Airline Database Management System

CS 340 Assignment 2 Deadline: 20th Oct

The purpose of this assignment is twofold, to enable you to design a database depending on use cases concerned with the administrative management within an airline, and linking the database to a user-friendly program.

Setting up **[0]**

MySQL is an open-source relational database management system. It enables you to set up a database instance on your own machine. The following setup instructions are for Ubuntu.

Install MySQL

```
$ sudo apt-get update
$ sudo apt-get install mysql-server
```

Install MySQL Connector

```
$ sudo pip3 install mysql-connector
```

Test MySQL Connector

```
# test.py
import mysql.connector

mydb = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd="{yourpassword}"
)

print(mydb)
```

Your password is the one you set at the time of MySQL installation.

Database Design **[15]**

You have to design a database schema to cater to the use cases that follow. The schema should be at least in third normal form (3NF)[1]. The submission for this part of the assignment will be in the form of an Entity-Relationship Diagram[2]. Draw IO is recommended.

Database Setup **[15]**

Now, you will need to implement the designed schema into MySQL. Apart from the Python program, you can access your database using the terminal as well.

```
$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 5.7.27-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql>
```

Here, you can execute SQL statements directly.

```
mysql> CREATE DATABASE <databasename>;
```

Use this to create your database and the tables for that database.

Database Population **[10]**

Once all the tables are created, populate them with tuples for each table. You can execute the required SQL statements in a similar manner to the above part.

[1] https://www.tutorialspoint.com/sql/third-normal-form.htm
[2] https://www.tutorialspoint.com/dbms/er_diagram_representation.htm

To get full credit, each table must have more than ten tuples. A helpful hint for this part of the assignment is to write the required statements in a .sql file and then import it to MySQL.

```
$ mysql -u root -p <databasename> < populate.sql
```

Linking to frontend **[60]**

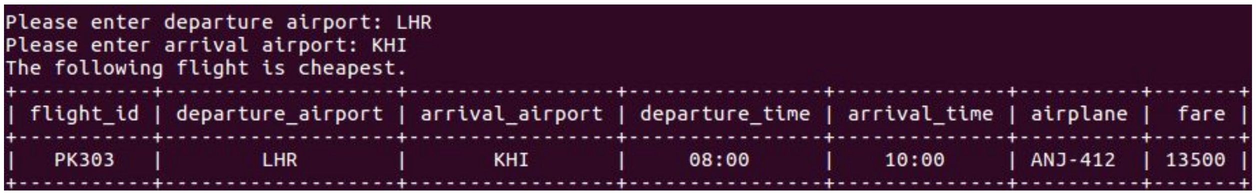
In this part of the assignment, you will implement a command line interface (CLI) to emulate an airline database management system in Python 3 (not 2.7). The following use cases need to be catered, with logical screen flows.

1. The program will show a welcome screen. **[1]**
2. The program will enable the receptionist or administrator to login, with distinct credentials. **[1]**
3. The receptionist will be able to;
    a. Create a new passenger record, with the required personal details. **[2]**
    b. Update details of an existing passenger record. **[5]**
    c. Using departure airport IATA code and arrival airport IATA code, view all available flights in a particular time period. **[5]**
    d. Generate ticket record for a particular passenger for a particular flight. **[5]**
    e. Using departure airport IATA code and arrival airport IATA code, view the cheapest flight. **[5]**
    f. View flight history of a particular passenger. **[5]**
    g. Cancel a particular ticket record. **[2]**
4. The administrator will be able to;
    a. Add a new flight record, with the required details. **[2]**
    b. Update details of an existing flight record. **[5]**
    c. Cancel a particular flight record. **[2]**
    d. View all flights landing and taking off for a particular airport on that day. **[5]**
    e. View every table of the database in tabular form. **[15]**

Make sure you handle **all** possible user errors, such as type and format errors. For example, if prompted for passenger phone number, anything other than an 11 digit numeric input should generate an error. The program should not crash at any instant.

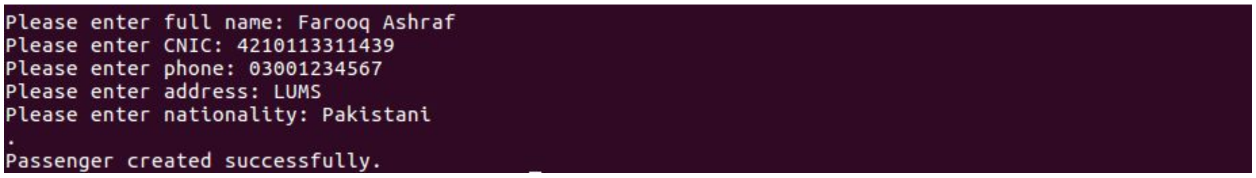
Sample Outputs

For example, use case 4a will ask the user to input departure and arrival airports. The output will be similar to the following.



```
Please enter departure airport: LHR
Please enter arrival airport: KHI
The following flight is cheapest.
+-----------+-------------------+-----------------+----------------+--------------+----------+-------+
| flight_id | departure_airport | arrival_airport | departure_time | arrival_time | airplane |  fare |
+-----------+-------------------+-----------------+----------------+--------------+----------+-------+
|   PK303   |        LHR        |       KHI       |     08:00      |    10:00     | ANJ-412  | 13500 |
+-----------+-------------------+-----------------+----------------+--------------+----------+-------+
```

Similarly, the interface handling use case 3a would be like the following.



```
Please enter full name: Farooq Ashraf
Please enter CNIC: 4210113311439
Please enter phone: 03001234567
Please enter address: LUMS
Please enter nationality: Pakistani
.
Passenger created successfully.
```

Bonus **[10]**

Use a Python CLI library such as PyInquirer[3] to write your program.

Submission Guidelines

The following is the recommended directory structure for the assignment.

```
|-- 21100XYZ
    |-- adms.py
    |-- er_diagram.pdf
```

Zip this folder into `2110XYZ.zip` and submit on LMS. No late submissions will be accepted.

[3] https://github.com/CITGuru/PyInquirer