:00
ET700	1	2024-06-18	45	1015	2024-06-18 23:00:00	2024-06-19 05:00:00
IB300	1	2024-06-24	66	1025	2024-06-24 12:00:00	2024-06-24 14:00:00
JL050	1	2024-06-12	60	1016	2024-06-12 11:00:00	2024-06-13 10:00:00
KE080	1	2024-06-13	80	1023	2024-06-13 10:00:00	2024-06-13 11:00:00
KL700	1	2024-06-26	10	1027	2024-06-26 14:00:00	2024-06-26 16:00:00
LA500	1	2024-06-23	88	1022	2024-06-23 20:00:00	2024-06-24 04:00:00
LH404	1	2024-06-04	5	1007	2024-06-04 15:10:00	2024-06-04 18:15:00
LX800	1	2024-06-27	5	1028	2024-06-27 13:00:00	2024-06-27 16:00:00

Creating the seat Table

```
1 •  SELECT * FROM airline.seat;
2
```

100% 28:1

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_numb...	leg_no	date	seat_no	customer_name	cphone
AA100	1	2024-06-08	15F	Tom Hanks	555-0112
AC033	1	2024-06-17	9F	Ryan Reynolds	555-0121
AF332	1	2024-06-05	4D	Jean Dupont	555-0109
AM400	1	2024-06-22	20E	Frida Kahlo	555-0126
AR110	1	2024-06-21	18D	Lionel Messi	555-0125
AZ600	1	2024-06-25	35B	Monica Bellucci	555-0129
BA112	1	2024-06-03	12A	John Smith	555-0106
BA112	1	2024-06-03	12B	Jane Doe	555-0107
CX100	1	2024-06-14	6C	Bruce Lee	555-0118
DL550	1	2024-06-06	1A	Michael Jordan	555-0110
EK001	1	2024-06-09	1A	Ali Veli	555-0113
ET700	1	2024-06-18	11A	Abebe Bikila	555-0122
IB300	1	2024-06-24	30A	Rafael Nadal	555-0128
JL050	1	2024-06-12	44H	Tanaka Sato	555-0116
KE080	1	2024-06-13	5A	Kim Min	555-0117
KL700	1	2024-06-26	40C	Vincent van Gogh	555-0130
LA500	1	2024-06-23	25F	Pedro Pascal	555-0127
LH404	1	2024-06-04	3C	Hans Müller	555-0108

Creating the reservation Table

```
CREATE TABLE reservation (
    reservation_id INT AUTO_INCREMENT PRIMARY KEY,
    flight_number VARCHAR(10),
    leg_no INT,
    date DATE,
    customer_name VARCHAR(50),
    customer_phone VARCHAR(15),
    seat_no VARCHAR(5),
    FOREIGN KEY (flight_number, leg_no, date)
        REFERENCES leg_instance(flight_number, leg_no, date)
);
```

Reporting and Analytics Queries

In this part of the project, reporting and analytics queries were implemented to analyze the database in line with the business logic.

INSERT 3 tuples to different tables

- Three records were inserted into different tables to populate the database with initial data.

```sql
-- insert 3 tuples

INSERT INTO airport (airport_code, name, city, state)
VALUES ('VCE', 'Venice Marco Polo Airport', 'Venice', 'IT');
SELECT * FROM airline.airport;

INSERT INTO airplane_type (type_name, max_seats, company)
VALUES ('Boeing 737-900ER', 215, 'Boeing');
SELECT * FROM airline.airplane_type;

INSERT INTO flight (flight_number, airline, weekdays)
VALUES ('TK888', 'Turkish Airlines', 'Mon, Wed, Fri');
SELECT * FROM airline.flight;
```

INSERT 3 tuples to different tables

	MXP	Malpensa Airport	Milan	IT
	ORD	O'Hare International	Chicago	USA
	PEK	Beijing Capital International	Beijing	CN
	SAW	Sabiha Gokcen Airport	Istanbul	TR
	SIN	Singapore Changi Airport	Singapore	SG
	SYD	Kingsford Smith Airport	Sydney	AU
	VCE	Venice Marco Polo Airport	Venice	IT
	YYZ	Toronto Pearson Internatio…	Toronto	CA
	ZRH	Zurich Airport	Zurich	CH
	NULL	NULL	NULL	NULL

	ATR 72-600	70	ATR
	Boeing 737 MAX 8	210	Boeing
	Boeing 737-800	189	Boeing
	Boeing 737-900ER	215	Boeing
	Boeing 747-400	416	Boeing
	Boeing 757-200	239	Boeing
	Boeing 767-300ER	269	Boeing
	Boeing 777-300ER	396	Boeing
	Boeing 787-9	290	Boeing
	Bombardier CRJ900	90	Bombar…
	Bombardier Q400	78	Bombar

	SA200	South African Airways	Wed, Sun
	SQ300	Singapore Airlines	Mon, W…
	TK101	Turkish Airlines	Daily
	TK202	Turkish Airlines	Mon, W…
	TK555	Turkish Airlines	Sat, Sun
	TK777	Turkish Airlines	Tue, Th…
	TK888	Turkish Airlines	Mon, W…
	TK999	Turkish Airlines	Daily
	UA880	United Airlines	Mon, Thu
	NULL	NULL	NULL

INSERT 5 tuples to same table

- Five records were added to the same table to test bulk data insertion.

```
-- 5 tuples to same table
INSERT INTO airplane_type (type_name, max_seats, company) VALUES
('Narrow Body', 180, 'Boeing'),
('White Body', 350, 'Airbus'),
('Reginal Jet', 90 , 'Embraer'),
('Turboprop', 70, 'ATR'),
('Medium Haul', 220, 'Airbus');
```

INSERT 5 tuples to same table

Ilyushin Il-96	262	Ilyushin
McDonnell Dougla…	172	McDon…
Medium Haul	220	Airbus
Mitsubishi SpaceJ…	88	Mitsubishi
Narrow Body	180	Boeing
Reginal Jet	90	Embraer
Sukhoi Superjet 100	98	Sukhoi
Tupolev Tu-204	210	Tupolev
Turboprop	70	ATR
White Body	350	Airbus
NULL	NULL	NULL

UPDATE 3 tuples in different tables

- Existing records in three different tables were updated to reflect changes in business data.

```
-- update 3 tuples
UPDATE fare SET amount = 550.00 WHERE flight_number = 'TK101' AND fare_code = 'ECO';
UPDATE airport SET name = 'Istanbul Grand Airport' WHERE airport_code = 'IST';
UPDATE airplane_type SET max_seats = 200 WHERE type_name = 'Boeing 737-800';
```

ATR 72-600	70	ATR
Boeing 737 MAX 8	210	Boeing
Boeing 737-600	132	Boeing
Boeing 737-800	200	Boeing
Boeing 737-900ER	215	Boeing
Boeing 747-400	416	Boeing
Boeing 757-200	239	Boeing
Boeing 767-300ER	269	Boeing

SA200	ECO	700.00	Standard
SQ300	ECO	900.00	2 Bags included
TK101	BUS	1200.00	Refundable
TK101	ECO	550.00	Non-refundable
TK202	ECO	450.00	No baggage
UA880	BUS	1500.00	Refundable
NULL	NULL	NULL	NULL

HND	Haneda Airport	Tokyo	JP
ICN	Incheon International	Seoul	KR
IST	Istanbul Grand Airport	Istanbul	TR
JFK	John F. Kennedy Internatio…	New York	USA
LAX	Los Angeles International	Los Angeles	USA
LHR	Heathrow Airport	London	UK
MAD	Adolfo Suárez Madrid–Bar…	Madrid	ES
MEL	Melbourne Airport	Melbourne	AU

DELETE 3 tuples in different tables

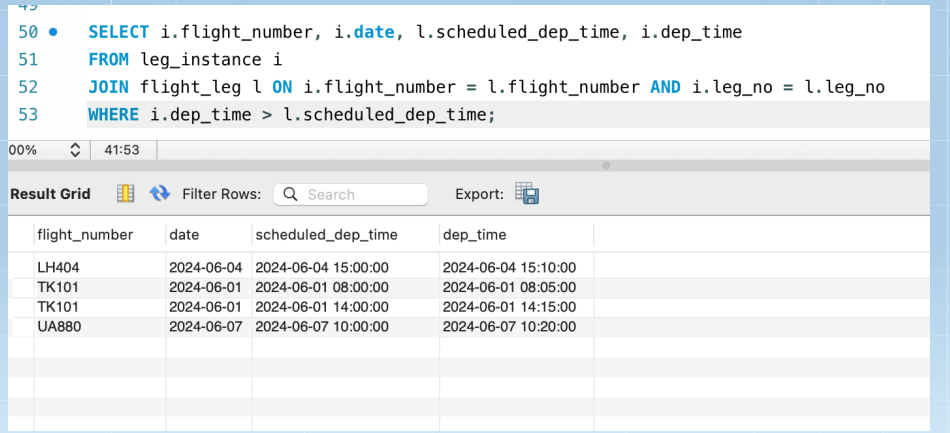
- Selected records were removed from different tables to demonstrate data deletion operations.

```sql
-- delete 3 tuples
DELETE FROM seat WHERE customer_name = 'Ahmet Demir';
DELETE FROM fare WHERE flight_number = 'TK202';
DELETE FROM can_land WHERE type_name = 'Gulfstream G650';
```

✓	511	18:00:25	DELETE FROM seat WHERE customer_name = 'Ahmet Demir'	1 row(s) affected
✓	512	18:00:25	DELETE FROM fare WHERE flight_number = 'TK202'	1 row(s) affected
✓	513	18:00:25	DELETE FROM can_land WHERE type_name = 'Gulfstream G6...	3 row(s) affected

Create 3 queries with WHERE clause and comparison operators

- Queries using comparison operators were created to filter records based on specific conditions.



```
50  SELECT i.flight_number, i.date, l.scheduled_dep_time, i.dep_time
51  FROM leg_instance i
52  JOIN flight_leg l ON i.flight_number = l.flight_number AND i.leg_no = l.leg_no
53  WHERE i.dep_time > l.scheduled_dep_time;
```

100% 41:53

Result Grid Filter Rows: Search Export:

flight_number	date	scheduled_dep_time	dep_time
LH404	2024-06-04	2024-06-04 15:00:00	2024-06-04 15:10:00
TK101	2024-06-01	2024-06-01 08:00:00	2024-06-01 08:05:00
TK101	2024-06-01	2024-06-01 14:00:00	2024-06-01 14:15:00
UA880	2024-06-07	2024-06-07 10:00:00	2024-06-07 10:20:00

Create 3 queries with WHERE clause and comparison operators

```
41  -- create 3 queries with WHERE comparison operators
42  SELECT type_name, max_seats, company
43  FROM airplane_type
44  WHERE max_seats > 300;
45
```

100% 1:40

Result Grid | Filter Rows: Search | Edit: | Export/Import:

type_name	max_seats	company
Airbus A340-300	375	Airbus
Airbus A350-900	325	Airbus
Airbus A380-800	853	Airbus
Boeing 747-400	416	Boeing
Boeing 777-300ER	396	Boeing
White Body	350	Airbus
NULL	NULL	NULL

```
46  SELECT flight_number, fare_code, amount, restrictions
47  FROM fare
48  WHERE amount >= 1000.00;
```

100% 1:50

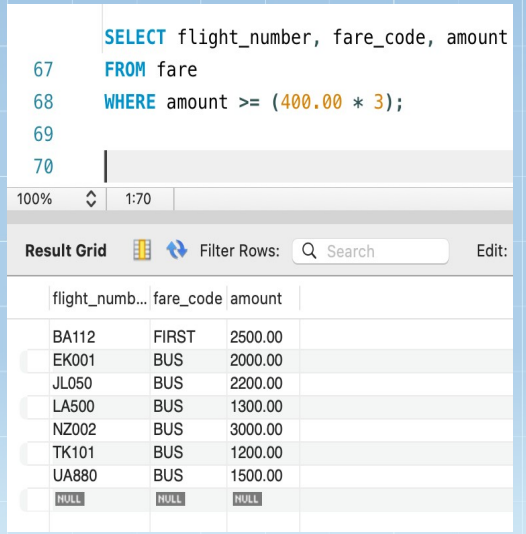
Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_numb...	fare_code	amount	restrictions
BA112	FIRST	2500.00	All inclusive
EK001	BUS	2000.00	Lounge access
JL050	BUS	2200.00	Refundable
KL700	BUS	1100.00	Lounge access
LA500	BUS	1300.00	Refundable
NZ002	BUS	3000.00	Refundable
QF001	ECO	1100.00	Non-refundable
TK101	BUS	1200.00	Refundable
UA880	BUS	1500.00	Refundable

fare 15

Create 3 queries with WHERE clause and arithmetic operators

- Arithmetic operations were applied in the WHERE clause to filter calculated values.

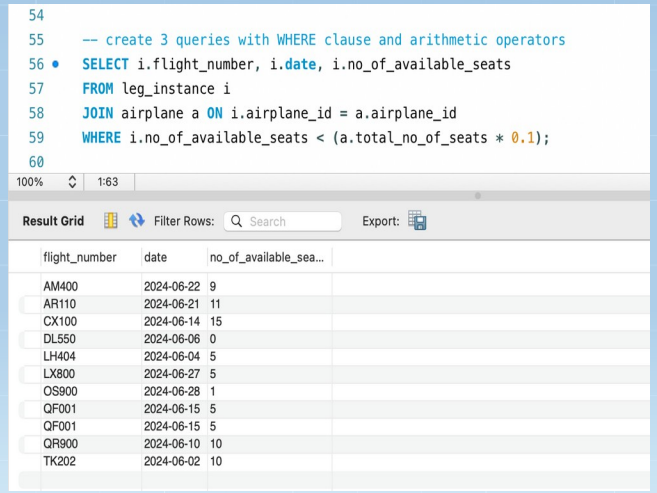


```
SELECT flight_number, fare_code, amount
FROM fare
WHERE amount >= (400.00 * 3);
```

100% 1:70

Result Grid Filter Rows: Search Edit:

flight_numb...	fare_code	amount
BA112	FIRST	2500.00
EK001	BUS	2000.00
JL050	BUS	2200.00
LA500	BUS	1300.00
NZ002	BUS	3000.00
TK101	BUS	1200.00
UA880	BUS	1500.00
NULL	NULL	NULL

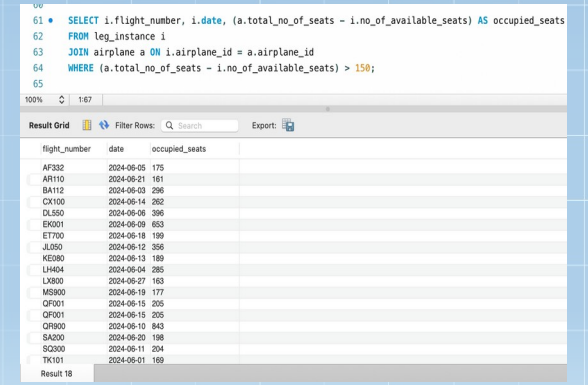


```
-- create 3 queries with WHERE clause and arithmetic operators
SELECT i.flight_number, i.date, i.no_of_available_seats
FROM leg_instance i
JOIN airplane a ON i.airplane_id = a.airplane_id
WHERE i.no_of_available_seats < (a.total_no_of_seats * 0.1);
```

100% 1:63

Result Grid Filter Rows: Search Export:

flight_number	date	no_of_available_sea...
AM400	2024-06-22	9
AR110	2024-06-21	11
CX100	2024-06-14	15
DL550	2024-06-06	0
LH404	2024-06-04	5
LX800	2024-06-27	5
OS900	2024-06-28	1
QF001	2024-06-15	5
QF001	2024-06-15	5
QR900	2024-06-10	10
TK202	2024-06-02	10



```
SELECT i.flight_number, i.date, (a.total_no_of_seats - i.no_of_available_seats) AS occupied_seats
FROM leg_instance i
JOIN airplane a ON i.airplane_id = a.airplane_id
WHERE (a.total_no_of_seats - i.no_of_available_seats) > 150;
```

100% 1:67

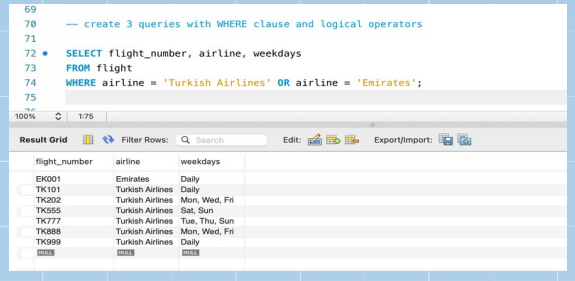
Result Grid Filter Rows: Search Export:

flight_number	date	occupied_seats
AF332	2024-06-05	175
AR110	2024-06-21	161
BA112	2024-06-03	296
CX100	2024-06-14	262
DL550	2024-06-06	396
EK001	2024-06-09	653
ET700	2024-06-18	199
JL050	2024-06-12	356
KE080	2024-06-13	189
LH404	2024-06-04	285
LX800	2024-06-27	163
MS900	2024-06-19	177
QF001	2024-06-15	205
QF001	2024-06-15	205
QR900	2024-06-10	843
SA200	2024-06-20	198
SQ300	2024-06-11	204
TK101	2024-06-01	169

Result 18

Create 3 queries with WHERE clause and logical operators

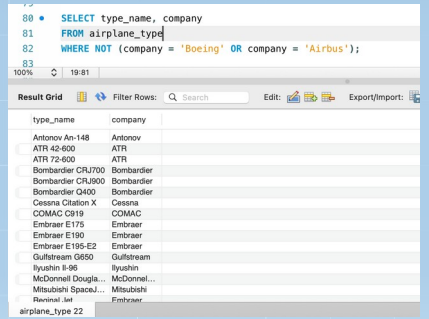
- Logical operators were used to combine multiple conditions in query filtering.



```
-- create 3 queries with WHERE clause and logical operators

SELECT flight_number, airline, weekdays
FROM flight
WHERE airline = 'Turkish Airlines' OR airline = 'Emirates';
```

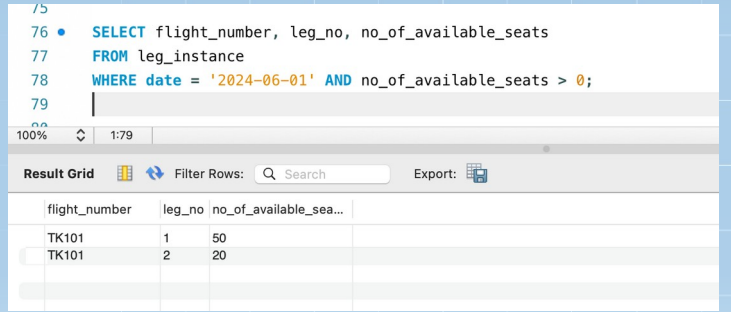
flight_number	airline	weekdays
EK001	Emirates	Daily
TK101	Turkish Airlines	Daily
TK202	Turkish Airlines	Mon, Wed, Fri
TK555	Turkish Airlines	Sat, Sun
TK777	Turkish Airlines	Tue, Thu, Sun
TK888	Turkish Airlines	Mon, Wed, Fri
TK999	Turkish Airlines	Daily



```
SELECT type_name, company
FROM airplane_type
WHERE NOT (company = 'Boeing' OR company = 'Airbus');
```

type_name	company
Antonov An-148	Antonov
ATR 42-600	ATR
ATR 72-600	ATR
Bombardier CRJ700	Bombardier
Bombardier CRJ900	Bombardier
Bombardier Q400	Bombardier
Cessna Citation X	Cessna
COMAC C919	COMAC
Embraer E175	Embraer
Embraer E190	Embraer
Embraer E195-E2	Embraer
Gulfstream G650	Gulfstream
Ilyushin Il-96	Ilyushin
McDonnell Dougla...	McDonnel...
Mitsubishi SpaceJ...	Mitsubishi
Regional Jet	Embraer

airplane_type 22



```
SELECT flight_number, leg_no, no_of_available_seats
FROM leg_instance
WHERE date = '2024-06-01' AND no_of_available_seats > 0;
```

flight_number	leg_no	no_of_available_sea...
TK101	1	50
TK101	2	20

Create 3 queries with special operators

- Special SQL operators were used to filter data using ranges, null values, pattern matching, and subqueries.

```sql
-- create 3 queries with special operators

SELECT flight_number, fare_code, amount
FROM fare
WHERE amount BETWEEN 500.00 AND 1000.00;
```

100% 41:88

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_numb...	fare_code	amount
AA100	ECO	650.00
AC033	ECO	600.00
AF332	ECO	520.00
AR110	ECO	900.00
BA112	ECO	600.00
CX100	ECO	880.00
DL550	ECO	700.00
EK001	ECO	800.00
IB300	ECO	500.00
KE080	ECO	850.00
LH404	ECO	550.00
LX800	ECO	550.00
QR900	ECO	750.00
SA200	ECO	700.00
SQ300	ECO	900.00
TK101	ECO	550.00
NULL	NULL	NULL

```sql
SELECT flight_number, airline
FROM flight
WHERE airline IN ('Lufthansa', 'Air France', 'British Airways');
```

100% 65:96

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_number	airline
AF332	Air France
BA112	British Airways
LH404	Lufthansa
NULL	NULL

```sql
SELECT name, city
FROM airport
WHERE name LIKE '%International%';
```

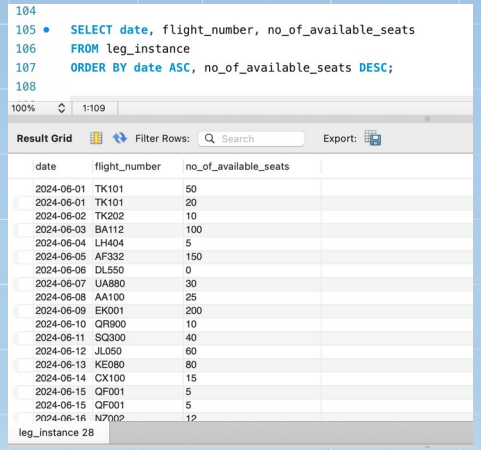
100% 1:93

Result Grid | Filter Rows: Search | Export:

name	city
Athens International Airport	Athens
El Dorado International	Bogota
Dubai International	Dubai
Ministro Pistarini International	Buenos Aires
Incheon International	Seoul
John F. Kennedy International	New York
Los Angeles International	Los Angeles
O'Hare International	Chicago
Beijing Capital International	Beijing
Toronto Pearson International	Toronto

Create 3 queries with ORDER BY clause

- Data was sorted in ascending or descending order for better readability.



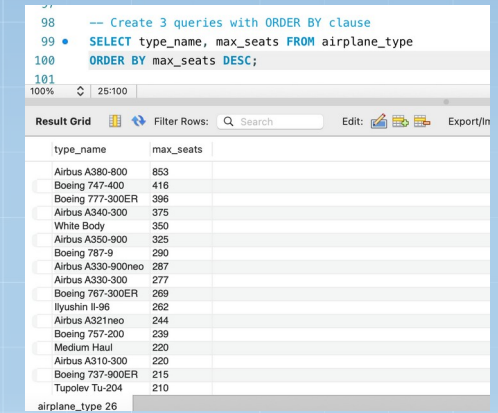
```
104
105 •  SELECT date, flight_number, no_of_available_seats
106    FROM leg_instance
107    ORDER BY date ASC, no_of_available_seats DESC;
108
```

100% 1:109

Result Grid Filter Rows: Search Export:

date	flight_number	no_of_available_seats
2024-06-01	TK101	50
2024-06-01	TK101	20
2024-06-02	TK202	10
2024-06-03	BA112	100
2024-06-04	LH404	5
2024-06-05	AF332	150
2024-06-06	DL550	0
2024-06-07	UA880	30
2024-06-08	AA100	25
2024-06-09	EK001	200
2024-06-10	QR900	10
2024-06-11	SQ300	40
2024-06-12	JL050	60
2024-06-13	KE080	80
2024-06-14	CX100	15
2024-06-15	QF001	5
2024-06-15	QF001	5
2024-06-16	N7002	12

leg_instance 28



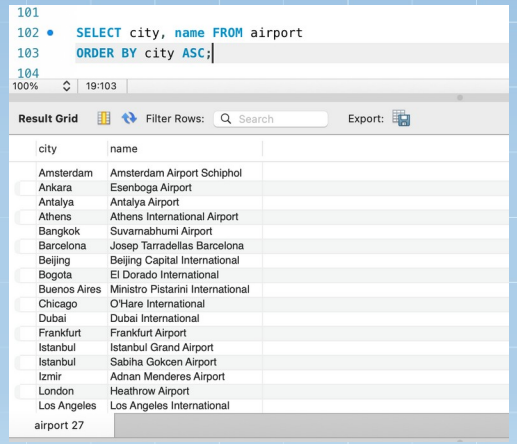
```
98     -- Create 3 queries with ORDER BY clause
99 •   SELECT type_name, max_seats FROM airplane_type
100    ORDER BY max_seats DESC;
101
```

100% 25:100

Result Grid Filter Rows: Search Edit: Export/In

type_name	max_seats
Airbus A380-800	853
Boeing 747-400	416
Boeing 777-300ER	396
Airbus A340-300	375
White Body	350
Airbus A350-900	325
Boeing 787-9	290
Airbus A330-900neo	287
Airbus A330-300	277
Boeing 767-300ER	269
Ilyushin Il-96	262
Airbus A321neo	244
Boeing 757-200	239
Medium Haul	220
Airbus A310-300	220
Boeing 737-900ER	215
Tupolev Tu-204	210

airplane_type 26



```
101
102 •  SELECT city, name FROM airport
103    ORDER BY city ASC;
104
```

100% 19:103

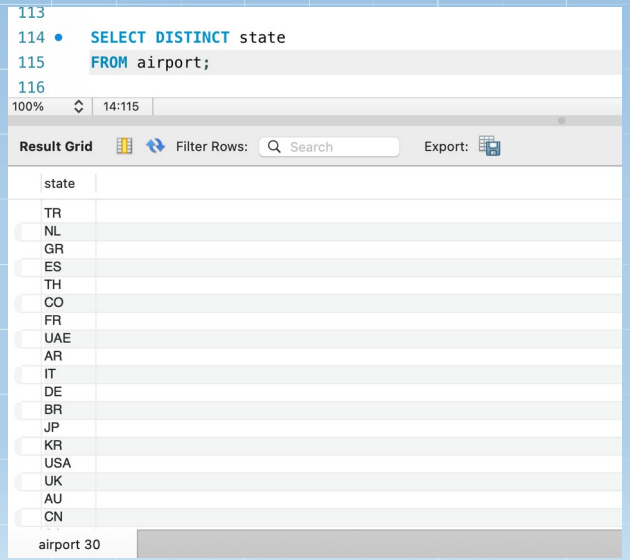
Result Grid Filter Rows: Search Export:

city	name
Amsterdam	Amsterdam Airport Schiphol
Ankara	Esenboga Airport
Antalya	Antalya Airport
Athens	Athens International Airport
Bangkok	Suvarnabhumi Airport
Barcelona	Josep Tarradellas Barcelona
Beijing	Beijing Capital International
Bogota	El Dorado International
Buenos Aires	Ministro Pistarini International
Chicago	O'Hare International
Dubai	Dubai International
Frankfurt	Frankfurt Airport
Istanbul	Istanbul Grand Airport
Istanbul	Sabiha Gokcen Airport
Izmir	Adnan Menderes Airport
London	Heathrow Airport
Los Angeles	Los Angeles International

airport 27

Create 3 queries with DISTINCT clause

- Duplicate records were eliminated to retrieve unique values.



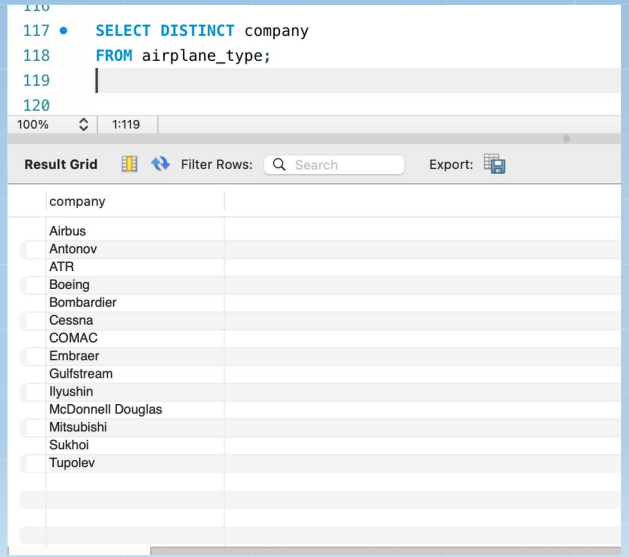
```
113
114 •  SELECT DISTINCT state
115    FROM airport;
116
```

100% 14:115

Result Grid | Filter Rows: Search | Export:

state
TR
NL
GR
ES
TH
CO
FR
UAE
AR
IT
DE
BR
JP
KR
USA
UK
AU
CN

airport 30

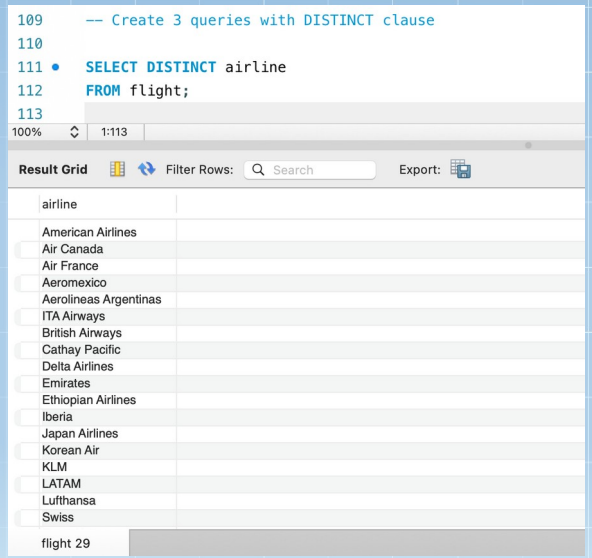


```
117 •  SELECT DISTINCT company
118    FROM airplane_type;
119
120
```

100% 1:119

Result Grid | Filter Rows: Search | Export:

company
Airbus
Antonov
ATR
Boeing
Bombardier
Cessna
COMAC
Embraer
Gulfstream
Ilyushin
McDonnell Douglas
Mitsubishi
Sukhoi
Tupolev



```
109    -- Create 3 queries with DISTINCT clause
110
111 •  SELECT DISTINCT airline
112    FROM flight;
113
```

100% 1:113

Result Grid | Filter Rows: Search | Export:

airline
American Airlines
Air Canada
Air France
Aeromexico
Aerolineas Argentinas
ITA Airways
British Airways
Cathay Pacific
Delta Airlines
Emirates
Ethiopian Airlines
Iberia
Japan Airlines
Korean Air
KLM
LATAM
Lufthansa
Swiss

flight 29

Create 7 queries with String Functions

- String functions were used to manipulate and format textual data.

```
123
124 •  SELECT CONCAT(airline, ' (', flight_number, ')') AS flight_identity
125    FROM flight;
126
```

100% 1:128

Result Grid Filter Rows: Search Export:

flight_identity
American Airlines (AA100)
Air Canada (AC033)
Air France (AF332)
Aeromexico (AM400)
Aerolineas Argentinas (AR110)
ITA Airways (AZ600)
British Airways (BA112)
Cathay Pacific (CX100)
Delta Airlines (DL550)
Emirates (EK001)
Ethiopian Airlines (ET700)
Iberia (IB300)
Japan Airlines (JL050)
Korean Air (KE080)

```
126
127 •  SELECT customer_name, REPLACE(cphone, '555', 'XXX') AS masked_phone
128    FROM seat;
129
130
131
```

100% 1:129

Result Grid Filter Rows: Search Export:

customer_name	masked_phone
Tom Hanks	XXX-0112
Ryan Reynolds	XXX-0121
Jean Dupont	XXX-0109
Frida Kahlo	XXX-0126
Lionel Messi	XXX-0125
Monica Bellucci	XXX-0129
John Smith	XXX-0106
Jane Doe	XXX-0107
Bruce Lee	XXX-0118
Michael Jordan	XXX-0110
Ali Veli	XXX-0113
Abebe Bikila	XXX-0122
Rafael Nadal	XXX-0128
Tanaka Sato	XXX-0116

Result 34

```
129
130 •  SELECT DISTINCT SUBSTRING(flight_number, 1, 2) AS carrier_prefix
131    FROM flight;
132
133
```

100% 1:132

Result Grid Filter Rows: Search Export:

carrier_prefix
AA
AC
AF
AM
AR
AZ
BA
CX
DL
EK
ET
IB
JL
KE

Result 35

Create 7 queries with String Func

```sql
SELECT TRIM(LOWER(city)) AS formatted_city, state
FROM airport;
```

100% 14:134

Result Grid Filter Rows: Search Export:

formatted_city	state
izmir	TR
amsterdam	NL
athens	GR
antalya	TR
barcelona	ES
bangkok	TH
bogota	CO
paris	FR
dubai	UAE
ankara	TR
buenos aires	AR
rome	IT
frankfurt	DE
sao paulo	BR

Result 36

```sql
-- Create 7 queries with String Functions
SELECT UPPER(airport_code) AS code, name
FROM airport;
```

100% 1:123

Result Grid Filter Rows: Search Export:

code	name
ADB	Adnan Menderes Airport
MAD	Adolfo Suárez Madrid–Barajas
AMS	Amsterdam Airport Schiphol
AYT	Antalya Airport
ATH	Athens International Airport
PEK	Beijing Capital International
CDG	Charles de Gaulle Airport
DXB	Dubai International
BOG	El Dorado International
ESB	Esenboga Airport

Result 32

```sql
SELECT type_name, INSTR(type_name, ' ') AS first_space_position
FROM airplane_type;
```

100% 20:141

Result Grid Filter Rows: Search Export:

type_name	first_space_position
Airbus A220-100	7
Airbus A220-300	7
Airbus A310-300	7
Airbus A319	7
Airbus A320-200	7
Airbus A321neo	7
Airbus A330-300	7
Airbus A330-900neo	7
Airbus A340-300	7
Airbus A350-900	7
Airbus A380-800	7
Antonov An-148	8
ATR 42-600	4
ATR 72-600	4

Result 38

```sql
SELECT name, LENGTH(name) AS name_char_count
FROM airport
WHERE LENGTH(name) > 20;
```

100% 25:138

Result Grid Filter Rows: Search Export:

name	name_char_count
Adnan Menderes Airport	22
Adolfo Suárez Madrid–Barajas	31
Amsterdam Airport Schiphol	26
Athens International Airport	28
Beijing Capital International	29
Charles de Gaulle Airport	25
El Dorado International	23
Incheon International	21
Istanbul Grand Airport	22
John F. Kennedy International	29
Josep Tarradellas Barcelona	27
Kingsford Smith Airport	23
Leonardo da Vinci–Fiumicino	29
Los Angeles International	25

Create 7 queries with Numeric Functions

- Numeric functions were applied to perform mathematical calculations.

```
143    -- Create 7 queries with Numeric Functions
144
145 •  SELECT AVG(amount) AS average_fare
146    FROM fare;
147
```

100% 11:146

Result Grid Filter Rows: Search Export:

average_fare
985.517241

Result 39

```
148 •  SELECT MAX(max_seats) AS largest_capacity
149    FROM airplane_type;
150
```

100% 20:149

Result Grid Filter Rows: Search Export:

largest_capac...
853

```
151 •  SELECT SUM(total_no_of_seats) AS fleet_total_seats
152    FROM airplane;
153
154
155
```

100% 1:154

Result Grid Filter Rows: Search Export:

fleet_total_seats
7083

Create 7 queries with Numeric

```
154 •  SELECT flight_number, amount, ROUND(amount * 0.075, 2) AS estimated_tax
155    FROM fare;
156
```

100% 1:157

Result Grid | Filter Rows: Search | Export:

flight_numb...	amount	estimated_t...
AA100	650.00	48.75
AC033	600.00	45.00
AF332	520.00	39.00
AM400	450.00	33.75
AR110	900.00	67.50
AZ600	480.00	36.00

```
157 •  SELECT customer_name, seat_no
158    FROM seat
159    ORDER BY RAND()
160    LIMIT 1;
```

100% 9:160

Result Grid | Filter Rows: Search | Export: | Fetch rows:

customer_name	seat_no
Vincent van Gogh	40C

seat 43

```
162 •  SELECT COUNT(*) AS total_flight_legs
163    FROM flight_leg;
164
```

100% 1:164

Result Grid | Filter Rows: Search | Export:

total_flight_le...
30

Result 44

```
165 •  SELECT
166        flight_number,
167        leg_no,
168        TIMESTAMPDIFF(MINUTE, scheduled_dep_time, scheduled_arr_time) AS actual_minutes,
169        CEIL(TIMESTAMPDIFF(MINUTE, scheduled_dep_time, scheduled_arr_time) / 60.0) AS billing_hours
170    FROM flight_leg;
171
```

100% 1:172

Result Grid | Filter Rows: Search | Export:

flight_numb...	leg_no	actual_minut...	billing_hou...
AA100	1	720	12
AC033	1	2100	35
AF332	1	150	3
AM400	1	360	6
AR110	1	480	8
AZ600	1	240	4
BA112	1	180	3
CX100	1	240	4
DL550	1	720	12
EK001	1	240	4

Create 10 queries with Date Functions

- Date functions were used to extract, compare, and calculate date-based values.

```
172    -- Create 10 queries with Date Functions
173
174 •  SELECT flight_number, date, DAYNAME(date) AS day_of_week
175    FROM leg_instance;
```

100% 1:178

Result Grid | Filter Rows: Search | Export:

flight_number	date	day_of_week
TK101	2024-06-01	Saturday
TK101	2024-06-01	Saturday
UA880	2024-06-07	Friday
TK202	2024-06-02	Sunday
AA100	2024-06-08	Saturday
BA112	2024-06-03	Monday
DL550	2024-06-06	Thursday
LH404	2024-06-04	Tuesday
AF332	2024-06-05	Wednesday
EK001	2024-06-09	Sunday
QR900	2024-06-10	Monday
NZ002	2024-06-16	Sunday
AC033	2024-06-17	Monday
OS900	2024-06-28	Friday
SQ300	2024-06-11	Tuesday

```
177 •  SELECT flight_number, date, MONTHNAME(date) AS flight_month
178    FROM leg_instance;
179
```

100% 19:178

Result Grid | Filter Rows: Search | Export:

flight_number	date	flight_month
TK101	2024-06-01	June
TK101	2024-06-01	June
UA880	2024-06-07	June
TK202	2024-06-02	June
AA100	2024-06-08	June
BA112	2024-06-03	June
DL550	2024-06-06	June
LH404	2024-06-04	June
AF332	2024-06-05	June
EK001	2024-06-09	June
QR900	2024-06-10	June
NZ002	2024-06-16	June
AC033	2024-06-17	June
OS900	2024-06-28	June
SQ300	2024-06-11	June

Create 10 queries with Date

```
SELECT flight_number, leg_no,
       TIMESTAMPDIFF(MINUTE, scheduled_dep_time, scheduled_arr_time) AS duration_minutes
FROM flight_leg;
```

100% 1:179

Result Grid Filter Rows: Search Export:

flight_numb...	leg_no	duration_minut...
AA100	1	720
AC033	1	2100
AF332	1	150
AM400	1	360
AR110	1	480
AZ600	1	240
BA112	1	180
CX100	1	240
DL550	1	720
EK001	1	240
ET700	1	360
IB300	1	120
JL050	1	1380
KE080	1	60
KL700	1	120
LA500	1	480
LH404	1	180

```
SELECT customer_name, YEAR(date) AS booking_year
FROM seat;
```

100% 1:186

Result Grid Filter Rows: Search Export:

customer_name	booking_year
Tom Hanks	2024
Ryan Reynolds	2024
Jean Dupont	2024
Frida Kahlo	2024
Lionel Messi	2024
Monica Bellucci	2024
John Smith	2024
Jane Doe	2024
Bruce Lee	2024
Michael Jordan	2024
Ali Veli	2024
Abebe Bikila	2024
Rafael Nadal	2024
Tanaka Sato	2024
Kim Min	2024
Vincent van Gogh	2024
Pedro Pascal	2024

```
SELECT flight_number, DAYOFMONTH(scheduled_dep_time) AS day_val
FROM flight_leg;
```

100% 17:188

Result Grid Filter Rows: Search Export:

flight_number	day_val
AA100	8
AC033	17
AF332	5
AM400	22
AR110	21
AZ600	25
BA112	3
CX100	14
DL550	6
EK001	9
ET700	18
IB300	24
JL050	12
KE080	13
KL700	26
LA500	23
LH404	4

```
SELECT flight_number, DATEDIFF('2024-06-01', date) AS days_since_flight
FROM leg_instance;
```

100% 19:191

Result Grid Filter Rows: Search Export:

flight_number	days_since_flight
TK101	0
TK101	0
UA880	-6
TK202	-1
AA100	-7
BA112	-2
DL550	-5
LH404	-3
AF332	-4
EK001	-8
QR900	-9
NZ002	-15
AC033	-16
OS900	-27
SQ300	-10
ET700	-17
JL050	-11

Create 10 queries with Date

```
193 •  SELECT flight_number, HOUR(scheduled_dep_time) AS departure_hour
194    FROM flight_leg;
195
```

100% 17:194

Result Grid Filter Rows: Search Export:

flight_number	departure_hour
AA100	18
AC033	19
AF332	13
AM400	10
AR110	23
AZ600	10
BA112	10
CX100	9
DL550	18
EK001	8
ET700	23
IB300	12
JL050	11
KE080	10
KL700	14
LA500	20
LH404	15

```
203 •  SELECT flight_number, date, QUARTER(date) AS fiscal_quarter
204    FROM leg_instance;
205
```

100% 19:204

Result Grid Filter Rows: Search Export:

flight_number	date	fiscal_quart...
TK101	2024-06-01	2
TK101	2024-06-01	2
UA880	2024-06-07	2
TK202	2024-06-02	2
AA100	2024-06-08	2
BA112	2024-06-03	2
DL550	2024-06-06	2
LH404	2024-06-04	2
AF332	2024-06-05	2
EK001	2024-06-09	2
QR900	2024-06-10	2
NZ002	2024-06-16	2
AC033	2024-06-17	2
OS900	2024-06-28	2
SQ300	2024-06-11	2
ET700	2024-06-18	2
JL050	2024-06-12	2

```
198
199 •  SELECT flight_number, scheduled_dep_time
200    FROM flight_leg
201    WHERE scheduled_dep_time > NOW();
202
```

100% 34:201

Result Grid Filter Rows: Search Export:

flight_number	scheduled_dep_time

```
196 •  SELECT flight_number, date, DATE_ADD(date, INTERVAL -1 DAY) AS checkin_opening_date
197    FROM leg_instance;
198
```

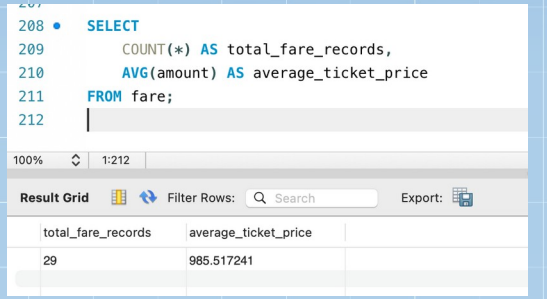
100% 19:197

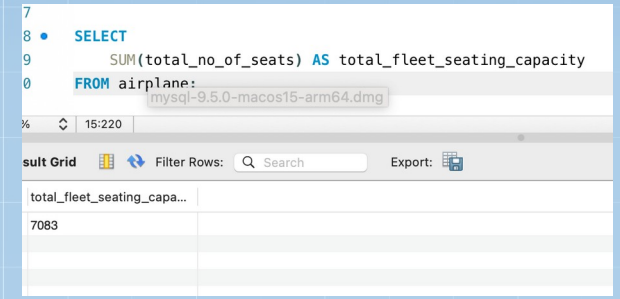
Result Grid Filter Rows: Search Export:

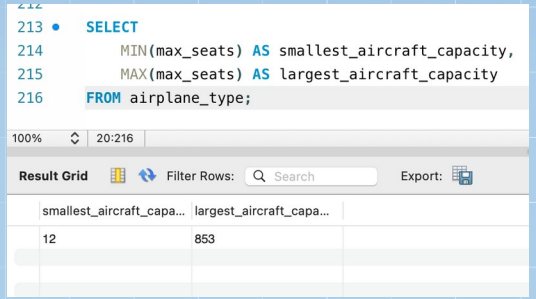
flight_number	date	checkin_opening_date
TK101	2024-06-01	2024-05-31
UA880	2024-06-07	2024-06-06
TK202	2024-06-02	2024-06-01
AA100	2024-06-08	2024-06-07
BA112	2024-06-03	2024-06-02
DL550	2024-06-06	2024-06-05
LH404	2024-06-04	2024-06-03
AF332	2024-06-05	2024-06-04
EK001	2024-06-09	2024-06-08
QR900	2024-06-10	2024-06-09
NZ002	2024-06-16	2024-06-15
AC033	2024-06-17	2024-06-16
OS900	2024-06-28	2024-06-27
SQ300	2024-06-11	2024-06-10
ET700	2024-06-18	2024-06-17
JL050	2024-06-12	2024-06-11
QF001	2024-06-15	2024-06-14

Create 3 queries with aggregate functions(COUNT, MIN, MAX, SUM, AVG)

- Aggregate functions were used to summarize data using count, minimum, maximum, sum, and average.

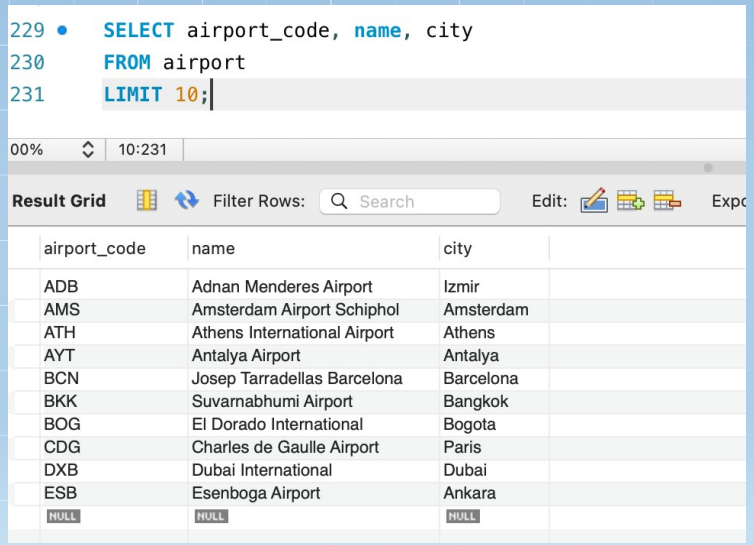






Use LIMIT clause

- The LIMIT clause was used to restrict the number of rows returned.

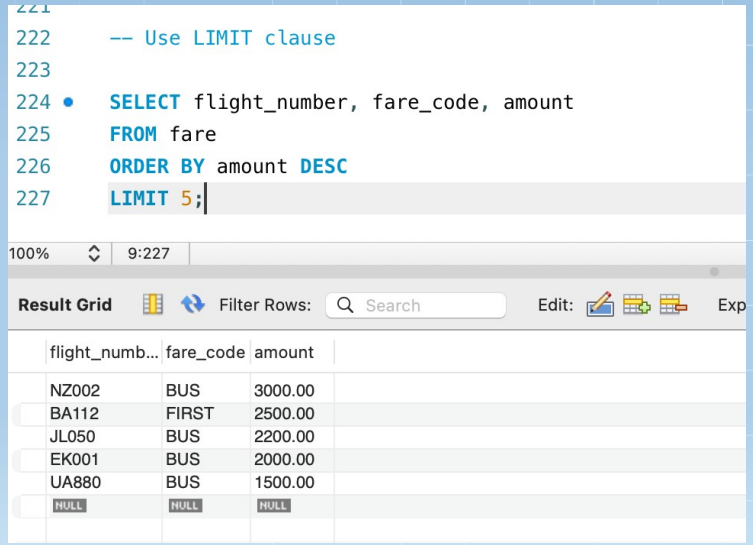


```
229 •  SELECT airport_code, name, city
230    FROM airport
231    LIMIT 10;
```

00% 10:231

Result Grid | Filter Rows: Search | Edit: | Expo

airport_code	name	city
ADB	Adnan Menderes Airport	Izmir
AMS	Amsterdam Airport Schiphol	Amsterdam
ATH	Athens International Airport	Athens
AYT	Antalya Airport	Antalya
BCN	Josep Tarradellas Barcelona	Barcelona
BKK	Suvarnabhumi Airport	Bangkok
BOG	El Dorado International	Bogota
CDG	Charles de Gaulle Airport	Paris
DXB	Dubai International	Dubai
ESB	Esenboga Airport	Ankara
NULL	NULL	NULL



```
222    -- Use LIMIT clause
223
224 •  SELECT flight_number, fare_code, amount
225    FROM fare
226    ORDER BY amount DESC
227    LIMIT 5;
```

100% 9:227

Result Grid | Filter Rows: Search | Edit: | Exp

flight_numb...	fare_code	amount
NZ002	BUS	3000.00
BA112	FIRST	2500.00
JL050	BUS	2200.00
EK001	BUS	2000.00
UA880	BUS	1500.00
NULL	NULL	NULL

Use ROLLUP

- ROLLUP was used to generate subtotal and total values in grouped results.

```
233    -- Use ROLLUP
234 •  SELECT airline, fare_code, AVG(amount) AS average_price
235    FROM fare
236    JOIN flight ON fare.flight_number = flight.flight_number
237    GROUP BY airline, fare_code WITH ROLLUP;
```

100% 41:237

Result Grid | Filter Rows: Search | Export:

airline	fare_code	average_price
Aerolineas Argentinas	ECO	900.000000
Aerolineas Argentinas	NULL	900.000000
Aeromexico	ECO	450.000000
Aeromexico	NULL	450.000000
Air Canada	ECO	600.000000
Air Canada	NULL	600.000000
Air France	ECO	520.000000
Air France	NULL	520.000000
Air New Zealand	BUS	3000.000000
Air New Zealand	NULL	3000.000000
American Airlines	ECO	650.000000
American Airlines	NULL	650.000000
British Airways	ECO	600.000000
British Airways	FIRST	2500.000000

```
239 •  SELECT city, name, COUNT(seat_no) AS total_bookings
240    FROM airport a
241    JOIN flight_leg fl ON a.airport_code = fl.departure_airport_code
242    JOIN seat s ON fl.flight_number = s.flight_number AND fl.leg_no = s.leg_no
243    GROUP BY city, name WITH ROLLUP;
```

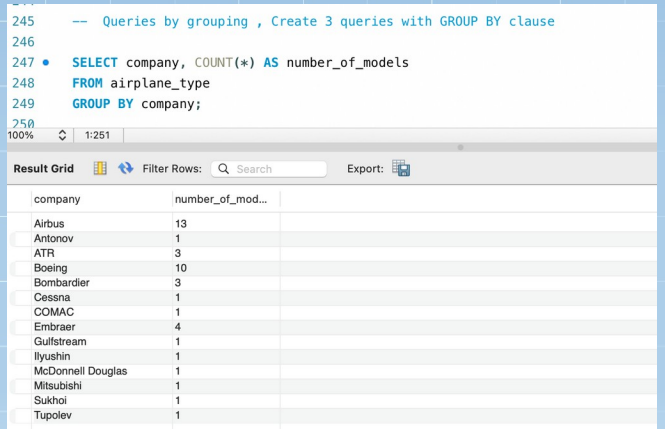
100% 33:243

Result Grid | Filter Rows: Search | Export:

city	name	total_bookings
Amsterdam	Amsterdam Airport Schiphol	1
Amsterdam	NULL	1
Beijing	Beijing Capital International	1
Beijing	NULL	1
Bogota	El Dorado International	2
Bogota	NULL	2
Buenos Aires	Ministro Pistarini International	1
Buenos Aires	NULL	1
Chicago	O'Hare International	1
Chicago	NULL	1
Dubai	Dubai International	3
Dubai	NULL	3
Frankfurt	Frankfurt Airport	1
Frankfurt	NULL	1

Create 3 queries with GROUP BY clause

- Records were grouped based on specific columns to enable summarized analysis.

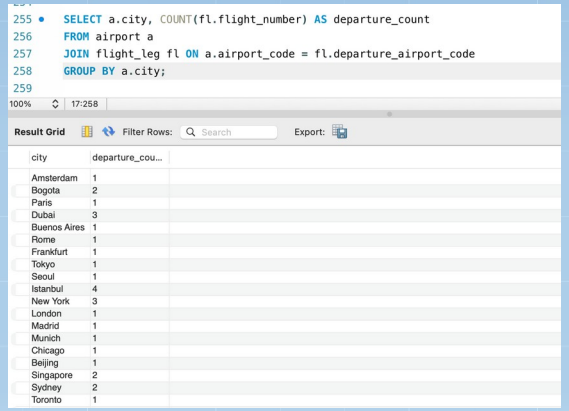


```
245    --  Queries by grouping , Create 3 queries with GROUP BY clause
246
247 •  SELECT company, COUNT(*) AS number_of_models
248    FROM airplane_type
249    GROUP BY company;
250
```

100% 1:251

Result Grid Filter Rows: Search Export:

company	number_of_mod...
Airbus	13
Antonov	1
ATR	3
Boeing	10
Bombardier	3
Cessna	1
COMAC	1
Embraer	4
Gulfstream	1
Ilyushin	1
McDonnell Douglas	1
Mitsubishi	1
Sukhoi	1
Tupolev	1

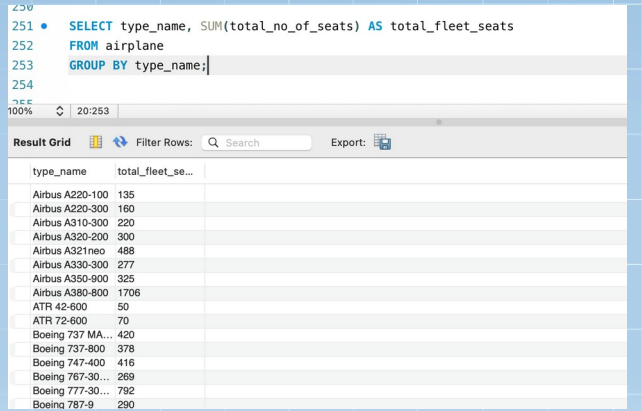


```
255 •  SELECT a.city, COUNT(fl.flight_number) AS departure_count
256    FROM airport a
257    JOIN flight_leg fl ON a.airport_code = fl.departure_airport_code
258    GROUP BY a.city;
259
```

100% 17:258

Result Grid Filter Rows: Search Export:

city	departure_cou...
Amsterdam	1
Bogota	2
Paris	1
Dubai	3
Buenos Aires	1
Rome	1
Frankfurt	1
Tokyo	1
Seoul	1
Istanbul	4
New York	3
London	1
Madrid	1
Munich	1
Chicago	1
Beijing	1
Singapore	2
Sydney	2
Toronto	1



```
251 •  SELECT type_name, SUM(total_no_of_seats) AS total_fleet_seats
252    FROM airplane
253    GROUP BY type_name;
254
```

100% 20:253

Result Grid Filter Rows: Search Export:

type_name	total_fleet_se...
Airbus A220-100	135
Airbus A220-300	160
Airbus A310-300	220
Airbus A320-200	300
Airbus A321neo	488
Airbus A330-300	277
Airbus A350-900	325
Airbus A380-800	1706
ATR 42-600	50
ATR 72-600	70
Boeing 737 MA...	420
Boeing 737-800	378
Boeing 747-400	416
Boeing 767-30...	269
Boeing 777-30...	792
Boeing 787-9	290

Create 3 queries with aggregate functions

- Aggregate functions were applied on grouped data to generate statistical insights.

```sql
-- Create 3 queries with aggregate functions(COUNT, MIN, MAX, SUM, AVG)

SELECT
    COUNT(*) AS total_fare_records,
    AVG(amount) AS average_ticket_price
FROM fare;
```

100% 1:266

Result Grid | Filter Rows: Search | Export:

total_fare_records	average_ticket_price
29	985.517241

```sql
SELECT
    MIN(max_seats) AS smallest_aircraft_capacity,
    MAX(max_seats) AS largest_aircraft_capacity
FROM airplane_type;
```

100% 20:270

Result Grid | Filter Rows: Search | Export:

smallest_aircraft_capa...	largest_aircraft_capa...
12	853

```sql
SELECT
    SUM(total_no_of_seats) AS total_fleet_seating_capacity
FROM airplane;
```

100% 1:275

Result Grid | Filter Rows: Search | Export:

total_fleet_seating_capa...
7083

Create 3 queries with HAVING clause

- The HAVING clause was used to filter grouped results based on aggregate conditions

```
275
276     --  Create 3 queries with HAVING clause
277
278 •   SELECT a.city, COUNT(fl.flight_number) AS departure_count
279     FROM airport a
280     JOIN flight_leg fl ON a.airport_code = fl.departure_airport_code
281     GROUP BY a.city
282     HAVING COUNT(fl.flight_number) > 3;
```

100% 36:282

Result Grid Filter Rows: Search Export:

city	departure_count
Istanbul	4

```
288
289 •   SELECT flight_number, MIN(amount) AS cheapest_ticket
290     FROM fare
291     GROUP BY flight_number
292     HAVING MIN(amount) > 500.00;
```

100% 29:292

Result Grid Filter Rows: Search Export:

flight_number	cheapest_ticket
AA100	650.00
AC033	600.00
AF332	520.00
AR110	900.00
BA112	600.00
CX100	880.00
DL550	700.00
EK001	800.00
JL050	2200.00
KE080	850.00
KL700	1100.00
LA500	1300.00
LH404	550.00
LX800	550.00

```
283
284 •   SELECT company, AVG(max_seats) AS average_seats
285     FROM airplane_type
286     GROUP BY company
287     HAVING AVG(max_seats) > 200;
```

100% 29:287

Result Grid Filter Rows: Search Export:

company	average_seats
Airbus	288.6154
Boeing	254.7000
Ilyushin	262.0000
Tupolev	210.0000

Create a query by LEFT JOIN

```
294     --  Create a query by LEFT JOIN
295
296 •   SELECT
297         f.flight_number,
298         f.airline,
299         fr.fare_code,
300         fr.amount
301     FROM flight f
302     LEFT JOIN fare fr ON f.flight_number = fr.flight_number;
303
```

100% 57:302

Result Grid | Filter Rows: Search | Export:

flight_number	airline	fare_code	amount
AA100	American Airlines	ECO	650.00
AC033	Air Canada	ECO	600.00
AF332	Air France	ECO	520.00
AM400	Aeromexico	ECO	450.00
AR110	Aerolineas Argentinas	ECO	900.00
AZ600	ITA Airways	ECO	480.00
BA112	British Airways	ECO	600.00

Create a query by RIGHT JOIN

```
304    --  Create a query by RIGHT JOIN
305
306    SELECT
307        at.type_name,
308        at.company,
309        a.airplane_id,
310        a.total_no_of_seats
311    FROM airplane a
312    RIGHT JOIN airplane_type at ON a.type_name = at.type_name;
313
```

100% 59:312

Result Grid | Filter Rows: Search | Export:

type_name	company	airplane_id	total_no_of_se...
Airbus A220-100	Airbus	1021	135
Airbus A220-300	Airbus	1022	160
Airbus A310-300	Airbus	1029	220
Airbus A319	Airbus	NULL	NULL
Airbus A320-200	Airbus	1003	150
Airbus A320-200	Airbus	1004	150
Airbus A321neo	Airbus	1014	244
Airbus A321neo	Airbus	1015	244

Create 3 queries with Joining 2 tables

```sql
316    -- 1. View Flight Routes with Airport Names
317    SELECT
318        f.flight_number,
319        f.leg_no,
320        a1.name AS Departure_Airport,
321        a2.name AS Arrival_Airport
322    FROM flight_leg f
323    JOIN airport a1 ON f.departure_airport_code = a1.airport_code
324    JOIN airport a2 ON f.arrival_airport_code = a2.airport_code;
```

100% 61:324

Result Grid Filter Rows: Search Export:

flight_number	leg_no	Departure_Airport	Arrival_Airport
AA100	1	John F. Kennedy International	Heathrow Airport
AC033	1	Toronto Pearson International	Kingsford Smith Airport
AF332	1	Charles de Gaulle Airport	John F. Kennedy International
AM400	1	El Dorado International	John F. Kennedy International
AR110	1	Ministro Pistarini International	El Dorado International
AZ600	1	Leonardo da Vinci–Fiumicino	John F. Kennedy International
BA112	1	Heathrow Airport	John F. Kennedy International
CX100	1	Beijing Capital International	John F. Kennedy International
DL550	1	John F. Kennedy International	Heathrow Airport
EK001	1	Dubai International	Heathrow Airport
ET700	1	Istanbul Grand Airport	Heathrow Airport
IB300	1	Adolfo Suárez Madrid–Barajas	John F. Kennedy International

```sql
326    -- 2. Identify Passengers and Their Flights
327    SELECT
328        s.customer_name,
329        s.seat_no,
330        s.flight_number,
331        f.airline
332    FROM seat s
333    JOIN flight f ON s.flight_number = f.flight_number;
334
```

100% 52:333

Result Grid Filter Rows: Search Export:

customer_name	seat_no	flight_numb...	airline
Tom Hanks	15F	AA100	American Airlines
Ryan Reynolds	9F	AC033	Air Canada
Jean Dupont	4D	AF332	Air France
Frida Kahlo	20E	AM400	Aeromexico
Lionel Messi	18D	AR110	Aerolineas Argentinas
Monica Bellucci	35B	AZ600	ITA Airways
John Smith	12A	BA112	British Airways
Jane Doe	12B	BA112	British Airways
Bruce Lee	6C	CX100	Cathay Pacific
Michael Jordan	1A	DL550	Delta Airlines
Ali Veli	1A	EK001	Emirates
Abebe Bikila	11A	ET700	Ethiopian Airlines
Rafael Nadal	30A	IB300	Iberia
Tanaka Sato	44H	JL050	Japan Airlines
Kim Min	5A	KE080	Korean Air

Create 3 queries with Joining 2

```
335    -- 3. Match Airplanes with Their Specifications
336 •  SELECT
337        a.airplane_id,
338        t.company,
339        t.type_name,
340        t.max_seats
341    FROM airplane a
342    JOIN airplane_type t ON a.type_name = t.type_name;
343
```

100% 51:342

Result Grid | Filter Rows: Search | Export:

airplane_id	company	type_name	max_seats
1021	Airbus	Airbus A220-100	135
1022	Airbus	Airbus A220-300	160
1029	Airbus	Airbus A310-300	220
1003	Airbus	Airbus A320-200	150
1004	Airbus	Airbus A320-200	150
1014	Airbus	Airbus A321neo	244
1015	Airbus	Airbus A321neo	244
1024	Airbus	Airbus A330-300	277
1008	Airbus	Airbus A350-900	325
1009	Airbus	Airbus A380-800	853
1010	Airbus	Airbus A380-800	853
1026	ATR	ATR 42-600	50
1025	ATR	ATR 72-600	70
1019	Boeing	Boeing 737 MAX 8	210
1020	Boeing	Boeing 737 MAX 8	210

Create 3 queries with Joining 3 tables

```sql
-- Create 3 queries with Joining 3 tables

-- 1. Link Passengers to Aircraft Specifications
SELECT
    s.customer_name,
    s.flight_number,
    i.airplane_id,
    t.company,
    t.type_name
FROM seat s
JOIN leg_instance i ON s.flight_number = i.flight_number
    AND s.leg_no = i.leg_no
    AND s.date = i.date
JOIN airplane_type t ON i.airplane_id IN (
    SELECT airplane_id FROM airplane WHERE type_name = t.type_name
);
```

100% 3:359

Result Grid | Filter Rows: Search | Export:

customer_name	flight_numb...	airplane_id	company	type_name
Tom Hanks	AA100	1004	Airbus	Airbus A320-200
Ryan Reynolds	AC033	1012	Embraer	Embraer E190
Jean Dupont	AF332	1008	Airbus	Airbus A350-900
Frida Kahlo	AM400	1021	Airbus	Airbus A220-100
Lionel Messi	AR110	1030	McDonnell Douglas	McDonnell Douglas M...
Monica Bellucci	AZ600	1026	ATR	ATR 42-600
John Smith	BA112	1005	Boeing	Boeing 777-300ER

```sql
-- 2. Route Details with Full Airport Names
SELECT
    f.airline,
    fl.flight_number,
    fl.scheduled_dep_time,
    a.name AS departure_airport,
    a.city
FROM flight f
JOIN flight_leg fl ON f.flight_number = fl.flight_number
JOIN airport a ON fl.departure_airport_code = a.airport_code;
```

100% 1:371

Result Grid | Filter Rows: Search | Export:

airline	flight_number	scheduled_dep_time	departure_airport	city
American Airlines	AA100	2024-06-08 18:00:00	John F. Kennedy International	New York
Air Canada	AC033	2024-06-17 19:00:00	Toronto Pearson International	Toronto
Air France	AF332	2024-06-05 13:00:00	Charles de Gaulle Airport	Paris
Aeromexico	AM400	2024-06-22 10:00:00	El Dorado International	Bogota
Aerolineas Argentinas	AR110	2024-06-21 23:00:00	Ministro Pistarini International	Buenos Aires
ITA Airways	AZ600	2024-06-25 10:00:00	Leonardo da Vinci–Fiumicino	Rome
British Airways	BA112	2024-06-03 10:00:00	Heathrow Airport	London
Cathay Pacific	CX100	2024-06-14 09:00:00	Beijing Capital International	Beijing

Create 3 queries with Joining 3

```
373     -- 3. Comprehensive Flight Instance Report
374 •   SELECT
375         f.airline,
376         li.date,
377         li.flight_number,
378         fr.fare_code,
379         fr.amount
380     FROM flight f
381     JOIN leg_instance li ON f.flight_number = li.flight_number
382     JOIN fare fr ON f.flight_number = fr.flight_number;
383
```

100% 1:383

Result Grid Filter Rows: Search Export:

airline	date	flight_number	fare_code	amount
American Airlines	2024-06-08	AA100	ECO	650.00
Air Canada	2024-06-17	AC033	ECO	600.00
Air France	2024-06-05	AF332	ECO	520.00
Aeromexico	2024-06-22	AM400	ECO	450.00
Aerolineas Argentinas	2024-06-21	AR110	ECO	900.00
ITA Airways	2024-06-25	AZ600	ECO	480.00
British Airways	2024-06-03	BA112	ECO	600.00
British Airways	2024-06-03	BA112	FIRST	2500.00
Cathay Pacific	2024-06-14	CX100	ECO	880.00
Delta Airlines	2024-06-06	DL550	ECO	700.00

Create 4 queries with subquery in WHERE clause

```
383
384     -- Create 4 queries with subquery in WHERE clause
385     -- Find all flights using aircraft with more than 300 seats
386     SELECT flight_number, date, airplane_id
387     FROM leg_instance
388     WHERE airplane_id IN (
389         SELECT airplane_id
390         FROM airplane
391         WHERE total_no_of_seats > 300
392     );
```

100% 34:391

Result Grid | Filter Rows: Search | Export:

flight_number	date	airplane_id
BA112	2024-06-03	1005
DL550	2024-06-06	1006
AF332	2024-06-05	1008
EK001	2024-06-09	1009
QR900	2024-06-10	1010
JL050	2024-06-12	1016

```
404     -- Identify fares that are cheaper than the average of all fares:
405     SELECT flight_number, fare_code, amount
406     FROM fare
407     WHERE amount < (
408         SELECT AVG(amount)
409         FROM fare
410     );
411
```

00% 3:410

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_number	fare_code	amount
AA100	ECO	650.00
AC033	ECO	600.00
AF332	ECO	520.00
AM400	ECO	450.00
AR110	ECO	900.00
AZ600	ECO	480.00
BA112	ECO	600.00
CX100	ECO	880.00
DL550	ECO	700.00

Create 4 queries with subquery in WHERE clause

```sql
-- Find flights that depart from airports capable of landing an 'Airbus A380-800':
SELECT flight_number, departure_airport_code
FROM flight_leg
WHERE departure_airport_code IN (
    SELECT airport_code
    FROM can_land
    WHERE type_name = 'Airbus A380-800'
);
```

100% 3:419

Result Grid | Filter Rows: Search | Export:

flight_number	departure_airport_c...
KL700	AMS
EK001	DXB
QR900	DXB
SA200	DXB
ET700	IST
MS900	IST
TK101	IST
TK202	IST
BA112	LHR

```sql
-- List airports in the same state as 'Istanbul Airport'
SELECT name, city, state
FROM airport
WHERE state = (
    SELECT state
    FROM airport
    WHERE name = 'Istanbul Airport'
);
```

100% 1:403

Result Grid | Filter Rows: Search | Edit: | Export/Import:

airport_code	name	city	state
ADB	Adnan Menderes Airport	Izmir	TR
AYT	Antalya Airport	Antalya	TR
ESB	Esenboga Airport	Ankara	TR
IST	Istanbul Grand Airport	Istanbul	TR
SAW	Sabiha Gokcen Airport	Istanbul	TR
NULL	NULL	NULL	NULL

Create 3 queries in the WHERE clause, including at least 1 correlated subquery

```
);

-- List flight legs where the available seats are less than 5% of that specific airplane's total capacity
SELECT li.flight_number, li.date, li.no_of_available_seats
FROM leg_instance li
WHERE li.no_of_available_seats < (
    SELECT a.total_no_of_seats * 0.05
    FROM airplane a
    WHERE a.airplane_id = li.airplane_id
);
```

3:439

Result Grid | Filter Rows: Search | Export:

flight_number	date	no_of_available_seats
DL550	2024-06-06	0
LH404	2024-06-04	5
LX800	2024-06-27	5
OS900	2024-06-28	1
QF001	2024-06-15	5
QF001	2024-06-15	5
QR900	2024-06-10	10

Create 3 queries in the WHERE clause, including at least 1

```sql
-- Identify fares that are the most expensive for their specific flight
SELECT f1.flight_number, f1.fare_code, f1.amount
FROM fare f1
WHERE f1.amount = (
    SELECT MAX(f2.amount)
    FROM fare f2
    WHERE f2.flight_number = f1.flight_number
);
```

3:448

Result Grid | Filter Rows: Search | Edit: | Export/Import:

flight_number	fare_code	amount
AA100	ECO	650.00
AC033	ECO	600.00
AF332	ECO	520.00
AM400	ECO	450.00
AR110	ECO	900.00
AZ600	ECO	480.00
BA112	FIRST	2500.00
CX100	ECO	880.00
DL550	ECO	700.00
EK001	BUS	2000.00

```sql
-- Create 3 queries in the WHERE clause, including at least 1 correlated subquery
-- Find airplane types that have a higher capacity than the average capacity of their own manufacturer
SELECT t1.type_name, t1.company, t1.max_seats
FROM airplane_type t1
WHERE t1.max_seats > (
    SELECT AVG(t2.max_seats)
    FROM airplane_type t2
    WHERE t2.company = t1.company
);
```

1:431

Result Grid | Filter Rows: Search | Edit: | Export/Import:

type_name	company	max_seats
Airbus A340-300	Airbus	375
Airbus A350-900	Airbus	325
Airbus A380-800	Airbus	853
ATR 72-600	ATR	70
Boeing 747-400	Boeing	416
Boeing 767-300ER	Boeing	269
Boeing 777-300ER	Boeing	396
Boeing 787-9	Boeing	290
Bombardier CRJ900	Bombardier	90
Embraer E190	Embraer	114

Create a correlated subquery using at least 1 WHERE EXISTS or WHERE NOT EXISTS.

```
450   -- Create a correlated subquery using at least 1 WHERE EXISTS or WHERE NOT EXISTS
451   -- Find airports that currently have NO flights scheduled to depart from them (NOT EXISTS)
452
453 • SELECT a.name, a.city
454   FROM airport a
455   WHERE NOT EXISTS (
456       SELECT 1
457       FROM flight_leg fl
458       WHERE fl.departure_airport_code = a.airport_code
459   );
```

3:459

Result Grid | Filter Rows: Search | Export:

name	city
Adnan Menderes Airport	Izmir
Athens International Airport	Athens
Antalya Airport	Antalya
Josep Tarradellas Barcelona	Barcelona
Suvarnabhumi Airport	Bangkok
Esenboga Airport	Ankara
São Paulo/Guarulhos	Sao Paulo
Los Angeles International	Los Angeles
Melbourne Airport	Melbourne
Malpensa Airport	Milan
Sabiha Gokcen Airport	Istanbul
Venice Marco Polo Airport	Venice

```
461   -- Find airplane types that are actually assigned to at least one flight instance (EXISTS)
462 • SELECT at.type_name, at.company
463   FROM airplane_type at
464   WHERE EXISTS (
465       SELECT 1
466       FROM airplane a
467       JOIN leg_instance li ON a.airplane_id = li.airplane_id
468       WHERE a.type_name = at.type_name
469   );
470
```

1:470

Result Grid | Filter Rows: Search | Edit: | Export/Import:

type_name	company
Airbus A220-100	Airbus
Airbus A220-300	Airbus
Airbus A310-300	Airbus
Airbus A320-200	Airbus
Airbus A321neo	Airbus
Airbus A330-300	Airbus
Airbus A350-900	Airbus
Airbus A380-800	Airbus
ATR 42-600	ATR
ATR 72-600	ATR
Boeing 737 MAX 8	Boeing
Boeing 737-800	Boeing

Create 3 queries with subquery in SELECT columns

```sql
-- Compare Flight Seats to Aircraft Maximum
SELECT
    airplane_id,
    type_name,
    total_no_of_seats,
    (SELECT max_seats
     FROM airplane_type
     WHERE airplane_type.type_name = airplane.type_name) AS designed_max_seats
FROM airplane;
```

15:488

Result Grid | Filter Rows: Search | Export:

airplane_id	type_name	total_no_of_se...	designed_max_se...
1001	Boeing 737-800	189	200
1002	Boeing 737-800	189	200
1003	Airbus A320-200	150	150
1004	Airbus A320-200	150	150
1005	Boeing 777-300ER	396	396
1006	Boeing 777-300ER	396	396
1007	Boeing 787-9	290	290
1008	Airbus A350-900	325	325
1009	Airbus A380-800	853	853
1010	Airbus A380-800	853	853
1011	Embraer E190	114	114
1012	Embraer E190	114	114

```sql
-- Create 3 queries with subquery in SELECT columns
-- 1. View Flight Fares vs. Average Fare
SELECT
    flight_number,
    fare_code,
    amount,
    (SELECT AVG(amount) FROM fare) AS global_avg_fare
FROM fare;
```

1:479

Result Grid | Filter Rows: Search | Export:

flight_number	fare_code	amount	global_avg_fa...
AA100	ECO	650.00	985.517241
AC033	ECO	600.00	985.517241
AF332	ECO	520.00	985.517241
AM400	ECO	450.00	985.517241
AR110	ECO	900.00	985.517241
AZ600	ECO	480.00	985.517241
BA112	ECO	600.00	985.517241
BA112	FIRST	2500.00	985.517241
CX100	ECO	880.00	985.517241
DL550	ECO	700.00	985.517241
EK001	BUS	2000.00	985.517241
EK001	ECO	800.00	985.517241

Create 3 queries with subquery in SELECT columns

```sql
-- Display Booking Count for Each Flight Leg
SELECT
    flight_number,
    leg_no,
    departure_airport_code,
    (SELECT COUNT(*)
     FROM seat
     WHERE seat.flight_number = flight_leg.flight_number
       AND seat.leg_no = flight_leg.leg_no) AS passenger_count
FROM flight_leg;
```

45:490

Result Grid | Filter Rows: Search | Export:

flight_number	leg_no	departure_airport_c...	passenger_cou...
KL700	1	AMS	1
AM400	1	BOG	1
LA500	1	BOG	1
AF332	1	CDG	1
EK001	1	DXB	1
QR900	1	DXB	1
SA200	1	DXB	1
AR110	1	EZE	1
AZ600	1	FCO	1
LH404	1	FRA	1
JL050	1	HND	1
KE080	1	ICN	1

Copy one table structure and data to new table

```
500
501     -- Copy one table structure and data to new table
502 •   CREATE TABLE airport_backup AS
503     SELECT * FROM airport;
```

00% 23:503

tion Output

	Time	Action	Response
603	13:54:17	CREATE TABLE airport_backup AS SELECT * FROM airport	32 row(s) affected Records: 32

Using MySQL user-defined variables (@variables), create at least 2 queries that list the tuple with the highest value in a category or a ranking.

```sql
-- 2. Identifying the Highest Fare per Airline
SET @row_num := 0;
SET @curr_airline := '';
SELECT airline, flight_number, fare_code, amount
FROM (
    SELECT
        @row_num := IF(@curr_airline = f.airline, @row_num + 1, 1) AS airline_rank,
        @curr_airline := f.airline AS dummy,
        f.airline,
        fr.flight_number,
        fr.fare_code,
        fr.amount
    FROM fare fr
    JOIN flight f ON fr.flight_number = f.flight_number
    ORDER BY f.airline, fr.amount DESC
) AS ranked_fares
WHERE airline_rank = 1;
```

100% 24:537

Result Grid Filter Rows: Search Export:

airline	flight_number	fare_code	amount
Aerolineas Argentinas	AR110	ECO	900.00
Aeromexico	AM400	ECO	450.00
Air Canada	AC033	ECO	600.00
Air France	AF332	ECO	520.00
Air New Zealand	NZ002	BUS	3000.00

```sql
-- Using MySQL user-defined variables (@variables),
--  create at least 2 queries that list the tuple with the highest value in a category or a ranking.
-- 1.Ranking Airplane Types by Capacity
SET @rank := 0;
SELECT rank_position, type_name, max_seats, company
FROM (
    SELECT
        @rank := @rank + 1 AS rank_position,
        type_name,
        max_seats,
        company
    FROM airplane_type
    ORDER BY max_seats DESC
) AS ranked_fleet
WHERE rank_position <= 3;
```

100% 16:508

Result Grid Filter Rows: Search Export:

rank_positi...	type_name	max_seats	company
1	Airbus A380-800	853	Airbus
2	Boeing 747-400	416	Boeing
3	Boeing 777-300ER	396	Boeing

Create a VIEW

```sql
use airline;
-- Create a VIEW
CREATE VIEW daily_flight_schedule AS
SELECT
    f.airline,
    fl.flight_number,
    fl.leg_no,
    a1.city AS departure_city,
    fl.scheduled_dep_time,
    a2.city AS arrival_city,
    fl.scheduled_arr_time,
    apt.type_name AS aircraft_model
FROM flight f
JOIN flight_leg fl ON f.flight_number = fl.flight_number
JOIN airport a1 ON fl.departure_airport_code = a1.airport_code
JOIN airport a2 ON fl.arrival_airport_code = a2.airport_code
JOIN leg_instance li ON fl.flight_number = li.flight_number AND fl.leg_no = li.leg_no
JOIN airplane ap ON li.airplane_id = ap.airplane_id
JOIN airplane_type apt ON ap.type_name = apt.type_name;
```

100% 1:20

ction Output

	Time	Action		Response
609	14:06:48	CREATE VIEW daily_flight_schedule AS SELECT	f.airline,...	0 row(s) affected

Create and call a stored procedure

```
21
22    --  Create a stored procedure
23    DELIMITER //
24    CREATE PROCEDURE ScheduleFlightLeg(
25        IN p_flight_number VARCHAR(10),
26        IN p_new_dep_time DATETIME
27    )
28    BEGIN
29        UPDATE flight_leg
30        SET scheduled_dep_time = p_new_dep_time
31        WHERE flight_number = p_flight_number AND leg_no = 1;
32        SELECT flight_number, leg_no, departure_airport_code, scheduled_dep_time
33        FROM flight_leg
34        WHERE flight_number = p_flight_number;
35    END //
36    DELIMITER ;
37
38
```

100% 12:36

Action Output

	Time	Action	Response
612	14:13:10	CREATE PROCEDURE ScheduleFlightLeg(IN p_flight_numb...	0 row(s) affected

```
37
38    -- Call the Stored Procedure
39    CALL ScheduleFlightLeg('TK101', '2024-06-01 09:30:00');
40
```

100% 56:39

Result Grid Filter Rows: Search Export:

flight_number	leg_no	departure_airport_c...	scheduled_dep_time
TK101	1	IST	2024-06-01 09:30:00
TK101	2	JFK	2024-06-01 14:00:00

Result 1

Action Output

	Time	Action	Response
613	14:13:41	CALL ScheduleFlightLeg('TK101', '2024-06-01 09:30:00')	2 row(s) returned

BUSİNESS TRİGGERS

```sql
use airline;
-- TRİGERS
-- BUSİNESS RULES
-- RULE 1: Data Integrity - Flight Scheduling Rule
DELIMITER //
CREATE TRIGGER before_flight_leg_insert
BEFORE INSERT ON flight_leg
FOR EACH ROW
BEGIN
    IF NEW.departure_airport_code = NEW.arrival_airport_code THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Error: Departure and arrival airports must be distinct.';
    END IF;
END //
DELIMITER ;
```

100% 12:15

Action Output

	Time	Action	Response
614	14:24:01	CREATE TRIGGER before_flight_leg_insert BEFORE INSERT O...	0 row(s) affected

BUSİNESS TRİGGERS

```sql
-- RULE 2: Reservation Capacity Constraint (SEAT and RESERVATION)
DELIMITER //
CREATE TRIGGER TRG_Rule2_ReservationConstraint
BEFORE INSERT ON reservation
FOR EACH ROW
BEGIN
    -- Check if physical seat is already taken for this instance
    IF EXISTS (SELECT 1 FROM seat WHERE flight_number = NEW.flight_number
               AND leg_no = NEW.leg_no AND date = NEW.date AND seat_no = NEW.seat_no) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Rule 2: Physical seat already assigned to another reservation.';
    END IF;
    -- Check if aircraft is full
    IF (SELECT no_of_available_seats FROM leg_instance
        WHERE flight_number = NEW.flight_number AND leg_no = NEW.leg_no AND date = NEW.date) <= 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Rule 2: No available seats for this flight instance.';
    END IF;
END //
DELIMITER ;
```

00% 29:20

ction Output

	Time	Action	Response
621	14:39:31	CREATE TRIGGER TRG_Rule2_ReservationConstraint BEFORE...	0 row(s) affected

BUSİNESS TRİGGERS

```
73
74    -- RULE 5: POSITIVE FARE VALUE RULE
75
76    DELIMITER //
77    CREATE TRIGGER TRG_Rule5_FareIntegrity
78    BEFORE INSERT ON fare
79    FOR EACH ROW
80    BEGIN
81        IF NEW.amount < 0 THEN
82            SIGNAL SQLSTATE '45000'
83            SET MESSAGE_TEXT = 'Rule 5: Fare amount cannot be negative.';
84        END IF;
85
86        IF NEW.restrictions IS NULL OR NEW.restrictions = '' THEN
87            SIGNAL SQLSTATE '45000'
88            SET MESSAGE_TEXT = 'Rule 5: Every fare must have recorded restrictions.';
89        END IF;
90    END //
91    DELIMITER ;
92
```

00% 12:91

ction Output

	Time	Action	Response
624	14:43:45	CREATE TRIGGER TRG_Rule5_FareIntegrity BEFORE INSERT...	0 row(s) affected

BUSİNESS TRİGGERS

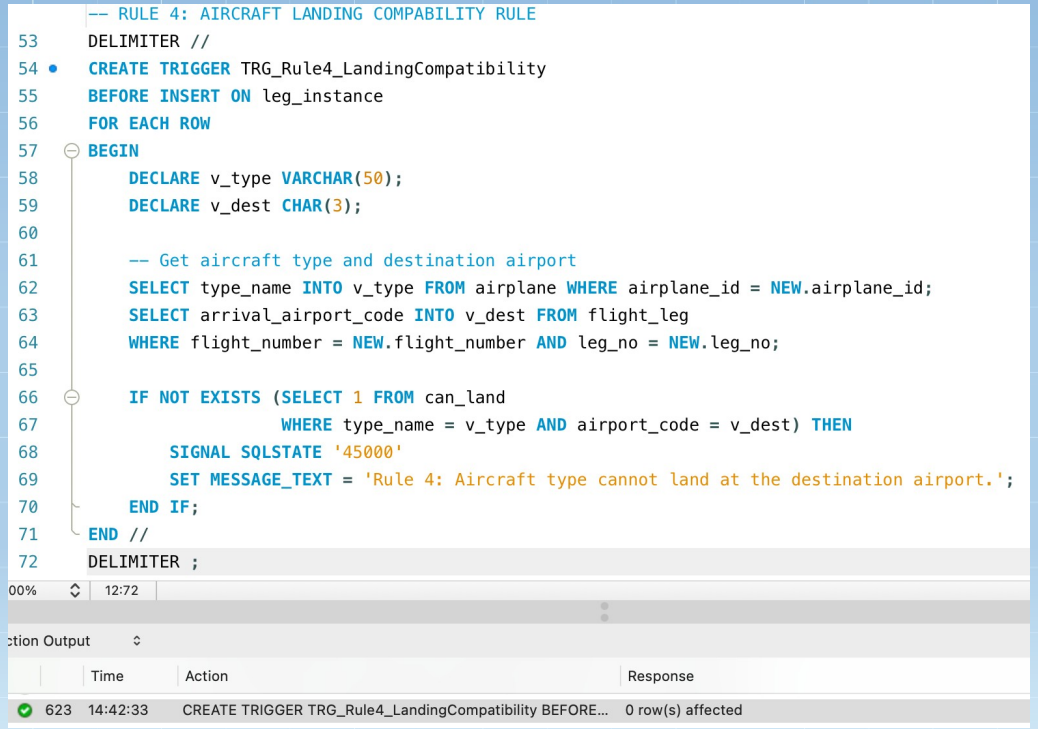
```
38    -- RULE 3: SEAT CAPACITY RULE
39    DELIMITER //
40    CREATE TRIGGER TRG_Rule3_UpdateAvailability
41    AFTER INSERT ON reservation
42    FOR EACH ROW
43    BEGIN
44        UPDATE leg_instance
45        SET no_of_available_seats = no_of_available_seats - 1
46        WHERE flight_number = NEW.flight_number
47          AND leg_no = NEW.leg_no
48          AND date = NEW.date;
49    END //
50    DELIMITER ;
51
```

00% 12:50

ction Output

	Time	Action	Response
622	14:41:28	CREATE TRIGGER TRG_Rule3_UpdateAvailability AFTER INSE...	0 row(s) affected

BUSİNESS TRİGGERS



```
-- RULE 4: AIRCRAFT LANDING COMPABILITY RULE
DELIMITER //
CREATE TRIGGER TRG_Rule4_LandingCompatibility
BEFORE INSERT ON leg_instance
FOR EACH ROW
BEGIN
    DECLARE v_type VARCHAR(50);
    DECLARE v_dest CHAR(3);

    -- Get aircraft type and destination airport
    SELECT type_name INTO v_type FROM airplane WHERE airplane_id = NEW.airplane_id;
    SELECT arrival_airport_code INTO v_dest FROM flight_leg
    WHERE flight_number = NEW.flight_number AND leg_no = NEW.leg_no;

    IF NOT EXISTS (SELECT 1 FROM can_land
                   WHERE type_name = v_type AND airport_code = v_dest) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Rule 4: Aircraft type cannot land at the destination airport.';
    END IF;
END //
DELIMITER ;
```

00% 12:72

ction Output

	Time	Action	Response
623	14:42:33	CREATE TRIGGER TRG_Rule4_LandingCompatibility BEFORE...	0 row(s) affected