FATİH DOĞAÇ
1901042654
DATABASE PROJECT REPORT

1. User Requirements:

a) Airlines

- **Add New Airline**: The system should allow administrators to add new airlines with details such as name and country.
- **View Airline Details**: Users should be able to view a list of all airlines and details of a specific airline.
- **Update Airline Information**: Administrators should be able to update the information of an existing airline.
- **Delete Airline**: Administrators should be able to delete an airline from the system.

b) Flights

- **Schedule New Flight**: The system should allow airlines to schedule new flights, including details such as origin, destination, departure time, and arrival time.
- **View Flight Schedule**: Users should be able to view all scheduled flights and details of a specific flight.
- **Update Flight Information**: Airlines should be able to update the information of scheduled flights.
- **Cancel Flight**: Airlines should be able to cancel scheduled flights.

c) Passengers

- **Register Passenger**: The system should allow passengers to register by providing their name and passport number.
- **View Passenger Details**: Users should be able to view a list of all registered passengers and details of a specific passenger.
- **Update Passenger Information**: Passengers should be able to update their information.
- **Delete Passenger**: Administrators should be able to delete a passenger's record from the system.

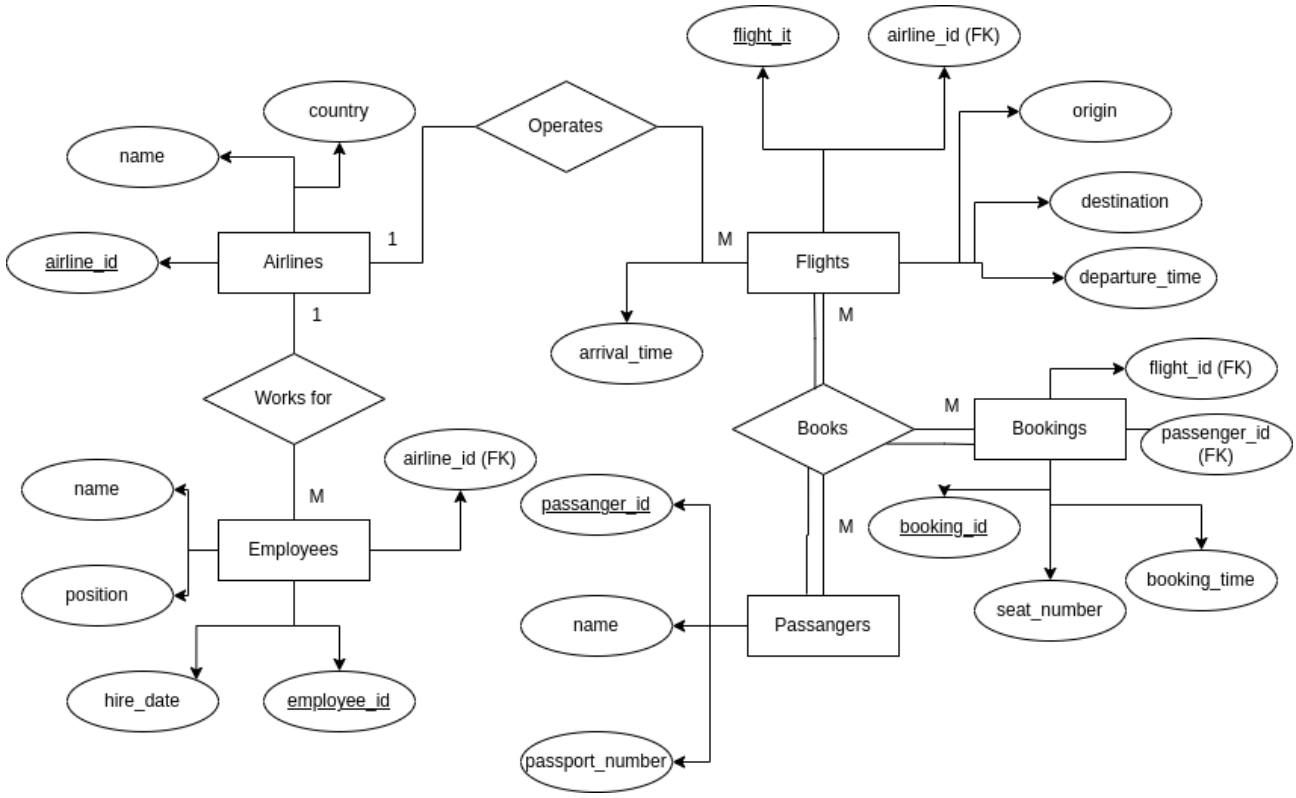
d) Bookings

- **Make Booking**: Passengers should be able to book flights by selecting a flight, seat number, and providing booking time.
- **View Booking Details**: Users should be able to view all bookings and details of a specific booking.
- **Update Booking Information**: Passengers should be able to update their booking information (e.g., change seat number).
- **Cancel Booking**: Passengers should be able to cancel their bookings.

e) Employees

- **Add New Employee**: The system should allow administrators to add new employees with details such as name, position, and hire date.
- **View Employee Details**: Users should be able to view a list of all employees and details of a specific employee.
- **Update Employee Information**: Administrators should be able to update the information of an existing employee.
- **Delete Employee**: Administrators should be able to delete an employee from the system.

2. E-R Diagram



3. Functional Dependencies

Airlines

airline_id -> name, country

Flights

flight_id -> airline_id, origin, destination, departure_time, arrival_time

airline_id, origin, destination, departure_time -> arrival_time

Passengers

passenger_id -> name, passport_number

passport_number -> passenger_id, name

Bookings

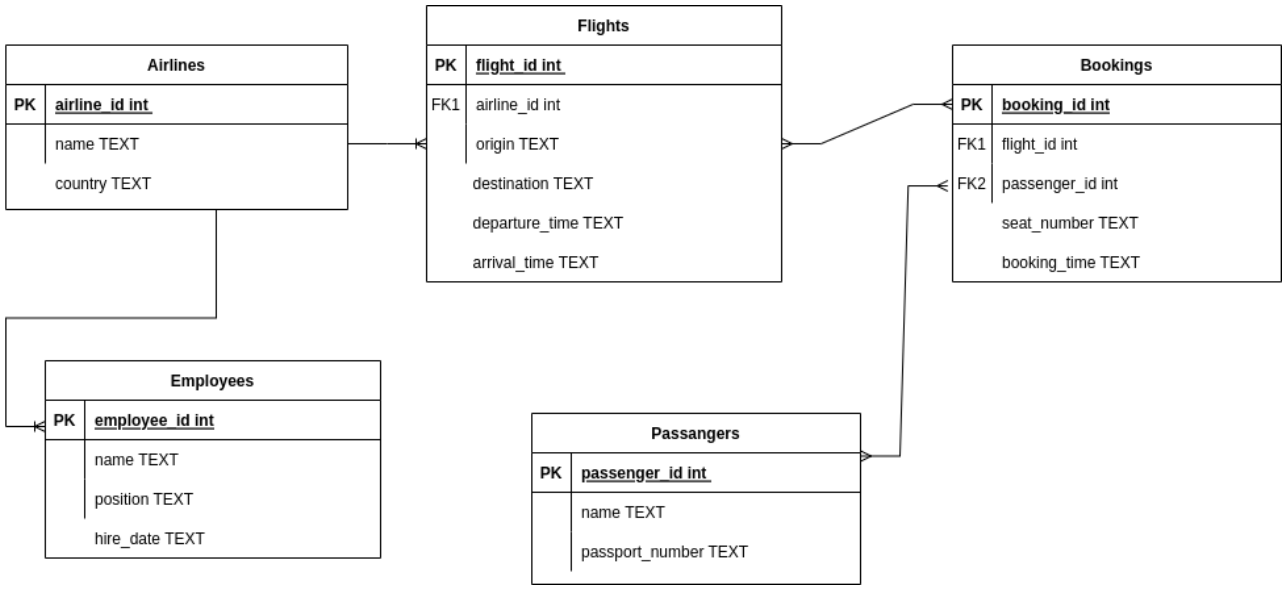
booking_id -> flight_id, passenger_id, seat_number, booking_time

flight_id, passenger_id -> booking_id, seat_number, booking_time

Employees

employee_id -> airline_id, name, position, hire_date

4. Database Schema



5. Database Implementation

I used sqlite3 python library for this part. The reason I choose this library is, it was very simple and very direct. You can just give SQL queries inside a function and it just executes them, totally it was what I wanted.

Creation done in create.py and showing tables done via show_tables.py

```
conn = sqlite3.connect('airport.db')
cursor = conn.cursor()

cursor.execute('''
    SELECT *
    FROM Airlines
''')

rows = cursor.fetchall()
print_query_results("Airlines", rows)
```

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ make
rm -f airport.db
python3 create.py
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 show_tables.py

Airlines:
airline_id | name      | country
---------------------------------
1          | Airline A | Country A
2          | Airline B | Country B
3          | Airline C | Country C

Flights:
flight_id | airline_id | origin | destination | departure_time   | arrival_time
-------------------------------------------------------------------------------
1         | 1          | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00
2         | 2          | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
3         | 1          | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00

Passengers:
passenger_id | name        | passport_number
-----------------------------------------
1            | John Doe    | P1234567
2            | Jane Smith  | P7654321
3            | Fatih Dogac | P1234569
4            | Yunus Emre  | P1236669

Employees:
employee_id | airline_id | name          | position     | hire_date
-----------------------------------------------------------------
1           | 1          | Alice Johnson | Pilot        | 2020-01-01
2           | 2          | Bob Brown     | Ground Staff | 2021-02-15
3           | 1          | Lebron James  | Janitor      | 2021-02-20
4           | 3          | Mike Tyson    | Security     | 2021-02-10

Bookings:
booking_id | flight_id | passenger_id | seat_number | booking_time
-------------------------------------------------------------------
1          | 1         | 1            | 12A         | 2024-06-14 12:00
2          | 2         | 2            | 14B         | 2024-06-14 13:00
3          | 2         | 3            | 14Z         | 2024-06-14 14:00
4          | 3         | 4            | 17C         | 2024-06-14 15:00
5          | 3         | 2            | 21X         | 2024-06-14 16:00
6          | 4         | 2            | 40S         | 2024-06-14 17:00
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$
```

6. User interface

All execuitons and showing the tables done in terminal.

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 triggers.py
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 test_triggers_delete_passenger.py
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 show_tables.py

Airlines:
airline_id | name      | country
-----------------------------------
1          | Airline A | Country A
2          | Airline B | Country B
3          | Airline C | Country C

Flights:
flight_id | airline_id | origin | destination | departure_time   | arrival_time
-------------------------------------------------------------------------------
1         | 1          | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00
2         | 2          | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
3         | 1          | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00

Passengers:
passenger_id | name        | passport_number
-----------------------------------------
1            | John Doe    | P1234567
2            | Jane Smith  | P7654321
3            | Fatih Dogac | P1234569

Employees:
employee_id | airline_id | name          | position     | hire_date
-------------------------------------------------------------------
1           | 1          | Alice Johnson | Pilot        | 2020-01-01
2           | 2          | Bob Brown     | Ground Staff | 2021-02-15
3           | 1          | Lebron James  | Janitor      | 2021-02-20
4           | 3          | Mike Tyson    | Security     | 2021-02-10

Bookings:
booking_id | flight_id | passenger_id | seat_number | booking_time
-----------------------------------------------------------------
1          | 1         | 1            | 12A         | 2024-06-14 12:00
2          | 2         | 2            | 14B         | 2024-06-14 13:00
3          | 2         | 3            | 14Z         | 2024-06-14 14:00
5          | 3         | 2            | 21X         | 2024-06-14 16:00
6          | 4         | 2            | 40S         | 2024-06-14 17:00
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$
```

7. Query Development

Left outer join, Right outer join and Full outer join

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 joins.py

Left Outer Join (Airlines - Flight):
airline_id | name      | country   | flight_id | airline_id | origin | destination | departure_time   | arrival_time
-----------------------------------------------------------------------------------------------------------------------
1          | Airline A | Country A | 1         | 1          | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00
1          | Airline A | Country A | 3         | 1          | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00
2          | Airline B | Country B | 2         | 2          | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
3          | Airline C | Country C | None      | None       | None   | None        | None             | None

Right Outer Join (Airlines - Flights):
flight_id | airline_id | origin | destination | departure_time   | arrival_time     | airline_id | name      | country
-----------------------------------------------------------------------------------------------------------------------
1         | 1          | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00 | 1          | Airline A | Country A
2         | 2          | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00 | 2          | Airline B | Country B
3         | 1          | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00 | 1          | Airline A | Country A

Full Outer Join (Airlines - Flights):
airline_id | name      | country   | flight_id | airline_id | origin | destination | departure_time   | arrival_time
-----------------------------------------------------------------------------------------------------------------------
1          | Airline A | Country A | 1         | 1          | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00
1          | Airline A | Country A | 3         | 1          | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00
2          | Airline B | Country B | 2         | 2          | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
3          | Airline C | Country C | None      | None       | None   | None        | None             | None
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 triggers.py
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 test_triggers_delete_passenger.py
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 show_tables.py
```

8. Triggers

1. Cascade Delete Bookings

```sql
CREATE TRIGGER cascade_delete_bookings
    AFTER DELETE ON Passengers
    FOR EACH ROW
    BEGIN
        DELETE FROM Bookings
        WHERE passenger_id = OLD.passenger_id;
    END;
```

When a passenger gets deleted, passenger's bookings are also deleted.

2. Enforce Referential Integrity

```sql
CREATE TRIGGER enforce_referential_integrity
    BEFORE INSERT ON Bookings
    FOR EACH ROW
    BEGIN
        SELECT RAISE(ABORT, 'Invalid flight_id or passenger_id')
        WHERE (SELECT COUNT(*) FROM Flights WHERE flight_id = NEW.flight_id) = 0
        OR (SELECT COUNT(*) FROM Passengers WHERE passenger_id = NEW.passenger_id) = 0;
    END;
```

It checks when inserting a booking, if the flight_id and passenger_id is valid.

3. Enforce Referential Integrity on Update

```sql
CREATE TRIGGER update_enforce_referential_integrity
    BEFORE UPDATE ON Bookings
    FOR EACH ROW
    BEGIN
        SELECT RAISE(ABORT, 'Invalid flight_id or passenger_id')
        WHERE (SELECT COUNT(*) FROM Flights WHERE flight_id = NEW.flight_id) = 0
        OR (SELECT COUNT(*) FROM Passengers WHERE passenger_id = NEW.passenger_id) = 0;
    END;
```

4. Assign employee id on insertion

```sql
CREATE TRIGGER assign_employee_id
    AFTER INSERT ON Employees
    BEGIN
        UPDATE Employees
        SET employee_id = COALESCE((SELECT MAX(employee_id) FROM Employees), 0)
        WHERE rowid = NEW.rowid;
    END;
```

It gives max(employee_id) + 1 to new employees id.

5. Prevent delete airline with employees

```
CREATE TRIGGER prevent_delete_airline_with_employees
    BEFORE DELETE ON Airlines
    FOR EACH ROW
    WHEN EXISTS (SELECT 1 FROM Employees WHERE airline_id = OLD.airline_id)
    BEGIN
        SELECT RAISE(ABORT, 'Cannot delete airline with employees');
    END;
```

If an airline has employees, it can't be deleted.

9. Views

1. Passangers and Bookings

```
CREATE TRIGGER prevent_delete_airline_with_employees
    BEFORE DELETE ON Airlines
    FOR EACH ROW
    WHEN EXISTS (SELECT 1 FROM Employees WHERE airline_id = OLD.airline_id)
    BEGIN
        SELECT RAISE(ABORT, 'Cannot delete airline with employees');
    END;
```

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 views.py
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 test_views_passanger_bookings.py

Passanger Bookings:
passenger_id | passenger_name | passport_number | booking_id | flight_id | seat_number | booking_time     | origin | destination | departure_time   | arrival_time
-------------------------------------------------------------------------------------------------------------------------------------------------------------
1            | John Doe       | P1234567        | 1          | 1         | 12A         | 2024-06-14 12:00 | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00
2            | Jane Smith     | P7654321        | 2          | 2         | 14B         | 2024-06-14 13:00 | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
3            | Fatih Dogac    | P1234569        | 3          | 2         | 14Z         | 2024-06-14 14:00 | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
4            | Yunus Emre     | P1236669        | 4          | 3         | 17C         | 2024-06-14 15:00 | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00
2            | Jane Smith     | P7654321        | 5          | 3         | 21X         | 2024-06-14 16:00 | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$
```

2. Employee and Airlines

```
CREATE VIEW Employee_Airlines AS
    SELECT e.employee_id, e.name AS employee_name, e.position, e.hire_date,
        a.name AS airline_name, a.country
    FROM Employees e
    INNER JOIN Airlines a ON e.airline_id = a.airline_id;
```

```
Employee Airlines:
employee_id | employee_name | position     | hire_date  | airline_name | country
-------------------------------------------------------------------------------
1           | Alice Johnson | Pilot        | 2020-01-01 | Airline A    | Country A
2           | Bob Brown     | Ground Staff | 2021-02-15 | Airline B    | Country B
3           | Lebron James  | Janitor      | 2021-02-20 | Airline A    | Country A
4           | Mike Tyson    | Security     | 2021-02-10 | Airline C    | Country C
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$
```

3. Flight schedules

```sql
CREATE VIEW Flight_Schedule AS
    SELECT f.flight_id, f.airline_id, a.name AS airline_name, a.country AS airline_country,
        f.origin, f.destination, f.departure_time, f.arrival_time
    FROM Flights f
    INNER JOIN Airlines a ON f.airline_id = a.airline_id;
```

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 test_views_passanger_bookings.py

Flight Schedules:
flight_id | airline_id | airline_name | airline_country | origin | destination | departure_time   | arrival_time
-----------------------------------------------------------------------------------------------------------
1         | 1          | Airline A    | Country A       | City A | City B      | 2024-06-12 08:00 | 2024-06-15 10:00
2         | 2          | Airline B    | Country B       | City C | City D      | 2024-06-15 09:00 | 2024-06-15 11:00
3         | 1          | Airline A    | Country A       | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00
```

4. Booking Statistics

It calculates total bookings of a flight and latest booking time.

```sql
CREATE VIEW Booking_Statistics AS
    SELECT f.flight_id, f.origin, f.destination,
        COUNT(b.booking_id) AS total_bookings,
        MAX(b.booking_time) AS latest_booking_time
    FROM Flights f
    LEFT JOIN Bookings b ON f.flight_id = b.flight_id
    GROUP BY f.flight_id, f.origin, f.destination;
```

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 test_views_passanger_bookings.py

Booking Statistics:
flight_id | origin | destination | total_bookings | latest_booking_time
-----------------------------------------------------------------------
1         | City A | City B      | 1              | 2024-06-14 12:00
2         | City C | City D      | 2              | 2024-06-14 14:00
3         | City X | City T      | 2              | 2024-06-14 16:00
```

5. Upcoming Flights

```sql
CREATE VIEW Upcoming_Flights AS
    SELECT flight_id, airline_id, origin, destination, departure_time, arrival_time
    FROM Flights
    WHERE departure_time > DATETIME('now');
```

```
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$ python3 test_views_passanger_bookings.py

Upcoming Flights:
flight_id | airline_id | origin | destination | departure_time   | arrival_time
-------------------------------------------------------------------------------
3         | 1          | City X | City T      | 2024-06-17 11:00 | 2024-06-18 11:00
koso@koso-ABRA-A5-V17-2:~/Desktop/24/db$
```

Video Link: https://youtu.be/0AH170CZpXo