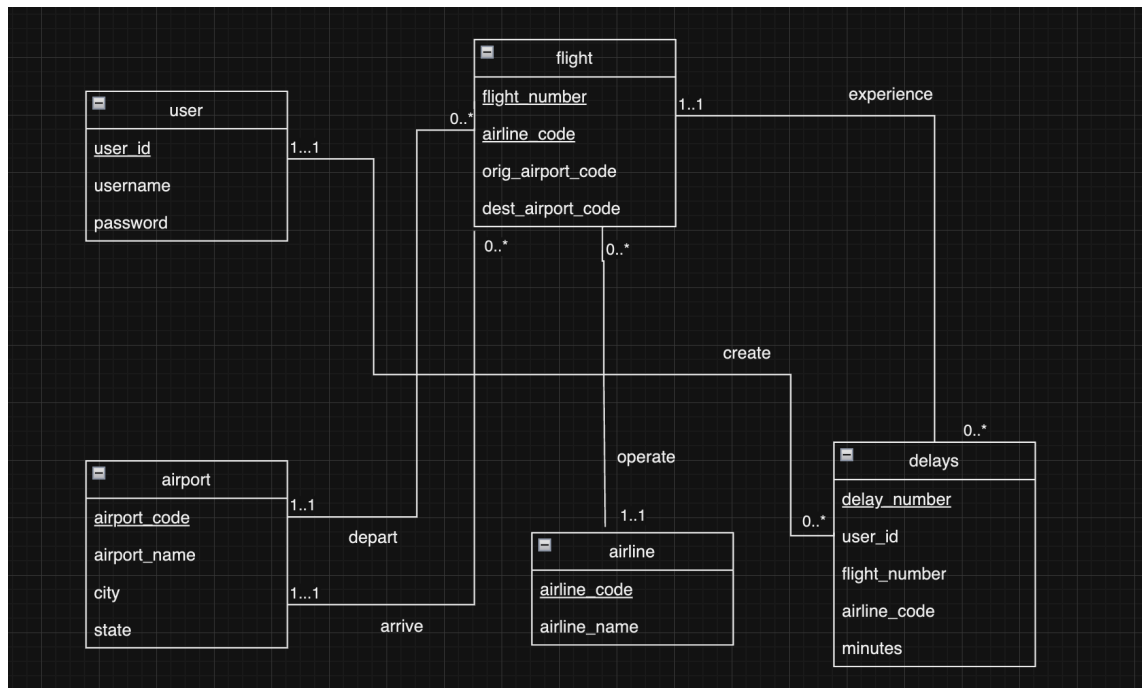# UML Diagram

Assumption

- We assume each delay can only be added by one user and each user can create zero or more delays
- We assume each flight can depart from one airport and each airport can have zero or more departure flights.
- We assume each flight can arrive at one airport and each airport can have zero or more arrival flights.
- We assume each flight has one airline and each airline can have zero or more flights.
- We assume each flight can experience zero or more delays and each delay can only be associated with one flight

Rules on determining whether a relationship should be represented as a separate table or solely as foreign keys

If one entity (e.g a user) can be associated with multiple instances of another entity (e.g. delays), a new table is created to represent the "many" side. However, if a relationship is one to one, we would just put data on both sides in the same table, with some being used as foreign keys.

Why use BCNF

Because it complies with the definition of BCNF. Most of our tables have only one primary key. The only one with two primary keys also comply with BCNF.

BCNF further reduces the possibility of anomalies in our data compared to 3NF. It ensures that all functional dependencies within the table are based on superkeys, which helps prevent more types of data anomalies, including insertion, update, and deletion anomalies. BCNF minimizes data redundancy by reducing partial dependencies. In 3NF, there may still be some redundancy, while BCNF minimizes it to a greater extent.

Normalizing Database

user(user_id, username, password) with FD:
user_id → username, password

flight(flight_number, airline_code, orig_airport_code, dest_airport_code) with FD:
flight_number, airline_code → origin_airport_code, dest_airport_code.

airport(airport_code, airport_name, city, state) with FD:
airport_code → airport_name, city, state

airline(airline_code, airline_name) with FD:
airline_code → airline_name

delays(delay_number, user_id, flight_number, airline_code, minutes) with FD:
delay_number → user_id, flight_number, airline_code, minutes

Since all of the above functional dependencies are non-trivial, and the left sides are all keys (superkeys), all of the above relations are in BCNF.

Relational Schema

Table-Name(Column1:Domain [PK], Column2:Domain [FK to table.column], Column3:Domain,...)

1. user(user_id:INT [PK], username:VARCHAR(255), password: INT)

2. delay(delay_number: INT [PK], user_id: INT [FK to user.user_id], flight_number: INT [FK to flight.flight_number], airline_code: VARCHAR(2)[FK to flight.airline_code], minutes: INT)

3. flight(flight_number: INT [PK], airline_code: VARCHAR(2) [PK], orig_airport_code:VARCHAR(4) [FK to airport.airport_code], dest_airport_code:VARCHAR(4) [FK to airport.airport_code])

4. airport(airport_code:VARCHAR(4) [PK], airport_name: VARCHAR(255), city: VARCHAR(255), state: VARCHAR(2))

5. airline(airline_code: VARCHAR(2) [PK], airline_name: VARCHAR(255))