Assignment no. 1

Emine Cambel

February 1, 2016

1 Problem Statement

Airport(Code, City, Country, Latitude, Longitude)

Flight(**Airline**, FlightNumber, **Origin**, **Destination**)

Schedule(**Airline**, **FlightNumber**, Date, DepartureTime, ArrivalTime)

Airline(Name, Abbreviation, HomeCountry, WebSite)

Ticket(Number, **Airline**, **FlightNumber**, **Date**, Cost)

In the above schema, note the following conventions:

- The primary key of each relation is underlined.
- Foreign keys are shown in **boldface**. Specifically:
  - The attribute **Airline** in the Flight relation is a foreign key which references the Airline relation.
  - The attributes **Origin** and **Destination** in the **Flight** relation are each foreign keys which reference the Airport relation.
  - The attribute pair **(Airline, FlightNumber)** in the Schedule relation is a foreign key which references the **Flight** relation.
  - The attribute triple **(Airline, FlightNumber, Date)** in the Ticket relation is a foreign key which references the **Schedule** relation.

2 Dissolved Queries

1. Find the code, latitude, and longitude of each airport which located in a city named Paris or Berlin.

   The **SELECT** statement is used to select data (here: Code, Latitude, Longitude) **FROM** database (Airport) with the Condition (each airport which is located in a city named Paris or Berlin).
   The **WHERE** clause is used to extract only those records that fulfill a specified criterion (here: City='Paris' OR City='Berlin'). **A.X** is a function that describes a choosen column X in table A (here: Type Aiport)

```
SELECT A.Code, A.Latitude, A.Longitude
FROM Airport A
WHERE A.City=’Paris’ OR A.City=’Berlin’;
```

2. Find the names of those airlines which have either a flight with origin the airport with code TXL or a flight with origin the airport with code SXF.

   The **DISTINCT** keyword is used to return only distinct (different) values. We have a look at the Flight table where we can get the origin airport of each flight. This origin flight we compare with the code ’TXL’ or ’SXF’. We use the **JOIN** clause to combine the rows from two or more tables. This is based on a common field between them.
   We keep in mind, that the attribute Airline in the Flight relation is a foreign key which references the Airline relation. The primary key in Airline table is Abbreviation. So because of that we notice that the Airline column in the Flight table refers to the Abbreviation in the Airline table. In order to this we can match (**JOIN.. ON..**) the Abbreviation column in Airline with the Airline column in Flight. We join on this in the Airline table so that we can **SELECT** the names of the Airline table that fulfill the conditions.

   ```
   SELECT DISTINCT A.Name
   FROM Flight F
   JOIN Airline A
   ON F.Airline = A.Abbreviation
   WHERE F.Origin = ’TXL’ OR F.Origin = ’SXF’;
   ```

3. Find the names of those airlines which have both a flight with origin the airport with code TXL and a flight with origin the airport with code SXF. (Hint: This may be done without using embedded subqueries by using the INTERSECT directive. Since MySQL does not support this directive, it is sufficient to give a solution which works only with PostgreSQL.)

   ```
   SELECT Name
   FROM
   WHERE Code=’TXL’ AND Code=’SXF’
   ```

4. Find the names of all airlines which have a scheduled flight on the date 2016-11-12.

   This is the same kind of querie like in 2, just with different information.

   ```
   SELECT DISTINCT A.Name
   FROM Schedule S
   JOIN Airline A
   ON S.Airline=A.Abbreviation
   WHERE S.Date=’2016-11-12’;
   ```

5. Find the airport code, city, and country for all airports which have a departure for which some ticket with ticket number less than 200 costs more than 4000.

   ```
   SELECT DISTINCT Code, Country, City FROM Schedule S, Ticket
   T JOIN Airport A ON WHERE DepartureTime IS NOT NUll AND
   T.Number¡200 AND T.Cost¿4000
   ```

6. Find the names of those airlines which have a flight whose destination is an airport which is located in Germany or France

   ```
   SELECT A.Name
   FROM Airline A
   WHERE
   ```

7. Find the names of those airlines with either the string “Air” or else the string “Luft” (both case insensitive) in their names.

   The **LIKE** operator is used to search for a specified pattern in a column.

   ```
       SELECT A.Name
       FROM Airline A
       WHERE A.NAME LIKE '%Luft%' OR A.NAME LIKE '%Air%';
   ```

8. Reduce the price of all tickets issued by Scandinavian airlines for the interval 2016-11-01 to 2016-11-20 inclusive by 20%. (Note: You may use the code “SK” in your query. It is not necessary to pattern match on “Scandinavian”)

   ```
   SELECT
   FROM
   WHERE
   ```

9. Find the codes of those airports located in Berlin, Germany which do not have any scheduled departures.

   ```
   SELECT Code
   FROM Schedule S
   JOIN Airport A ON WHERE S.DepartureTime is Null
   ```

10. Find the flights for the airline with code SK which are scheduled on 2016-11-12 with a departure time before noon (12:00). In addition to the flight number, give the airport codes for both the origin and the destination.

    ```
    SELECT
    FROM
    WHERE
    ```