Homework #3

Instructions:

--You will essentially copy your queries (and output, when requested) into this Word document and then submit the document in Canvas

--SUBMIT ALL QUERIES USING THE EXAMPLE FORMATTING BELOW…

Example Question:

Select "bmi" and "age" for 5 records. Order the output from highest to lowest "bmi." Include your output.

Answer:

Query

```
SELECT    bmi,
          age
FROM      health
ORDER BY  bmi DESC
LIMIT     5;
```

Output

```
bmi       age
6388.49   48
5858.59   54
4324.40   27
4320.96   48
3745.48   66
```

--Notes about the formatting:

- --Use Courier New font (because it is fixed-width)
- --Put each SQL clause on a new line
- --Use all caps for all SQL clauses and keywords
- --Write each field in the SELECT statement on a new line
- --Use tabs to clearly separate SQL clauses from field names, table names, etc.
- --You will need to manually type field names related to your output

--***IMPORTANT: As discussed in class, do not extract, copy, move, share, or take screenshots of any of the data in the database. The only query results that should leave MySQL Workbench are those included in your homework submission. Use copy/paste to directly move your results from MySQL Workbench into this Word document. If I, or the IT department, detect any unauthorized access or usage you will automatically receive an F for the course. Please ask if you are not sure if a specific use is authorized.***

--In general, the assignment will be graded for completeness. However, I reserve the right to grade a question or two for correctness.

Hints:

--All field names and the table name are case-sensitive for the sanford database

--In most cases there is no such thing as a single "right" answer. If two different queries generate the same desired output then both are acceptable.
--To copy output directly from the results window in MySQL Workbench, it is typically easiest to right-click and choose the "tab separated" option

Name: KEY

1.) Confirm that every record in the airline_ontime.ontime table has a matching record in the airline_ontime.airports table. Do this by running two queries. First, count the number of records in the airline_ontime.ontime table. Next, join the airline_ontime.ontime table with the airline_ontime.airports table and count the number of records in that result set. Perform the join on the "Origin" airport code and the corresponding field in the airports table. Documentation for the airports table can be found here: https://openflights.org/data.html. (Be sure to use the correct join. One type of join will automatically yield the same result as in your first query—you do not want to run that join because it is uninformative in this situation.)

Query

```
SELECT          COUNT(*)
FROM            ontime;

SELECT          COUNT(*)
FROM            ontime
INNER JOIN      airports
ON              ontime.Origin = airports.IATA;
```

Output

```
COUNT(*)
7453215

COUNT(*)
7453215
```

2.) Provide a count ("Total_Rec") of the number of non-cancelled, non-diverted flights departing from each airport outside of the United States. Include country name and airport name in your output. Order your output from highest to lowest "Total_Rec."

Query

```
SELECT          airports.Country,
                airports.Name,
                COUNT(*) AS Total_Rec
FROM            ontime
```

```
INNER JOIN     airports
ON             ontime.Origin = airports.IATA
WHERE          airports.Country != 'United States' AND
               ontime.Cancelled != 1 AND
               ontime.Diverted != 1
GROUP BY       airports.Country,
               airports.Name
ORDER BY       Total_Records DESC;
```

Output

```
Country        Name                                    Total_Rec
Puerto Rico    Luis Munoz Marin International Airport  23332
Virgin Islands Cyril E. King Airport                   2913
Puerto Rico    Rafael Hernandez Airport                1348
Puerto Rico    Mercedita Airport                       986
Virgin Islands Henry E Rohlsen Airport                 470
```

3.) What are the 10 most common flights that originate outside of the 'United States'? Ignore all cancelled and diverted flights. Your output should include the name of the airport where the flight departs, the name of the airport where the flight lands, and the total count of flight records for each departure/arrival pair. Drop the string ' Airport' from your departaure and arrival airport names to conserve space in your output.

Query

```
SELECT         TRIM(TRAILING ' Airport' FROM a1.Name) AS
               Depart_Name,
               TRIM(TRAILING ' Airport' FROM a2.Name) AS
               Arrive_Name,
               COUNT(*) AS Total_Flights
FROM           ontime
INNER JOIN     airports AS a1
ON             ontime.Origin = a1.IATA
INNER JOIN     airports AS a2
ON             ontime.Dest = a2.IATA
WHERE          a1.Country != 'United States' AND
               ontime.Cancelled != 1 AND
               ontime.Diverted != 1
GROUP BY       Depart_Name,
               Arrive_Name
ORDER BY       Total_Flights DESC
LIMIT          10;
```

Output

```
Depart_Name                    Arrive_Name                    Total_Flights
Luis Munoz Marin International John F Kennedy International          3895
```

```
Luis Munoz Marin International   Orlando International                          2607
Luis Munoz Marin International   Miami International                            2566
Luis Munoz Marin International   Newark Liberty International                   1862
Luis Munoz Marin International   Chicago O'Hare International                   1538
Luis Munoz Marin International   Philadelphia International                     1435
Luis Munoz Marin International   General Edward Lawrence Logan International    1361
Luis Munoz Marin International   Hartsfield Jackson Atlanta International       1262
Luis Munoz Marin International   Fort Lauderdale Hollywood International        1081
Luis Munoz Marin International   Dallas Fort Worth International                1042
```

4.) Count the total number of flight records that both depart and arrive in the 'United States' where the altitude change between departure and arrival locations is at least 3000 feet. Ignore all records that are cancelled or diverted.

Query

```
SELECT          COUNT(*) AS Total_Records
FROM            ontime
INNER JOIN      airports AS a1
ON              ontime.Origin = a1.IATA
INNER JOIN      airports AS a2
ON              ontime.Dest = a2.IATA
WHERE           a1.Country = 'United States' AND
                a2.Country = 'United States' AND
                ontime.Cancelled != 1 AND
                ontime.Diverted != 1 AND
                ABS(a1.Altitude - a2.Altitude) >= 3000;
```

Output

```
Total_Records
728123
```

5.) For the next few questions we'll focus on time zones spanning the "lower 48" (i.e. the United States excluding Alaska and Hawaii). Our ultimate goal is to determine the total number of flights that span at least three time zones. But, we'll break the problem into parts. First, we want to consider the following time zone values from our "airports" table:

--America/New_York (Eastern)
--America/Chicago (Central)
--America/Denver and America/Phoenix (Mountain)
--America/Los_Angeles (Pacific)

Moving from east to west, write a CASE statement to assign the numbers 1 to 4 to each time zone group.

Query

```
CASE
WHEN Timezone = 'America/New_York' THEN 1
```

```
WHEN Timezone = 'America/Chicago' THEN 2
WHEN Timezone IN ('America/Denver','America/Phoenix') THEN
     3
WHEN Timezone = 'America/Los_Angeles' THEN 4
ELSE NULL
END AS TZ_Num
```

6.) Next, we'll apply our CASE statement from the previous problem to assign each origin and destination airport a TZ_Num. Only consider flights that depart and arrive in the United States. Ignore cancelled and diverted flights. (So, this means each flight record will have two time zone group numbers.) Run the query, but you do not need to include your output.

Query

```
SELECT          CASE
                WHEN a1.Timezone = 'America/New_York' THEN 1
                WHEN a1.Timezone = 'America/Chicago' THEN 2
                WHEN a1.Timezone IN
                ('America/Denver','America/Phoenix') THEN 3
                WHEN a1.Timezone = 'America/Los_Angeles'
                THEN 4
                ELSE NULL
                END AS TZ_Dep_Num,
                CASE
                WHEN a2.Timezone = 'America/New_York' THEN 1
                WHEN a2.Timezone = 'America/Chicago' THEN 2
                WHEN a2.Timezone IN
                ('America/Denver','America/Phoenix') THEN 3
                WHEN a2.Timezone = 'America/Los_Angeles'
                THEN 4
                ELSE NULL
                END AS TZ_Arr_Num
FROM            ontime
INNER JOIN      airports AS a1
ON              ontime.Origin = a1.IATA
INNER JOIN      airports AS a2
ON              ontime.Dest = a2.IATA
WHERE           a1.Country = 'United States' AND
                a2.Country = 'United States' AND
                ontime.Cancelled != 1 AND
                ontime.Diverted != 1;
```

7.) Utilize the previous query (with a new CASE statement) to determine the total number of flights that span at least three time zones.

Query

```
SELECT    CASE
          WHEN ABS(TZ_Dep_Num - TZ_Arr_Num) >= 2 THEN 'Yes'
          WHEN ABS(TZ_Dep_Num - TZ_Arr_Num) < 2 THEN 'No'
          END AS Three_TZ_Flight,
          COUNT(*)
FROM (
     SELECT         CASE
                    WHEN a1.Timezone = 'America/New_York' THEN 1
                    WHEN a1.Timezone = 'America/Chicago' THEN 2
                    WHEN a1.Timezone IN
                    ('America/Denver','America/Phoenix') THEN 3
                    WHEN a1.Timezone = 'America/Los_Angeles'
                    THEN 4
                    ELSE NULL
                    END AS TZ_Dep_Num,
                    CASE
                    WHEN a2.Timezone = 'America/New_York' THEN 1
                    WHEN a2.Timezone = 'America/Chicago' THEN 2
                    WHEN a2.Timezone IN
                    ('America/Denver','America/Phoenix') THEN 3
                    WHEN a2.Timezone = 'America/Los_Angeles'
                    THEN 4
                    ELSE NULL
                    END AS TZ_Arr_Num
     FROM           ontime
     INNER JOIN     airports AS a1
     ON             ontime.Origin = a1.IATA
     INNER JOIN     airports AS a2
     ON             ontime.Dest = a2.IATA
     WHERE          a1.Country = 'United States' AND
                    a2.Country = 'United States' AND
                    ontime.Cancelled != 1 AND
                    ontime.Diverted != 1
     ) AS SubQ
GROUP BY
     Three_TZ_Flight;
```

Output

Three_TZ_Flight	COUNT(*)
NULL	217953
No	6135200
Yes	864901

8.) (This one is pretty difficult. Think about what your output needs to look like and the steps you need to get there.) For each carrier, determine the average distance that each of

its planes flies each day. Only include planes that have a tail number beginning with 'N'. IMPORTANT: only consider plane-days where the plane flies for exactly one carrier in that day. (And do not exclude cases by "hard-coding" those instances.) Exclude all cancelled and diverted flight records. Your output should include the airline carrier code, the total number of records for each carrier code, and the average distance that each of its planes flies each day. HINTS—having two subqueries within a query is perfectly fine…so is having multiple JOIN conditions for a single JOIN statement.

Query

```
SELECT          UniqueCarrier,
                COUNT(*),
                AVG(Tot_Dist)
FROM
(
     SELECT          UniqueCarrier,
                     TailNum,
                     DATE(CONCAT(Year,'-',Month,'-
                     ',DayofMonth)) AS Date_Field
     FROM            ontime
     WHERE           Cancelled = 0 AND
                     Diverted = 0 AND
                     TailNum LIKE 'N%'
     GROUP BY        UniqueCarrier,
                     TailNum,
                     Date_Field
) AS SubQ1
INNER JOIN
(
     SELECT          TailNum,
                     DATE(CONCAT(Year,'-',Month,'-
                     ',DayofMonth)) AS Date_Field,
                     COUNT(DISTINCT(UniqueCarrier)) AS
                     Num_Car,
                     SUM(Distance) AS Tot_Dist
     FROM            ontime
     WHERE           Cancelled = 0 AND
                     Diverted = 0 AND
                     TailNum LIKE 'N%'
     GROUP BY        TailNum,
                     Date_Field
     HAVING          Num_Car = 1
) AS SubQ2
ON              SubQ1.TailNum = SubQ2.TailNum AND
                SubQ1.Date_Field = SubQ2.Date_Field
GROUP BY        UniqueCarrier
ORDER BY        COUNT(*) DESC
```

```
LIMIT          7;
```

Output

```
UniqueCarrier  COUNT(*)  AVG(Tot_Dist)
AA             187255    3500.6994
WN             173839    4189.6496
UA             130763    3959.0233
DL             124297    3554.1577
US             113695    3503.0464
CO             104591    3493.3098
NW             101901    3071.3040
```

Comments

I think that this is a very good problem because it employs a lot of the SQL that we have learned to this point. However, my answer is not necessarily the only or "best" answer—there are definitely other approaches. SubQ2 plane (TailNum) and date and calculates the total distance that each plane flies on a given day. It also calculates the total number of unique carriers that each plane is associated with on a given day—the HAVING clause subsequently filters any plane-days where a plane is associated with multiple carriers. SubQ2 is joined with a very similar subquery (SubQ1) in order to recapture the actual carrier associated with each plane-day. Since we already calculated the total distance that each plane flies on a given day, we can simply average those values for each carrier (this is what we are doing in the "outer" query). COUNT(*) in the outer query simply gives us the total number of plane-days for each carrier.

9.) Just for good measure, run a query to see if there are any flights with a distance less than 20 miles. Exclude cancelled and diverted flights. Include origin airport code, origin city, destination airport code, destination city, ActualElapsedTime, and distance traveled. LAZY NEW YORKERS!

Query

```
SELECT         Origin,
               a1.City AS City_Depart,
               Dest,
               a2.City AS City_Arrive,
               ActualElapsedTime,
               Distance
FROM           ontime
```

```
INNER JOIN     airports AS a1
ON             ontime.Origin = a1.IATA
INNER JOIN     airports AS a2
ON             ontime.Dest = a2.IATA
WHERE          Distance < 20 AND
               Cancelled = 0 AND
               Diverted = 0;
```

Output

Origin	City_Depart	Dest	City_Arrive	ActualElapsedTime	Distance
JFK	New York	LGA	New York	35	11
JFK	New York	LGA	New York	102	11
JFK	New York	LGA	New York	88	11
SFO	San Francisco	OAK	Oakland	43	11
JFK	New York	LGA	New York	47	11
JFK	New York	LGA	New York	95	11
JFK	New York	LGA	New York	72	11