# COSC3380: Database ER design, Normalization and Web App An Enterprise Airline Database System

# HOMEWORK 3 REPORT

Team 23, and our assignment is target 3: airport flight management

**1. Introduction**

You will extend an existing database and develop a web application program that combines SQL statements and a host language which can be executed on a web browser.

**2. Target application (3):** airport flight management.

flight id, airplane, takeoff timestamp, landing timestamp, passengers, crew, checkin, baggage, gate, refueling, maintenance, cleaning, food/beverage, movie Y/N, wifi Y/N. You can assume passengers, tickets and crew info are already stored by another app.

**3. A flight management system(FMS)**

A flight management system is a specialized computer system that automates a wide variety of in-flight tasks.

Airport management is about management of airport and airline operations. It includes managing, supervising, coordinating operations and maintenance of the airport.

Airport Management ensures that flight operations run smoothly, passengers receive quality service, flights take off and land timely, security is maintained and commercial operations like duty-free shopping at airports, and airport retail return a good profit.

A primary function is in-flight management of the flight plan.

**4. Our web page contain 7 parts:**

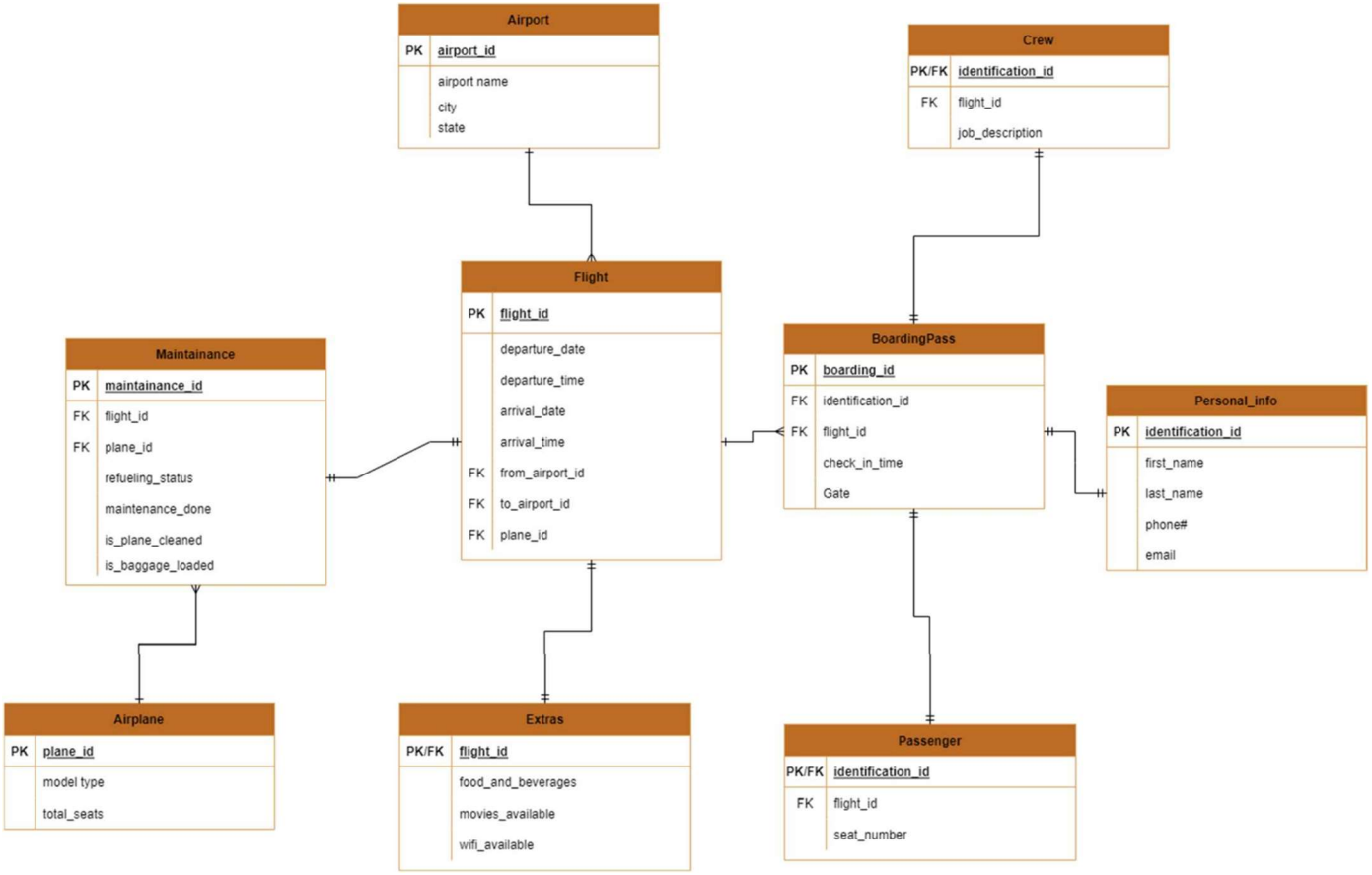
Each table should be able to be reached through the web application. This can be done by clicking each individual tab on the web application. The tabs will change to a crimson color when hovering over them, and when clicked the information section will display the designated table to the user. Here is what one is able to do on each of the individual tabs. VIDEO LINK: https://www.youtube.com/channel/UCfkSQt0u2HwOPYCjLj6Zkig/videos

- Search flights tab
  - The user can insert a flight id and all the information regarding that flight id will be displayed on the screen upon clicking search, if the flight id does not exist nothing will display and an alert will tell the user of the case.
- All flights tab
  - Here the user can add a new flight by inserting new values to the flight table. A user must input flight id, date1, time1, date2, time2, airport code, airport code, and plane id. Upon insertion the information will be inserted into the table and user will be able to see immediately. By creating a new flight, the web app also inserts a new row for flight extras, and plane support, which the user should update the information later. The user is also able to update and delete existing flights, this is the only place on the web app where the user can delete something. All values must be acceptable in order for a transaction to go through, for example if an airport code is not an actual airport code defined through the airport table, a flight cannot be added or updated.
- Flights extras
  - Here the user can see what will be available during a flight such as food andbeverages, movies and wifi. If something turns out to be available that was currently not, it can be updated, same as if something turns out to no be available. On the edit modal, there is options for each

value, several options for food and beverage, yes and no only for movies and Wi-Fi. This prevents the user from having to type this information manually and make a mistake.

- Plane support
  - Here the user can see whether the plane is ready for take off or not. It displays fuel status, maintenance needed, if the plane is cleaned, and if the baggage has beenloaded or not. The user is also able to update this to keep up with work done for a flight. On the edit modal, there is options for each value, options for fuel status, maintenance, cleaning, and baggage. This prevents the user from having to type this information manually and make a mistake. The only manual typing value is a plane id, and if the plane id does not refer to a proper plane model through the planes table, the transaction should not go through.
- Boarding information
  - Here the user can see all the boarding information for each crew member and passenger. It is just a display, and no information can be edited here as crew and passenger information is to be handled by another database.
- Search passenger/ search crew
  - Here the user can use the identification id of a person to look them up and get their personal information, check in time, and gate. If the identification id does not exist or is inputted wrong, an alert will tell the user of the case
- Airports
  - Here the user can see all the airports by code and see their name and what city and state they belong to.
- Planes
  - Here the user can see all the planes by code and see what model and how many total seats it has. The user is also able to insert new plane models but only if the plane model is not already there.

**ER diagram:**




## 5. Summary

Through out the web app there are several tables from the data base that can be reached. Each table is referred through a different tab. In each tab, the user is able to either search for information, see information, or update/add/delete information. At the bottom of the screen there is a show me button, this button allows the user to see the last ran sql query ran by the web app. That is a summary of what the web app can do and here are a few sql queries that are performed throughout the web app:

This is used to display the information in the boarding information tab

```
SELECT * FROM board_pass
JOIN personal ON board_pass.identification_id = personal.identification_id
JOIN passenger ON personal.identification_id = passenger.identification_id
UNION
SELECT * FROM board_pass JOIN personal ON board_pass.identification_id =
personal.identification_id
JOIN crew ON personal.identification_id = crew.identification_id
ORDER BY board_id;
```

This is a very basic query to display the airports

```
SELECT * FROM airports ORDER BY state, airport_id;
```

To search for a flight, a temporary table is used to hold the information and a transaction is used to update and retrieve that information

```
BEGIN TRANSACTION;
DELETE FROM hold3;
INSERT INTO hold3
SELECT flights.flight_id, departured, departure, arrivald, arrival, from_airport,
to_airport,
flights.plane_id, food_and_bev, movies, wifi, support_no, refueling, maintenance,
cleaning, baggage
FROM flights
JOIN extras ON flights.flight_id = extras.flight_id
JOIN support ON flights.flight_id = support.flight_id
WHERE flights.flight_id = 100001;
SELECT * FROM hold3;
DELETE FROM hold3;
COMMIT;
```

Here is a transaction to update some information

```
BEGIN TRANSACTION;
UPDATE extras SET food_and_bev = "Snacks and Water",
movies = "YES",
wifi = "NO"
WHERE flight_id = 100001;
COMMIT;
```