UML Diagram

Assumptions:

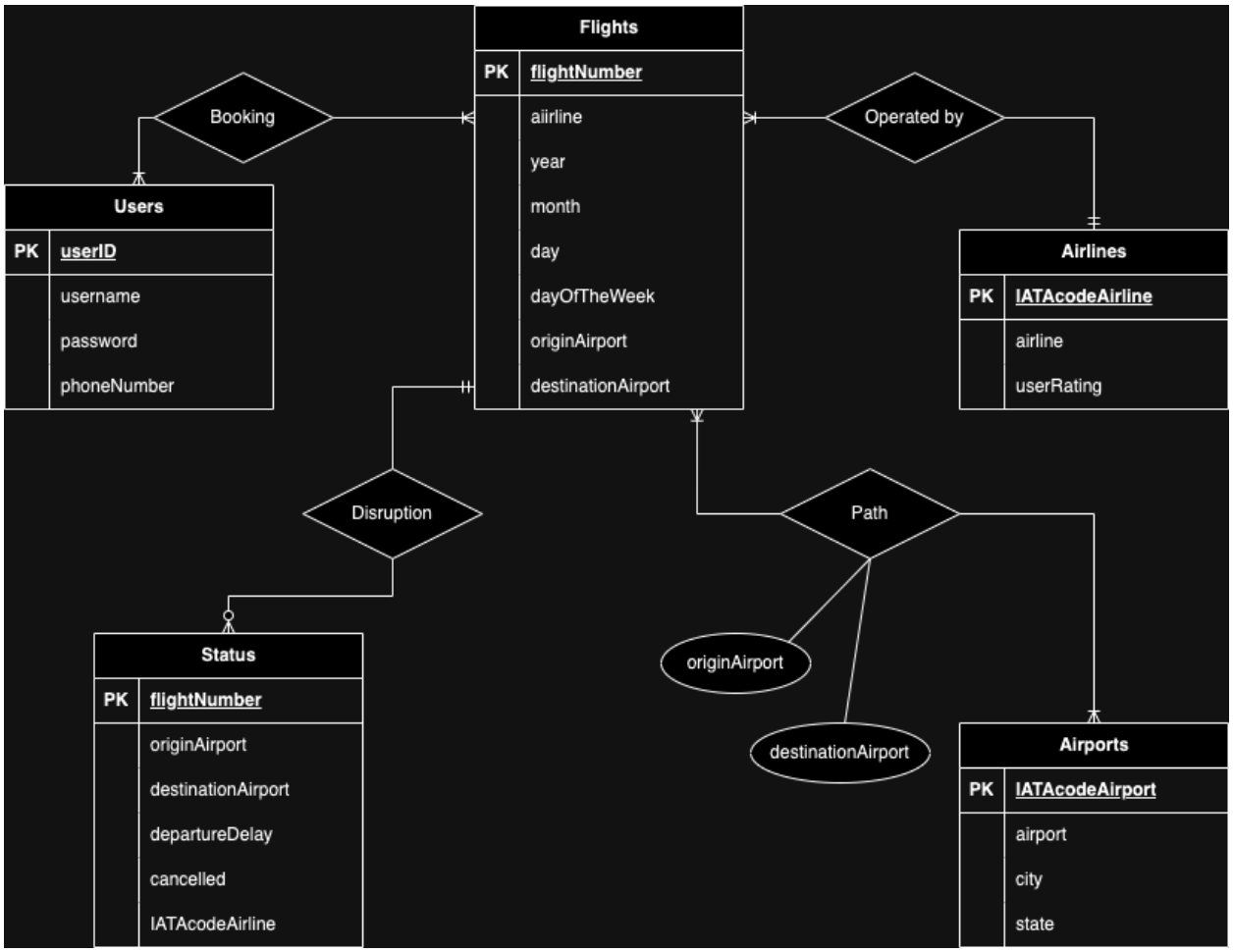
- We are assuming every user in our system has at least 1 booking
- We are assuming every flight has at least 1 user booked
- We are assuming each booking is only for one user but users can have multiple bookings
- Each booking will contain a flight, it will never be empty
- We are assuming every flight belongs to only 1 airline
- We are assuming every airline has at least 1 airplane/flight
- We are assuming every airport is working and has at least 1 flight
- We are assuming every flight, if not in path, belongs to at least 1 airport
- We are assuming every airport in our database is running with departure and arrival flights
- We are assuming every flight has a origin airport and destination airport
- We are assuming every flight might have a disruption in the status (could be none)
- We are assuming every disruption is only shown for 1 flight
- Our dataset will only contain flights in the United States
- Our dataset will not contain any diverted flights
- We will not take into account layover flights. Flights from one location to another to another will be considered 2 different flight paths, rather than 1 flight path with a layover
- Our dataset will not contain reasons for flight delays

Relationships & Cardinality:

- Booking: users have a booking of at least 1 flight and each flight has at least 1 person booked
    - Users to flights has a 1-many relationship because every user in our database will have a booking of at least 1 flight (users can have more than 1)
    - Flights to users has a 1-many relationship because every flight in our database has more than 1 user booking (flights have have more than 1)
- Operated by: airlines have at least 1 flight and each flight has only 1 airline
    - Airlines to flights has a 1-many relationship because every airline has at least 1 flight (airlines can have more than 1) since all airlines are assumed to be running
    - Flights to airlines has a 1-1 relationship because every flight has only 1 airline (flights cannot have more than 1) since flights cannot be used for multiple airlines
- Path: airports have at least 1 flight and each flight has at least 1 airport
    - Airports to flights has a 1-many relationship because airports have at least 1 flight at all times (airports can have more than 1) since all airports are assumed to be running
    - Flights to airports has a 1-many relationship because every flight is linked to at least 1 airport (flights can have more than 1 airport) through origin and destination airports

- Disruption:
    - Flights to status has a 0-many relationship because every flight can have none or many delays and/or cancellations
    - Status to flights has a 1-1 relationship because every status can only be displayed for 1 flight (a single status cannot be shown for multiple flights)

Diagram:



Relationship Schema:

Users (
    userID: INT [PK],
    username: VARCHAR(100),
    password: VARCHAR(100),
    phoneNumber: VARCHAR(15)
)

Flights (

```
    flightNumber: INT [PK] [FK for Status.flightNumber],
    airline: VARCHAR(100),
    year: INT,
    month: VARCHAR(15),
    day: VARCHAR(2),
    dayOfTheWeek: VARCHAR(15),
    originAirport: VARCHAR(100),
    destinationAirport: VARCHAR(100)
)

Airlines (
    IATAcodeAirline: VARCHAR(3) [PK],
    airline: VARCHAR(100),
    userRating: INT
)

Status (
    flightNumber: INT [PK] [FK for Flights.flightNumber],
    originAirport: VARCHAR(100),
    destinationAirport: VARCHAR(100),
    departureDelay: INT,
    canceled: BOOL,
    IATAcodeAirline: VARCHAR(3) [FK for Airlines.IATAcodeAirline]
)

Airports (
    IATAcodeAirport: VARCHAR(3) [PK],
    airport: VARCHAR(100),
    city: VARCHAR(100),
    state: VARCHAR(100)
)
```

We chose to use 3NF because it improves the organization of our data and removes duplicate values, thus improving data accuracy and quality. This form of normalization also helps maintain data integrity by reducing the risk of errors when inserting, updating, and deleting data.