**Project 1 - Airline Database**

ISTM 622 - 601

Group 1-3

Brian Newman, Bhavishya Tyagi, Chuchu Yao, Pooja Vaswani, Anjali Shukla

**Create Database:**

Create database AirlineStat;

Use AirlineStat;

**Create Table:**

CREATE TABLE airline(

OP\_UNIQUE\_CARRIER VARCHAR(10),

OP\_CARRIER\_AIRLINE\_ID INT(5),

OP\_CARRIER\_DESC VARCHAR(100),

PRIMARY KEY (OP\_CARRIER\_AIRLINE\_ID)

);

CREATE TABLE airport(

AIRPORT\_ID INT(5),

Airport\_Code VARCHAR(5),

Airport\_Desc VARCHAR(100),

PRIMARY KEY (Airport\_Code )

);

CREATE TABLE flight(

FL\_DATE DATE,

OP\_CARRIER\_AIRLINE\_ID INT(5),

OP\_CARRIER\_FL\_NUM INT(5),

**ORIGIN\_Airport\_Code VARCHAR(5),**

**DEST\_Airport\_Code VARCHAR(5),**

CANCELLED INT(1),

DIVERTED INT(1),

ACTUAL\_ELAPSED\_TIME INT(6),

AIR\_TIME INT(6),

FLIGHTS INT(1),

DISTANCE INT(10) ,

CARRIER\_DELAY INT(10) ,

WEATHER\_DELAY INT(10) ,

NAS\_DELAY INT(10) ,

SECURITY\_DELAY INT(10) ,

LATE\_AIRCRAFT\_DELAY INT(10),

PRIMARY KEY (FL\_DATE, OP\_CARRIER\_AIRLINE\_ID **,** OP\_CARRIER\_FL\_NUM, ORIGIN**\_**Airport\_Code, DEST**\_**Airport\_Code, ACTUAL\_ELAPSED\_TIME),

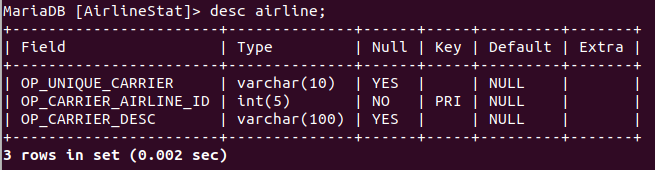
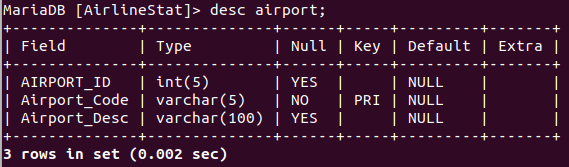
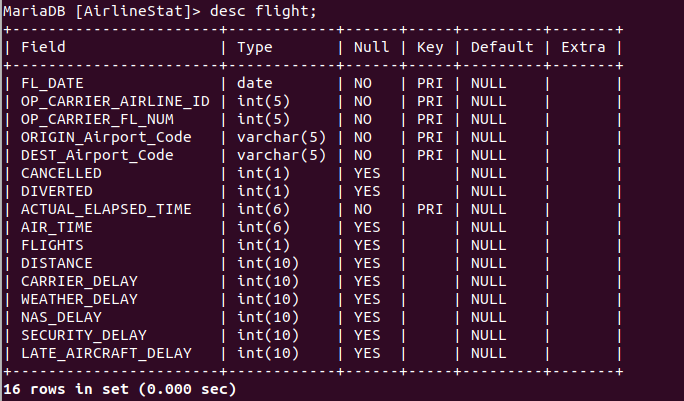
FOREIGN KEY(OP\_CARRIER\_AIRLINE\_ID ) REFERENCES airline(OP\_CARRIER\_AIRLINE\_ID ),

FOREIGN KEY(**ORIGIN\_**Airport\_Code) REFERENCES airport(Airport\_Code),

FOREIGN KEY(**DEST\_**Airport\_Code ) REFERENCES airport(Airport\_Code)

);

**Show Structure:**

* desc airline;
  + 
* desc airport;
  + 
* desc flight;
  + 

**Combining CSV:**

* This is because the files were split by month.

copy \*.csv combinedData.csv

* This is command prompt on Windows.

**Transforming the Data:**

Powershell Script: Change Empty to 0

|  |
| --- |
| $inFilePath = "D:\Google Drive\\_Fall 2019\ISTM 622\Group1-3\Newdataset\flightcase.csv"  $outFilePath = "D:\Google Drive\\_Fall 2019\ISTM 622\Group1-3\Newdataset\flightcaseEDIT.csv"  Import-Csv $inFilePath | % {  if (-not $\_.CANCELLED )  { $\_.CANCELLED = 0 }  if (-not $\_.DIVERTED )  { $\_.DIVERTED = 0 }  if (-not $\_.ACTUAL\_ELAPSED\_TIME )  { $\_.ACTUAL\_ELAPSED\_TIME = 0 }  if (-not $\_.AIR\_TIME )  { $\_.AIR\_TIME = 0 }  if (-not $\_.FLIGHTS )  { $\_.FLIGHTS = 0 }  if (-not $\_.DISTANCE )  { $\_.DISTANCE = 0 }  if (-not $\_.CARRIER\_DELAY)  { $\_.CARRIER\_DELAY = 0 }  if (-not $\_.WEATHER\_DELAY)  { $\_.WEATHER\_DELAY = 0 }  if (-not $\_.NAS\_DELAY)  { $\_.NAS\_DELAY = 0 }  if (-not $\_.SECURITY\_DELAY)  { $\_.SECURITY\_DELAY = 0 }  if (-not $\_.LATE\_AIRCRAFT\_DELAY)  { $\_.LATE\_AIRCRAFT\_DELAY = 0 }  $\_ # echo all records, so they can be exported back to a file  } | Export-Csv $outFilePath -NoType |

**Loading the Data:**

LOAD DATA INFILE '/home/big/Desktop/dataset/airline.csv'

INTO TABLE airline

FIELDS TERMINATED BY ','

optionally enclosed BY '"'

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES;

LOAD DATA INFILE '/home/big/Desktop/dataset/airport.csv'

INTO TABLE airport

FIELDS TERMINATED BY ','

optionally enclosed BY '"'

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES;

LOAD DATA INFILE '/home/big/Desktop/dataset/Final17.csv'

INTO TABLE flight

FIELDS TERMINATED BY ','

optionally ENCLOSED BY '"'

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES;

**Loading the Data:**

LOAD DATA INFILE '/home/big/Desktop/dataset/Final18.csv'

INTO TABLE flight

FIELDS TERMINATED BY ','

optionally ENCLOSED BY '"'

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES;

LOAD DATA INFILE '/home/big/Desktop/dataset/Final19.csv'

INTO TABLE flight

FIELDS TERMINATED BY ','

optionally ENCLOSED BY '"'

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES;

**Connecting to Database with Python:**

* Run the commands on the cluster
  + sudo apt-get libmariadb-dev
  + CREATE USER 'root'@'10.20.0.167' IDENTIFIED BY '1234';
  + GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'10.20.0.167';

**HTML:**

<!DOCTYPE **html**>

<**html**>

<**head**>

<**title**>AirLine</**title**>

</**head**>

*<!-- Adding some style to table -->*

<**style type="text/css"**>

th:tr{color: blue;}

tr:nth-of-type(2n){border: 1px solid black;background-color: rgba(150, 150, 150, 0.5);}

td{padding: 8px 8px;border: 1px solid black;}

body{background-color: powderblue;text-align: center;color: black;font-family: Arial,Helvetica,

sans-serif;}

</**style**>

<**body**>

<**h1**>Airline Stats </**h1**>

<**h2**>Data is from January to June from 2017,2018,2019! </**h2**>

<**p**>This is a dashboard!</**p**>

<**table style="**margin-left: 20px;**"**>

*<!-- Table headers for Query1 -->*

<**th**>

<**tr style="**color: Black; font-weight: bold; **"**>

<**td**>Average flight delays</**td**>

</**tr**>

<**tr style="**color: Black; **"**>

<**td**>{{ query1 }} Minutes</**td**>

</**tr**>

</**th**>

</**table**>

<**table style="**margin-left: 20px;**"**>

*<!-- Table headers for Query2 -->*

<**th**>

<**tr style="**color: Black; font-weight: bold; **"**>

<**td**>Showing number of flights whose distance is greater than 1000 that were

cancelled.</**td**>

</**tr**>

<**tr style="**color: Black; **"**>

<**td**>{{ query2 }}</**td**>

</**tr**>

</**th**>

</**table**>

<**table style="**margin-left: 20px;**"**>

*<!-- Table headers for Query3 -->*

<**th**>

<**tr style="**color: Black; font-weight: bold; **"**>

<**td**>Showing number of flights whose distance is less than 1000 that were cancelled.</**td**>

</**tr**>

<**tr style="**color: Black; **"**>

<**td**>{{ query3 }}</**td**>

</**tr**>

</**th**>

</**table**>

<**table style="**margin-left: 20px;**"**>

*<!-- Table headers for Query4 -->*

<**th**>

<**tr style="**color: Black; font-weight: bold; **"**>

<**td colspan="2"**>Showing the number of cancelled flights from each location.</**td**>

</**tr**>

</**th**>

*<!-- For loop logic of jinja template -->*

{%for i in query4%}

*<!-- table rows -->*

<**tr**>

<**td**>{{i[0]}}</**td**>

<**td**>{{i[1]}}</**td**>

{%endfor%}

</**tr**>

</**table**>

<**table style="**margin-left: 20px;**"**>

<**p**></**p**>

*<!-- Table headers for Query5 -->*

<**th**>

<**tr style="**color: Black; font-weight: bold;**"**>

<**td colspan="5"**>Showing flight delayed by categories in minutes</**td**>

</**tr**>

<**tr style="**color: Black; font-weight: bold;**"**>

<**td**>Carrier Delay</**td**>

<**td**>Weather Delay</**td**>

<**td**>NAS Delay</**td**>

<**td**>Security Delay</**td**>

<**td**>Late Craft Delay</**td**>

</**tr**>

</**th**>

*<!-- For loop logic of jinja template -->*

{%for i in query5%}

*<!-- table rows -->*

<**tr**>

<**td**>{{i[0]}}</**td**>

<**td**>{{i[1]}}</**td**>

<**td**>{{i[2]}}</**td**>

<**td**>{{i[3]}}</**td**>

<**td**>{{i[4]}}</**td**>

{%endfor%}

</**tr**>

</**table**>

<**p**></**p**>

<**table style="**margin-left: 20px;**"**>

*<!-- Table headers for Query6 -->*

<**th**>

<**tr style="**color: Black; font-weight: bold;**"**>

<**td colspan="5"**>Showing flight delayed by categories in numbers</**td**>

</**tr**>

<**tr style="**color: Black; font-weight: bold;**"**>

<**td**>Carrier Delay</**td**>

<**td**>Weather Delay</**td**>

<**td**>NAS Delay</**td**>

<**td**>Security Delay</**td**>

<**td**>Late Craft Delay</**td**>

</**tr**>

</**th**>

<**tr**>

<**td**>{{query6}}</**td**>

<**td**>{{query7}}</**td**>

<**td**>{{query8}}</**td**>

<**td**>{{query9}}</**td**>

<**td**>{{query10}}</**td**>

</**tr**>

</**table**>

</**body**>

</**html**>

**Python Application and Queries:**

**from** flask **import** Flask, render\_template, request

**import** pusher

app = Flask(\_\_name\_\_)

channels\_client = pusher.Pusher(

app\_id=**'865352'**,

key=**'2929f012f8448e1e7297'**,

secret=**'3718786561642f32bace'**,

cluster=**'us2'**,

ssl=**True**

)

*##################################################################################################*

**import** mysql.connector **as** mariadb

mariadb\_connection = mariadb.connect(user=**'root'**, password=**'1234'**, host=**'10.20.0.167'**, database=**'AirlineStat'**)

cursor = mariadb\_connection.cursor()

cursor.execute(**"Select AVG(CARRIER\_DELAY+WEATHER\_DELAY+NAS\_DELAY+SECURITY\_DELAY+LATE\_AIRCRAFT\_DELAY ) from flight;"**)

query1 = cursor.fetchone()

query1 = query1[0]

print(query1)

cursor.execute(**"Select COUNT(CANCELLED) from flight where CANCELLED=1 and DISTANCE>1000;"**)

query2 = cursor.fetchone()

query2 = query2[0]

print(query2)

cursor.execute(**"Select COUNT(CANCELLED) from flight where CANCELLED=1 and DISTANCE<=1000;"**)

query3 = cursor.fetchone()

query3 = query3[0]

print(query3)

cursor.execute(**"Select ORIGIN\_Airport\_Code, COUNT(CANCELLED) FROM flight where CANCELLED=1 group by ORIGIN\_Airport\_Code;"**)

query4 = cursor.fetchall()

cursor.execute(**"Select SUM(CARRIER\_DELAY) as carrier\_delay, SUM(WEATHER\_DELAY) as weather\_delay,sum(NAS\_DELAY) as nas\_delay, "**

**"sum(SECURITY\_DELAY) as security\_delay,sum(LATE\_AIRCRAFT\_DELAY) as late\_craft\_delay From flight;"**)

query5 = cursor.fetchall()

cursor.execute(**"Select count(CARRIER\_DELAY) as carrier\_delay from flight where CARRIER\_DELAY >0;"**)

query6 = cursor.fetchone()

query6 = query6[0]

print(query6)

cursor.execute(**"Select count(WEATHER\_DELAY) as weather\_delay from flight where WEATHER\_DELAY >0;"**)

query7 = cursor.fetchone()

query7 = query7[0]

print(query7)

cursor.execute(**"select count(NAS\_DELAY) as nas\_delay from flight where NAS\_DELAY>0;"**)

query8 = cursor.fetchone()

query8 = query8[0]

print(query8)

cursor.execute(**"select count(SECURITY\_DELAY) as security\_delay from flight where SECURITY\_DELAY>0;"**)

query9 = cursor.fetchone()

query9 = query9[0]

print(query9)

cursor.execute(**"select count(LATE\_AIRCRAFT\_DELAY) as late\_aircraft\_delay from flight where LATE\_AIRCRAFT\_DELAY>0;"**)

query10 = cursor.fetchone()

query10 = query10[0]

print(query10)

*##################################################################################################*

@app.route(**'/'**)

**def** dashboard():

*# Display website*

*#len = Length of list and then ports over*

**return** render\_template(**'dashboard.html'**, query1 = query1, query2 = query2,

query3 = query3, len = len(query4), query4 = query4,

len1 = len(query5), query5 = query5, query6 = query6,

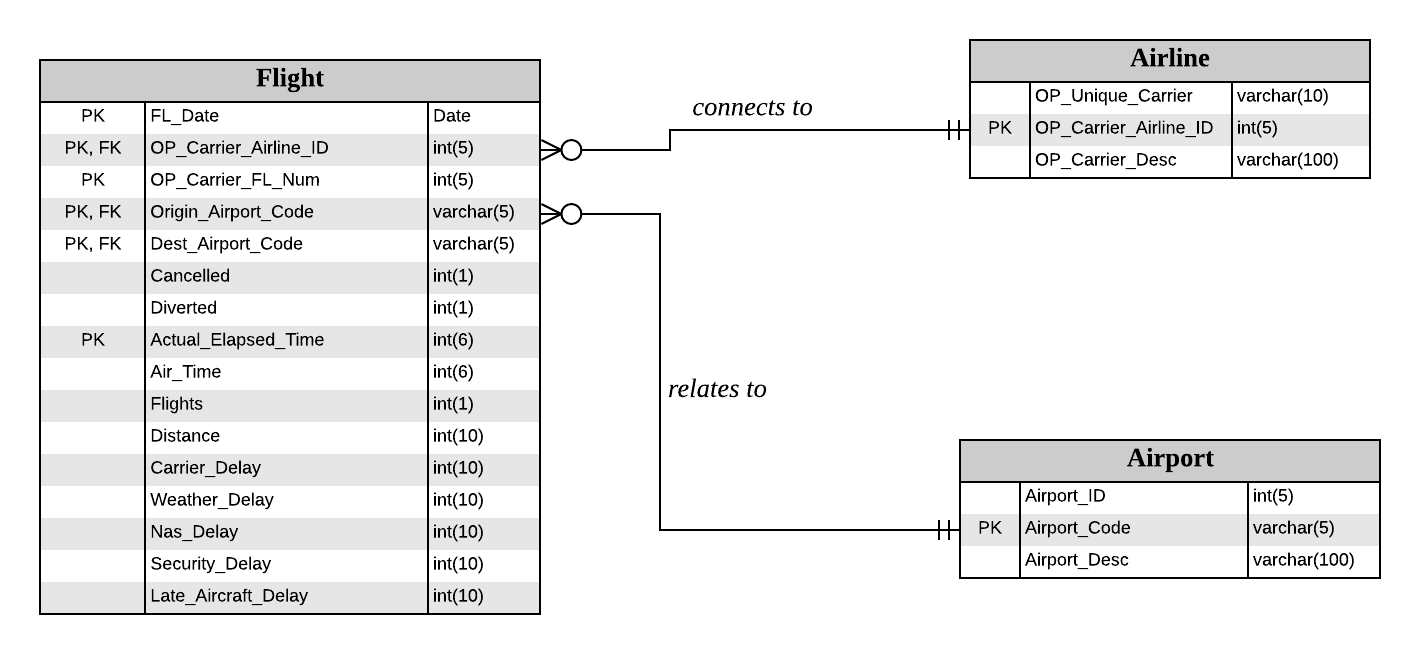
query7 = query7, query8 = query8, query9 = query9,

query10 = query10)

**if** \_\_name\_\_ == **'\_\_main\_\_'**:

app.run(debug=**False**)

**ERD:**



**Cluster and Part 1 Demo:**

* https://youtu.be/2XAHXM5ELnE