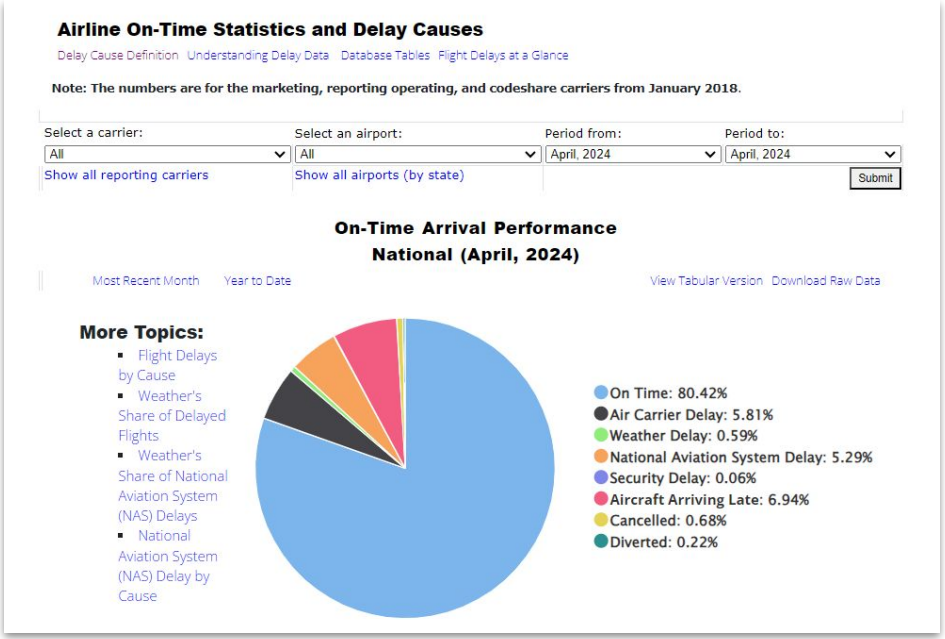
Leaving on a Jet Plane

Shannon Hoffman, Brittany Thomas Holtegaard, Elisabeth Jansen, Jose Vela

Introduction & Process

- Selected “data engineering track” to explore flight delay data
- Data sources:
  - Bureau of Transportation Statistics
  - Airport Latitude/Longitude Coordinates
- 22,569 data points
- Jan 1 - Dec 31, 2023



Introduction & Process

Air Carrier: The cause of the cancellation or delay was due to circumstances within the airline's control (e.g. maintenance or crew problems, aircraft cleaning, baggage loading, fueling, etc.).

Extreme Weather: Significant meteorological conditions (actual or forecasted) that, in the judgment of the carrier, delays or prevents the operation of a flight such as tornado, blizzard or hurricane.

National Aviation System (NAS): Delays and cancellations attributable to the national aviation system that refer to a broad set of conditions, such as non-extreme weather conditions, airport operations, heavy traffic volume, and air traffic control.

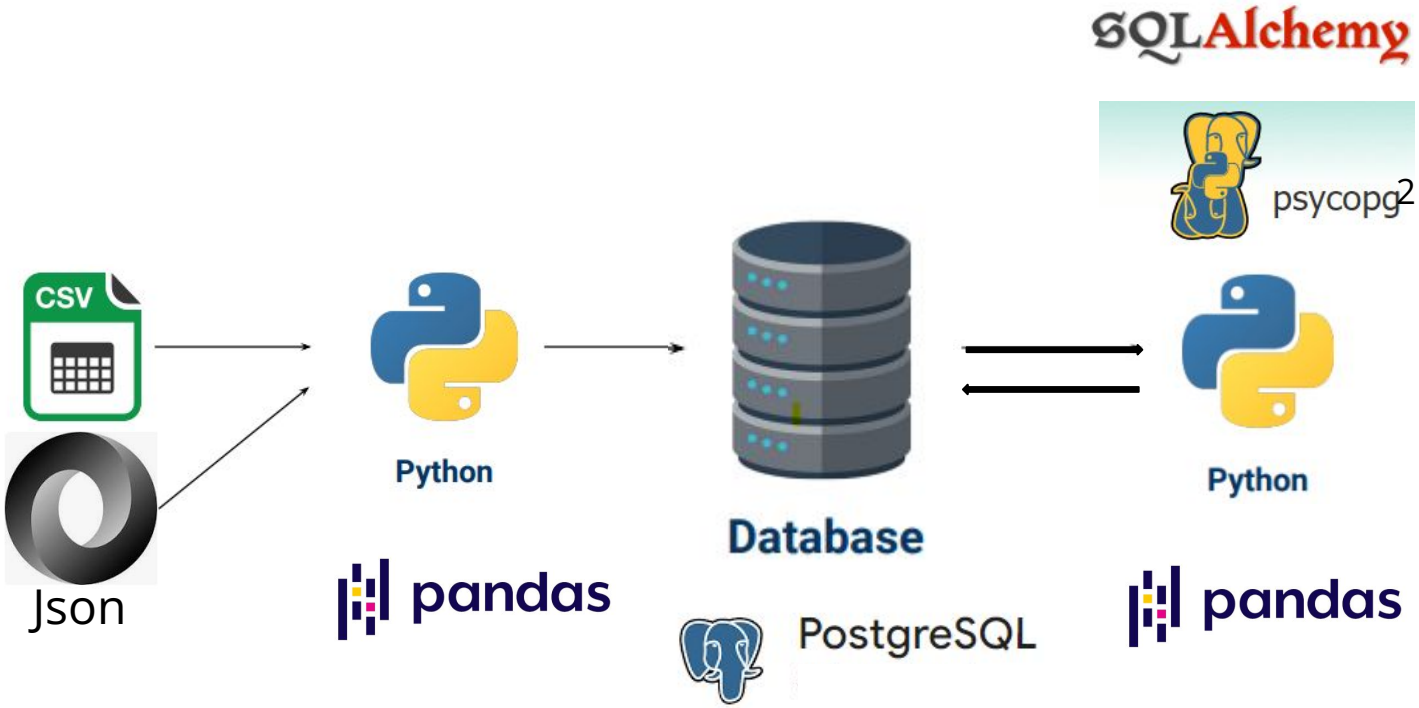
Late-arriving aircraft: A previous flight with same aircraft arrived late, causing the present flight to depart late.

Security: Delays or cancellations caused by evacuation of a terminal or concourse, re-boarding of aircraft because of security breach, inoperative screening equipment and/or long lines in excess of 29 minutes at screening areas.

Column Name	Column Definition
year	YYYY format
month	MM format (1-12)
carrier	Code assigned by assigned by US DOT to identify a unique airline carrier.
carrier_name	Unique airline (carrier) is defined as one holding and reporting under the same DOT certificate regardless of its Code, Name, or holding company/corporation.
airport	A three character alpha-numeric code issued by the U.S. Department of Transportation which is the official designation of the airport.
airport_name	a place from which aircraft operate that usually has paved runways and maintenance facilities and often serves as a terminal
arr_flights	Arrival Flights
arr_del15	Arrival Delay Indicator, 15 Minutes or More Arrival delay equals the difference of the actual arrival time minus the scheduled arrival time. A flight is considered on-time when it arrives less than 15 minutes after its published arrival time.
carrier_ct	Carrier Count for airline cause of delay
weather_ct	Weather Count for airline cause of delay
nas_ct	NAS (National Air System) Count for airline cause of delay
security_ct	Security County for airline cause of delay
late_aircraft_ct	Late Aircraft Delay Count for airline cause of delay
arr_cancelled	flight cancelled
arr_diverted	flight diverted
arr_delay	Difference in minutes between scheduled and actual arrival time. Early arrivals show negative numbers.
carrier_delay	Carrier Delay, in Minutes
weather_delay	Weather Delay, in Minutes
nas_delay	National Air System Delay, in Minutes
security_delay	Security Delay, in Minutes
late_aircraft_delay	Late Aircraft Delay, in Minutes

FROM:
https://www.bts.gov/explore-topics-and-geography/topics/airline-time-performance-and-causes-flight-delays#q0

Extract Transform Load (ETL)



Adapted from class slides with official software logos.

Python

pandas

```
[1]: # Dependencies
import pandas as pd
from pathlib import Path
```

```
[2]: # Use Pandas to the read data.
data_df = pd.read_csv("2023_data.csv")
data_df.head()
```

[2]:

	year	month	carrier	carrier_name	airport	airport_name	arr_flights	arr_del15	carrier_ct	weather_ct	...	security_ct	late_aircraft_ct	arr_cancelled	ar
0	2023	12	9E	Endeavor Air Inc.	ABE	Allentown/Bethlehem/Easton, PA: Lehigh Valley ...	72.0	5.0	2.46	1.00	...	0.0	0.81	0.0	
1	2023	12	9E	Endeavor Air Inc.	AEX	Alexandria, LA: Alexandria International	62.0	7.0	4.25	0.00	...	0.0	1.75	0.0	
2	2023	12	9E	Endeavor Air Inc.	AGS	Augusta, GA: Augusta Regional at Bush Field	95.0	10.0	5.94	0.00	...	0.0	3.00	0.0	
3	2023	12	9E	Endeavor Air Inc.	ALB	Albany, NY: Albany International	23.0	2.0	0.56	0.00	...	0.0	1.44	1.0	
4	2023	12	9E	Endeavor Air Inc.	ATL	Atlanta, GA: Hartsfield-Jackson Atlanta Intern...	2111.0	256.0	76.88	8.75	...	0.0	117.94	1.0	

5 rows × 21 columns

```
[3]: data_df.count()
```

[3]:

year	22621
month	22621
carrier	22621
carrier_name	22621
airport	22621
airport_name	22621
arr_flights	22574
arr_del15	22569
carrier_ct	22574
weather_ct	22574
nas_ct	22574
security_ct	22574
late_aircraft_ct	22574
arr_cancelled	22574
arr_diverted	22574
arr_delay	22574
carrier_delay	22574
weather_delay	22574
nas_delay	22574
security_delay	22574
late_aircraft_delay	22574

dtype: int64

city
Allentown/Bethlehem/Easton, PA
Alexandria, LA
Augusta, GA
Albany, NY
Atlanta, GA

```
# write cleaned data to new csv file
data_df.to_csv("2023_data_cleaned.csv", index=False, header=True)
```

2023_data_cleaned.csv will be imported into pgAdmin SQL database for further data exploration

Process: Cleaning

Process: Cleaning Jose

Collect data source that we could use with our flight data source for purpose of having airports plot on a map.

https://github.com/ip2location/ip2location-iata-icao/blob/master/iata-icao.csv

Then filter for by county_code.

	country_code	region_name	iata	icao	airport	latitude	longitude
0	AE	Abu Zaby	AAN	OMAL	Al Ain International Airport	24.2617	55.6092
1	AE	Abu Zaby	AUH	OMAA	Abu Dhabi International Airport	24.4330	54.6511
2	AE	Abu Zaby	AYM	NaN	Yas Island Seaplane Base	24.4670	54.6103
3	AE	Abu Zaby	AZI	OMAD	Al Bateen Executive Airport	24.4283	54.4581
4	AE	Abu Zaby	DHF	OMAM	Al Dhafra Air Base	24.2482	54.5477
...	...	...	...	...	...	...	...
8941	ZW	Masvingo	MVZ	FVMV	Masvingo Airport	-20.0553	30.8591
8942	ZW	Matabeleland North	HWN	FVWN	Hwange National Park Airport	-18.6299	27.0210
8943	ZW	Matabeleland North	VFA	FVFA	Victoria Falls Airport	-18.0959	25.8390
8944	ZW	Matabeleland North	WKI	FVWT	Hwange Town Airport	-18.3630	26.5198
8945	ZW	Midlands	GWE	FVTL	Thornhill Air Base	-19.4364	29.8619

	country_code	region_name	iata	icao	airport	latitude	longitude
6602	US	Alabama	AIV	KAIV	George Downer Airport	33.1065	-88.1978
6603	US	Alabama	ALX	KALX	Thomas C. Russell Field	32.9147	-85.9630
6604	US	Alabama	ANB	KANB	Anniston Regional Airport	33.5882	-85.8581
6605	US	Alabama	ASN	KASN	Talladega Municipal Airport	33.5699	-86.0509
6606	US	Alabama	AUO	KAUO	Auburn University Regional Airport	32.6151	-85.4340
...	...	...	...	...	...	...	...
8625	US	Wyoming	SAA	KSAA	Shively Field	41.4449	-106.8240
8626	US	Wyoming	SHR	KSHR	Sheridan County Airport	44.7692	-106.9800
8627	US	Wyoming	THP	KTHP	Hot Springs County-Thermopolis Municipal Airport	43.7136	-108.3900
8628	US	Wyoming	TOR	KTOR	Torrington Municipal Airport	42.0645	-104.1530
8629	US	Wyoming	WRL	KWRL	Worland Municipal Airport	43.9657	-107.9510

Process: Cleaning Jose

Reading in both csv to merge latitude and longitude to its aligning “airport”.

Collecting columns of latitude and longitude and iata (renamed to airport) for new database we will export to csv file to use for combining with main DB.

]:

	airport	latitude	longitude
6602	AIV	33.1065	-88.1978
6603	ALX	32.9147	-85.9630
6604	ANB	33.5882	-85.8581
6605	ASN	33.5699	-86.0509
6606	AUO	32.6151	-85.4340
...	...	...	...
8625	SAA	41.4449	-106.8240
8626	SHR	44.7692	-106.9800
8627	THP	43.7136	-108.3900
8628	TOR	42.0645	-104.1530
8629	WRL	43.9657	-107.9510

2028 rows × 3 columns

Updated DataFrame with longitude and latitude:

	month	carrier_name	airport	city	total_arrivals	total_delays_ct	carrier_ct	weather_ct	nat_air_sys_ct	security_ct	...	flight_cancelled	flight_diverted	total_delays_min	carrier_delay_min	weather_delay_min	nat_air_sys_delay_min	security_delay_min	late_aircraft_delay_min	longitude	latitude
0	12	Endeavor Air Inc.	ABE	Allentown/Bethlehem/Easton, PA	72.0	5.0	2.46	1.00	0.73	0.0	...	0.0	0.0	672.0	61.0	574.0	20.0	0.0	17.0	-84.374393	40.165883
1	12	Endeavor Air Inc.	AEX	Alexandria, LA	62.0	7.0	4.25	0.00	1.00	0.0	...	0.0	0.0	348.0	252.0	0.0	33.0	0.0	63.0	-92.540955	31.329274
2	12	Endeavor Air Inc.	AGS	Augusta, GA	95.0	10.0	5.94	0.00	1.06	0.0	...	0.0	0.0	859.0	536.0	0.0	47.0	0.0	276.0	-122.396008	37.779418
3	12	Endeavor Air Inc.	ALB	Albany, NY	23.0	2.0	0.56	0.00	0.00	0.0	...	1.0	0.0	75.0	9.0	0.0	0.0	0.0	66.0	-115.002136	55.001251
4	12	Endeavor Air Inc.	ATL	Atlanta, GA	2111.0	256.0	76.88	8.75	52.43	0.0	...	1.0	0.0	21424.0	8906.0	732.0	1487.0	0.0	10299.0	-84.429271	33.637799

[5 rows x 21 columns]

Process: Cleaning Jose

Export to csv for pg admin.

Build schema and import csv into table for pg admin.

Clean to only keep airport, longitude and latitude.

	airport	longitude	latitude
0	ABE	-84.374393	40.165883
1	AEX	-92.540955	31.329274
2	AGS	-122.396008	37.779418
3	ALB	-115.002136	55.001251
4	ATL	-84.429271	33.637799
...	...	...	...
340	COU	130.424506	33.713871
341	GUM	-120.521556	34.919978
342	SPN	116.627613	-8.304813
343	ACK	-97.620488	35.396246
344	HYA	81.538920	30.075730

345 rows × 3 columns

	month	carrier_name	airport	city	total_arrivals	total_delays_ct	carrier_ct	weather_ct	n
0	12	Endeavor Air Inc.	ABE	Allentown/Bethlehem/Easton, PA	72.0	5.0	2.46	1.00	
1	12	Endeavor Air Inc.	AEX	Alexandria, LA	62.0	7.0	4.25	0.00	
2	12	Endeavor Air Inc.	AGS	Augusta, GA	95.0	10.0	5.94	0.00	
3	12	Endeavor Air Inc.	ALB	Albany, NY	23.0	2.0	0.56	0.00	
4	12	Endeavor Air Inc.	ATL	Atlanta, GA	2111.0	256.0	76.88	8.75	
...	...	...	...	...	...	...	...	...	...
22564	1	Air Wisconsin Airlines Corp	STL	St. Louis, MO	2.0	1.0	0.00	1.00	
22565	1	Air Wisconsin Airlines Corp	SYR	Syracuse, NY	31.0	6.0	3.49	0.00	
22566	1	Air Wisconsin Airlines Corp	TUL	Tulsa, OK	27.0	6.0	1.94	0.00	
22567	1	Air Wisconsin Airlines Corp	TVC	Traverse City, MI	62.0	17.0	2.65	0.00	
22568	1	Air Wisconsin Airlines Corp	TYS	Knoxville, TN	62.0	12.0	5.80	0.77	

22569 rows × 19 columns

```
drop table if exists locations

create table locations(
    airport varchar (10) not null,
    latitude float not null,
    longitude float not null
)
select * from locations
```

Data Output | Messages | Notifications

SQL

	airport character varying (10)	latitude double precision	longitude double precision
1	AIV	33.1065	-88.1978
2	ALX	32.9147	-85.963
3	ANB	33.5882	-85.8581
4	ASN	33.5699	-86.0509
5	AUO	32.6151	-85.434
6	BFM	30.6268	-88.0681
7	BHM	33.5629	-86.7535
8	DCU	34.6527	-86.9454
9	DHN	31.3213	-85.4496
10	ETS	31.2997	-85.8999

Total rows: 1000 of 2028 Query complete 00:00:00.776

[2]:
```
from sqlalchemy import create_engine
import pandas as pd
```

[6]:
```
# Load the existing CSV file
dirty_plots_df = pd.read_csv('long_lat_test.csv')
dirty_plots_df.head()
```

[6]:

	month	carrier_name	airport	city	total_arrivals	total_delays_ct	carrier_ct	weather_ct	nat_a
0	12	Endeavor Air Inc.	ABE	Allentown/Bethlehem/Easton, PA	72.0	5.0	2.46	1.00	
1	12	Endeavor Air Inc.	AEX	Alexandria, LA	62.0	7.0	4.25	0.00	
2	12	Endeavor Air Inc.	AGS	Augusta, GA	95.0	10.0	5.94	0.00	
3	12	Endeavor Air Inc.	ALB	Albany, NY	23.0	2.0	0.56	0.00	
4	12	Endeavor Air Inc.	ATL	Atlanta, GA	2111.0	256.0	76.88	8.75	

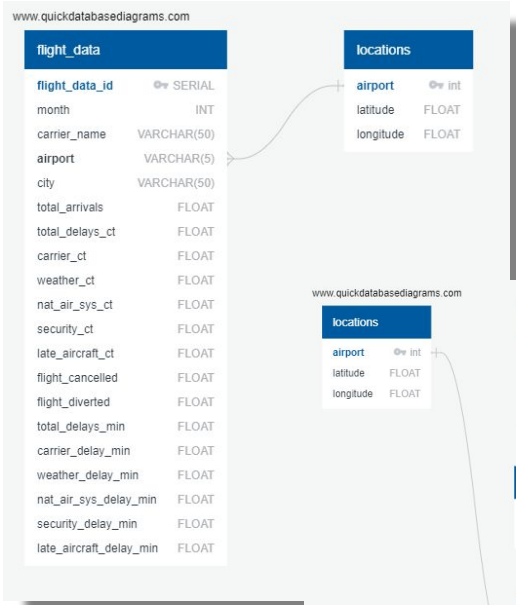
5 rows × 21 columns

[7]:
```
clean_plots_df = dirty_plots_df[['airport', 'longitude', 'latitude']]
clean_plots_df.head()
```

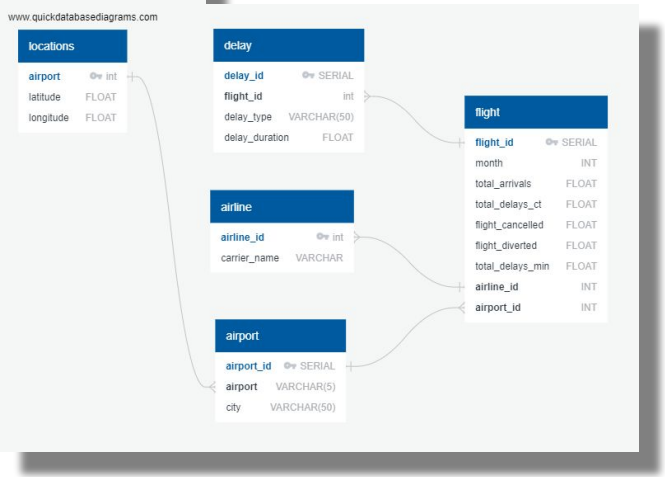
[7]:

	airport	longitude	latitude
0	ABE	-84.374393	40.165883
1	AEX	-92.540955	31.329274
2	AGS	-122.396008	37.779418
3	ALB	-115.002136	55.001251
4	ATL	-84.429271	33.637799

Process: Database Creation



ERD



Schema & PostgreSQL/ pgAdmin

```sql
DROP TABLE IF EXISTS flight_data;
DROP TABLE IF EXISTS locations;

CREATE TABLE flight_data (
    flight_data_id SERIAL PRIMARY KEY,
    month INT,
    carrier_name VARCHAR(50),
    airport VARCHAR(5),
    city VARCHAR(50),
    total_arrivals FLOAT,
    total_delays_ct FLOAT,
    carrier_ct FLOAT,
    weather_ct FLOAT,
    nat_air_sys_ct FLOAT,
    security_ct FLOAT,
    late_aircraft_ct FLOAT,
    flight_cancelled FLOAT,
    flight_diverted FLOAT,
    total_delays_min FLOAT,
    carrier_delay_min FLOAT,
    weather_delay_min FLOAT,
    nat_air_sys_delay_min FLOAT,
    security_delay_min FLOAT,
    late_aircraft_delay_min FLOAT
);

CREATE TABLE locations (
    airport   VARCHAR(3) PRIMARY KEY,
    latitude FLOAT (50),
    longitude FLOAT (50)
);
```

Psycopg2 Library

Psycopg2 allows for postgres database connection through creating an engine utilizing host, port, database name, username, and password.

```
# Define connection parameters
host = '127.0.0.1'
port = '5432'  # default PostgreSQL port
database = 'project3'
user = 'postgres'
password = '[redacted]'

# Create the connection string
connection_string = f'postgresql+psycopg2://{user}:{password}@{host}:{port}/{database}'

# Create the database engine
engine = create_engine(connection_string)
```

```
cur = conn.cursor()
cur.execute("UPDATE Employee set EMAI = 'updated@gmail.com' WHERE ID =1 ")
conn.commit()
```

```
# Define SQL query for full table
query = 'SELECT * FROM flight_data'
# Read the data into a pandas DataFrame
df = pd.read_sql(query, engine)
# Display the DataFrame
df.head()
```

SQL queries can then be carried out using pandas read_sql function or directly through psycopg2 by creating a cursor and using the execute function.

Map Creation: Jose

Using cleaned data, we are able to convert csv file to geojson format.

(https://leafletjs.com/examples/geojson/) for documentation.

```geometry_plots.js > US_data > "features" > "geometry"
var US_data = {
  "type": "FeatureCollection",
  "features": [
  {
    "type": "Feature",
    "geometry": {
      "type": "Point",
      "coordinates":  [ -84.3743929,40.1658829 ]
    },
    "properties": {
    "airport":"ABE",
    "total_delays_ct":5,
    }
  },
  {
    "type": "Feature",
    "geometry": {
      "type": "Point",
      "coordinates":  [ -92.54095486627727,31.32927365 ]
    },
    "properties": {
    "airport":"AEX",
    "total_delays_ct":7,
    }
  },
  {
    "type": "Feature",
    "geometry": {
      "type": "Point",
      "coordinates":  [ -122.3960076,37.7794178 ]
    },
    "properties": {
    "airport":"AGS",
    "total_delays_ct":10,
    }
  },
  {
    "type": "Feature",
```

Ln 601, Col 24 Spaces: 3 UTF-8 CRLF JavaS

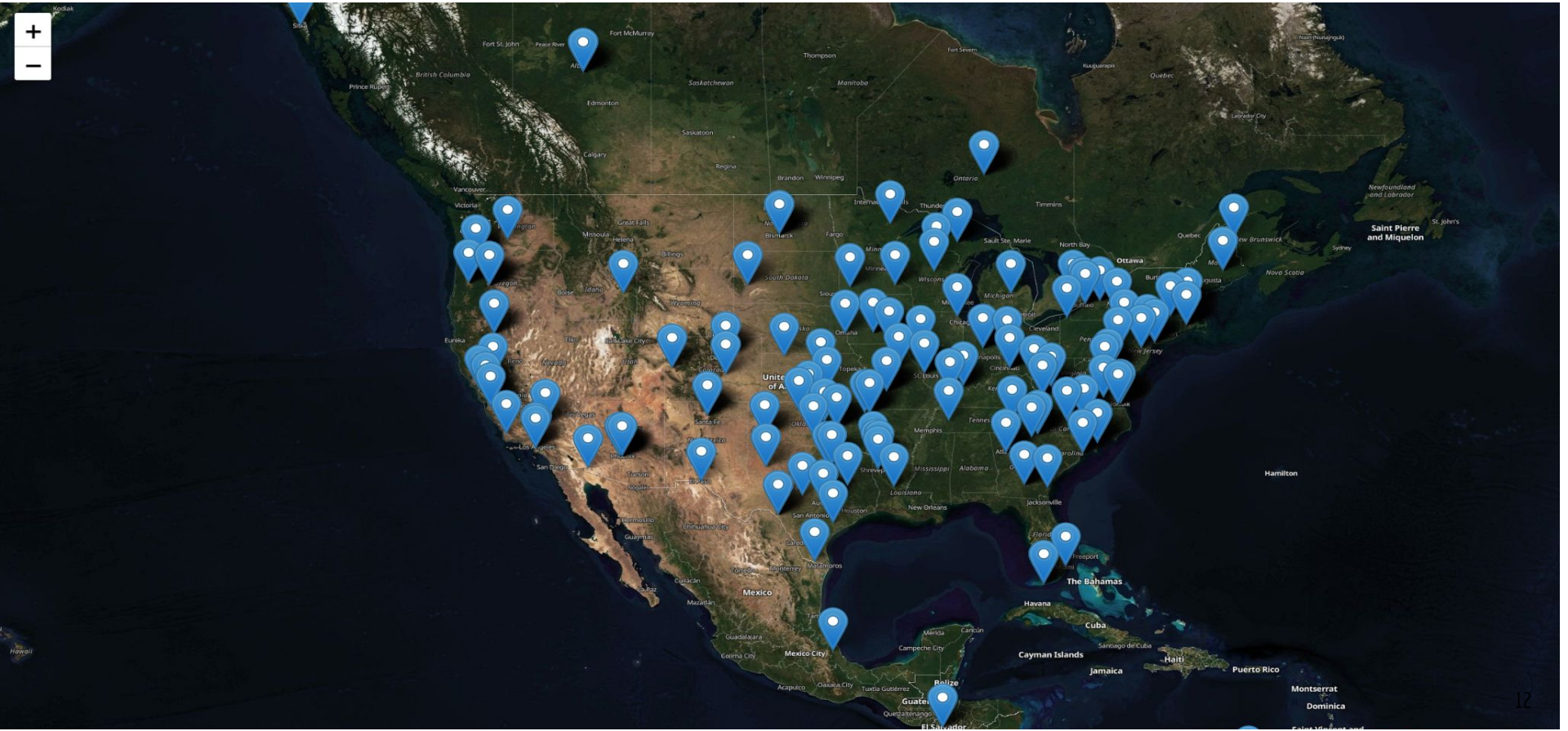
```plots_index.html > html > head > title
<!DOCTYPE html>
<html lang="en">
<head>
  <link rel="stylesheet" href="https://unpkg.com/leaflet@1.9.4/dist/leaflet.css" integrity="sha256-p4NxAoJBhIIN+hmNHrzRCf9tD/m
  <script src="https://unpkg.com/leaflet@1.9.4/dist/leaflet.js" integrity="sha256-20nQCchB9co0qIjJZRGuk2/Z9VM+kNiyxNV1lvTlZBo=
  <script src="geometry_plots.js"></script>

  <style>
    #map {position: absolute;top:0; bottom: 0; left: 0; right: 0;}
  </style>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>plot_index </title>
</head>
<body>
  <div id="map"></div>
  <script>
    var map = L.map("map").setView([38.690003,-100.809859],4);
    L.tileLayer("https://api.maptiler.com/maps/hybrid/{z}/{x}/{y}.jpg?key=TWFFdt2UIcIqMQFcGETR",{
      attribution: '<a href="https://www.maptiler.com/copyright/" target="_blank">&copy; MapTiler</a> <a href="https://www.ope
    }).addTo(map);

    var myLayer: any
    va                    ON().addTo(map);
      myLayer.addData(US_data);
  </script>
</body>
</html>
```

Convert of geojson to js. Then var data as US_data. We are able to run it as a script into our html code. Code will plot unique airports location.

	security_delay_min	late_aircraft_delay_min	longitude	latitude
0	0.0	17.0	-84.374393	40.165883
1	0.0	63.0	-92.540955	31.329274
2	0.0	276.0	-122.396008	37.779418
3	0.0	66.0	-115.002136	55.001251
4	0.0	10299.0	-84.429271	33.637799
...	...	...	...	...
1313	0.0	396.0	130.424506	33.713871
1335	0.0	894.0	-120.521556	34.919978
1397	0.0	0.0	116.627613	-8.304813
4072	0.0	365.0	-97.620488	35.396247
5950	0.0	0.0	81.53892	30.07573

[345 rows x 21 columns]

Pretty pictures: Jose's map



Top 5s - Delays by Airline

Connecting (with psycopg2) to PostgreSQL database to query in Jupyter Notebooks *(psycopg2 allows us to connect to the PostgreSQL database and execute SQL queries from within a Python environment.).*

```
from sqlalchemy import create_engine
import pandas as pd
```

```
# Using default parameters for a local PostgreSQL instance
host = 'localhost'
port = '5432'
database = '2023_flight_data'
user = 'postgres'
password = 'XXXXXXXXXX'

# Create the database connection string
connection_string = f'postgresql+psycopg2://{user}:{password}@{host}:{port}/{database}'

# Create Database Connection
engine = create_engine(connection_string)
conn = engine.connect()
```

```
# Query to fetch the data
query = "SELECT * FROM flight_data;"

# Load data into a pandas DataFrame
df = pd.read_sql(query, engine)

# Display the DataFrame
print(df.head())
```

```
query_unique_carriers = """
SELECT COUNT(DISTINCT carrier_name) AS total_unique_carriers
FROM flight_data;
"""

df_unique_carriers = pd.read_sql(query_unique_carriers, engine)
print(df_unique_carriers)
```

```
   total_unique_carriers
0                     21
```

```
query_unique_airports = """
SELECT COUNT(DISTINCT airport) AS total_unique_airports
FROM flight_data;
"""

df_unique_airports = pd.read_sql(query_unique_airports, engine)
print(df_unique_airports)
```

```
   total_unique_airports
0                    359
```

Top 5s - Delays by Airline

Total number of Airlines: 21

Rank	Total Delays - ct	Total Delays - Hr	Flts Cancelled - ct	Flts Diverted - ct
1	Southwest Airlines (309963)	American Airlines Network (304923.5)	Southwest Airlines (14325)	Southwest Airlines (2902)
2	American Airlines Network (213850)	Southwest Airlines (256699.5)	United Airlines Network (10270)	American Airlines Network (2495)
3	Delta Airlines Network (159257)	Delta Airlines Network (189609.5)	Delta Air Lines Network (10016)	SkyWest Airlines Inc. (2051)
4	United Airlines Network (148386)	United Airlines Network (175673.6)	American Airlines Network (9978)	Delta Air Lines Network (2039)
5	SkyWest Airlines Inc. (108189)	SkyWest Airlines Inc. (140363.4)	SkyWest Airlines Inc. (8186)	United Airlines Network (1910)

Top 5s - Delays by Airline (specific delays by count)

Rank	Carrier Delay	Weather	Nat Air Sys	Security	Late Aircraft
1	Southwest Airlines	SkyWest Airlines Inc.	Southwest Airlines	**Spirit Airlines**	Southwest Airlines
2	Delta Airlines Network	American Airlines Network	American Airlines Network	Southwest Airlines	American Airlines Network
3	American Airlines Network	Delta Air Lines Network	United Airlines Network	American Airlines Network	United Airlines Network
4	SkyWest Airlines Inc.	United Airlines Network	Delta Air Lines Network	**Alaska Airlines Network**	Delta Airlines Network
5	United Airlines Network	Southwest Airlines	**Spirit Airlines**	SkyWest Airlines Inc.	**JetBlue Airways**

Top 5s - Delays by Airline (specific delays by hour)

Rank	Carrier Delay	Weather	Nat Air Sys	Security	Late Aircraft
1	American Airlines Network	SkyWest Airlines Inc.	Southwest Airlines	**Spirit Airlines**	American Airlines Network
2	Delta Air Lines Network	American Airlines Network	American Airlines Network	Southwest Airlines	Southwest Airlines
3	SkyWest Airlines Inc.	Delta Air Lines Network	United Airlines Network	American Airlines Network	United Airlines Network
4	Southwest Airlines	United Airlines Network	**Spirit Airlines**	SkyWest Airlines Inc.	Delta Air Lines Network
5	United Airlines Network	Southwest Airlines	Delta Air Lines Network	**Alaska Airlines Network**	**JetBlue Airways**

Top 5s - Delays by Airline (takeaways)

Top 5 airlines (# of flights):

1. Southwest
2. Delta
3. American
4. United
5. SkyWest

Airline	Total Delays (count)	Total # of Flts	Delay percentage
Frontier Airlines	54525	177542	31%
JetBlue Airways	83550	274852	30%
Spirit Airlines	75428	263871	29%
Allegiant Air	29653	115539	26%
Hawaiian Airlines Network	18958	80967	23%

Top 5s - Delays by Airport

Total number of Airports: 359

Rank	Total Delays - ct	Total Delays - Hr	Flts Cancelled - ct	Flts Diverted - ct
1	DEN (62805)	DFW (89335.9)	DFW (4691)	DEN (1167)
2	DFW (60305)	ORD (76050.9)	DEN (4587)	DFW (956)
3	ORD (56837)	DEN (74204.33)	EWR (4574)	LGA (702)
4	ATL (55727)	ATL (70323.02)	LGA (4553)	ORD (699)
5	LAS (48933)	CLT (59510.08)	ORD (3932)	MCO (661)

Top 5s - Delays by Airport (specific delays by count)

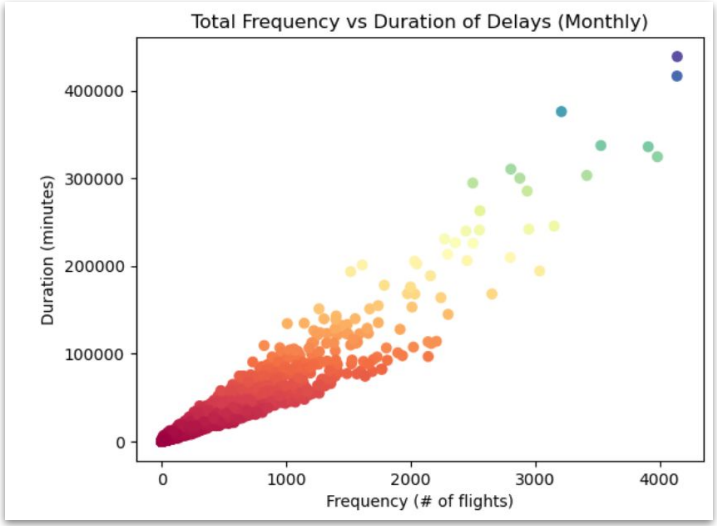
Rank	Carrier Delay	Weather	Nat Air Sys	Security	Late Aircraft
1	ORD	DFW	DEN	DFW	DFW
2	DEN	DEN	LAS	ANC	DEN
3	ATL	ATL	ORD	LAX	ATL
4	DFW	ORD	DFW	LAS	ORD
5	LAX	CLT	EWR	ATL	LAS

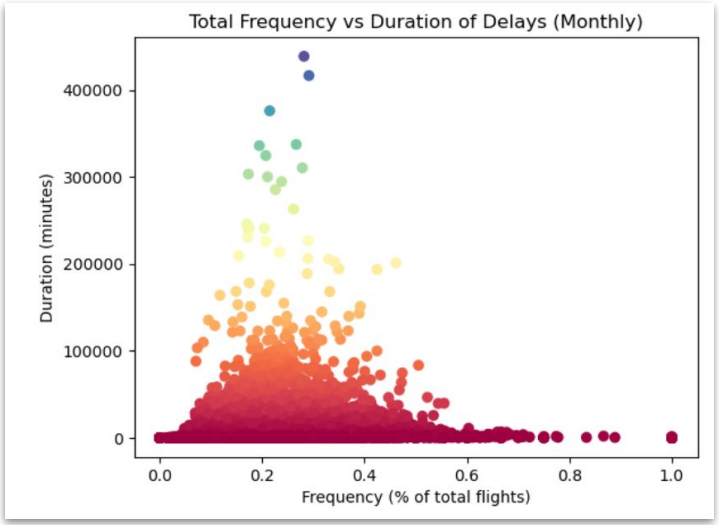
Top 5s - Delays by Airport (specific delays by hours)

Rank	Carrier Delay	Weather	Nat Air Sys	Security	Late Aircraft
1	DFW	DFW	DEN	DFW	DFW
2	ATL	DEN	LAS	CLT	ORD
3	ORD	ORD	ORD	LAS	DEN
4	DEN	ATL	EWR	ATL	CLT
5	CLT	CLT	MCO	LAX	ATL

Frequency of Delays vs Duration of Delay

Exploring possible correlation or insights regarding frequency and duration of flight delays overall and by carrier (airline).





pandas

Calculated frequency of delays as function of total arrivals

	flight_data_id	month	carrier_name	airport	city	total_arrivals	total_delays_ct	percent_delayed
0	1	12	Endeavor Air Inc.	ABE	Allentown/Bethlehem/Easton, PA	72.0	5.0	0.069444
1	2	12	Endeavor Air Inc.	AEX	Alexandria, LA	62.0	7.0	0.112903
2	3	12	Endeavor Air Inc.	AGS	Augusta, GA	95.0	10.0	0.105263

Frequency of Delays vs Duration of Delay by Airline

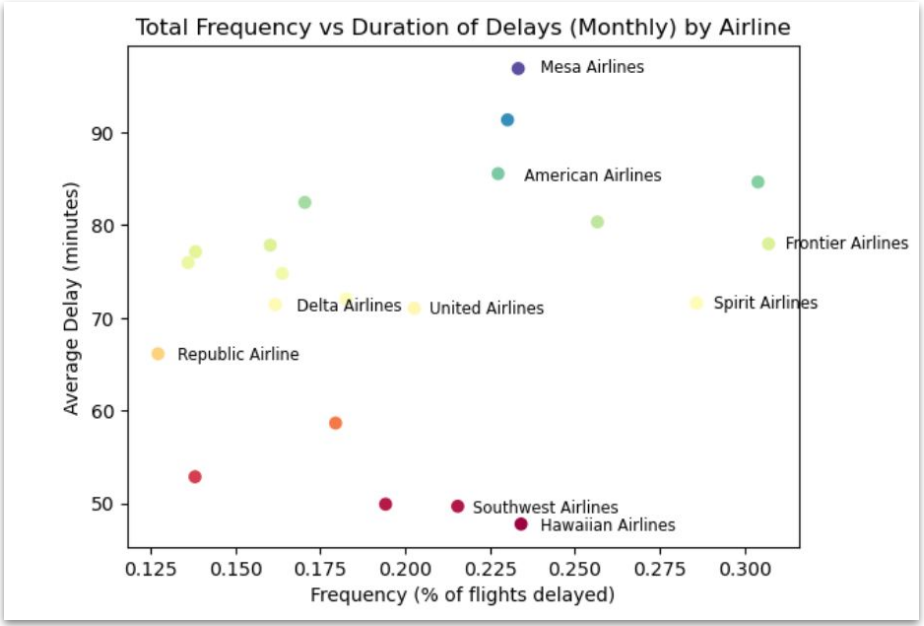
```
#extract columns of data from SQL flight_data table to perform same analysis above but grouped by airline
query_airlines = """
SELECT carrier_name, SUM(total_arrivals) AS total_arrivals, SUM(total_delays_ct) AS total_delays_ct, SUM(total_delays_min) AS total_delays_min
FROM flight_data
GROUP BY carrier_name
ORDER BY total_arrivals DESC;
"""
df_byairline = pd.read_sql(query_airlines, engine)
df_byairline
```

SQLAlchemy psycopg

```
airfrequency = df_byairline['total_delays_ct']
airduration = df_byairline['total_delays_min']
airratio = df_byairline['total_delays_ct']/df_byairline['total_arrivals']
airavemin = df_byairline['total_delays_min']/df_byairline['total_delays_ct']

df_byairline['percent_delayed'] = airratio
df_byairline['average_delay'] = airavemin
df_byairline
```

```
rank_time = df_byairline.sort_values(['average_delay'], ascending=False)
rank_time
```

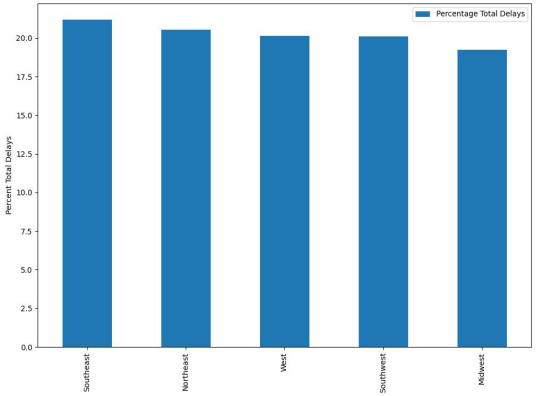
	carrier_name	total_arrivals	total_delays_ct	total_delays_min	percent_delayed	average_delay
15	Mesa Airlines Inc.	88678.0	20690.0	2004563.0	0.233316	96.885597
19	Air Wisconsin Airlines Corp	58078.0	13367.0	1221136.0	0.230156	91.354530
2	American Airlines Network	940531.0	213850.0	18295407.0	0.227372	85.552523
6	JetBlue Airways	274852.0	83550.0	7072583.0	0.303982	84.650904
17	CommuteAir LLC dba CommuteAir	70808.0	12067.0	994944.0	0.170419	82.451645
13	Allegiant Air	115539.0	29653.0	2382440.0	0.256649	80.343979
12	Frontier Airlines	177542.0	54525.0	4252311.0	0.307110	77.988281
4	SkyWest Airlines Inc.	675285.0	108189.0	8421804.0	0.160212	77.843441
14	Piedmont Airlines	99047.0	13687.0	1055754.0	0.138187	77.135530
10	Endeavor Air Inc.	201517.0	27408.0	2081666.0	0.136008	75.951036
11	PSA Airlines Inc.	194205.0	31800.0	2378261.0	0.163744	74.788082
20	GoJet Airlines LLC d/b/a United Express	45052.0	8226.0	592645.0	0.182589	72.045344
7	Spirit Airlines	263871.0	75428.0	5400279.0	0.285852	71.595150
1	Delta Air Lines Network	984986.0	159257.0	11376572.0	0.161685	71.435303
3	United Air Lines Network	732212.0	148386.0	10540416.0	0.202654	71.033763

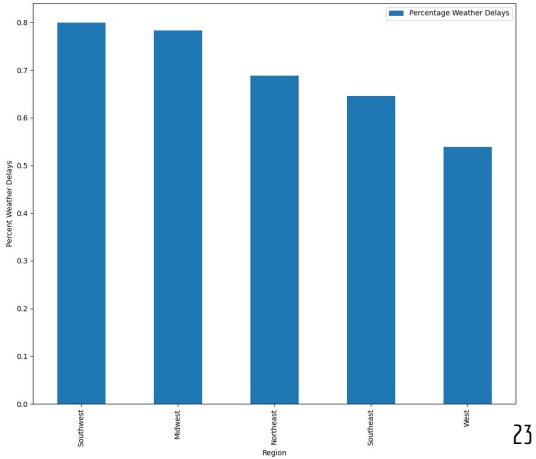


Regional Analysis

```
from sqlalchemy import text

# Define the SQL query
query = '''
SELECT DISTINCT airport, city,
   SUM(total_arrivals) AS Total_Arrivals,
   SUM(total_delays_ct) AS total_delays,
   SUM(carrier_ct) AS carrier_delays,
   SUM(weather_ct) AS weather_delays,
   SUM(nat_air_sys_ct) AS nat_air_sys_delays,
   SUM(late_aircraft_ct) AS late_aircraft_delays,
   month
FROM flight_data
WHERE city LIKE '%CT%' OR city LIKE '%DE%' OR city LIKE '%DC%' OR city LIKE
'%ME%' OR city LIKE '%MD%' OR city LIKE '%MA%' OR city LIKE '%NH%' OR city
LIKE '%NJ%' OR city LIKE '%NY%' OR city LIKE '%PA%' OR city LIKE '%PR%' OR
city LIKE '%RI%' OR city LIKE '%VT%' OR city LIKE '%VI%' OR city LIKE '%VA%'
OR city LIKE '%WV%'
GROUP BY city, airport, month
'''

# Create a connection from the Engine
conn = engine.connect()

# Execute the query and fetch the result as a list of tuples
result = conn.execute(text(query)).fetchall()
```

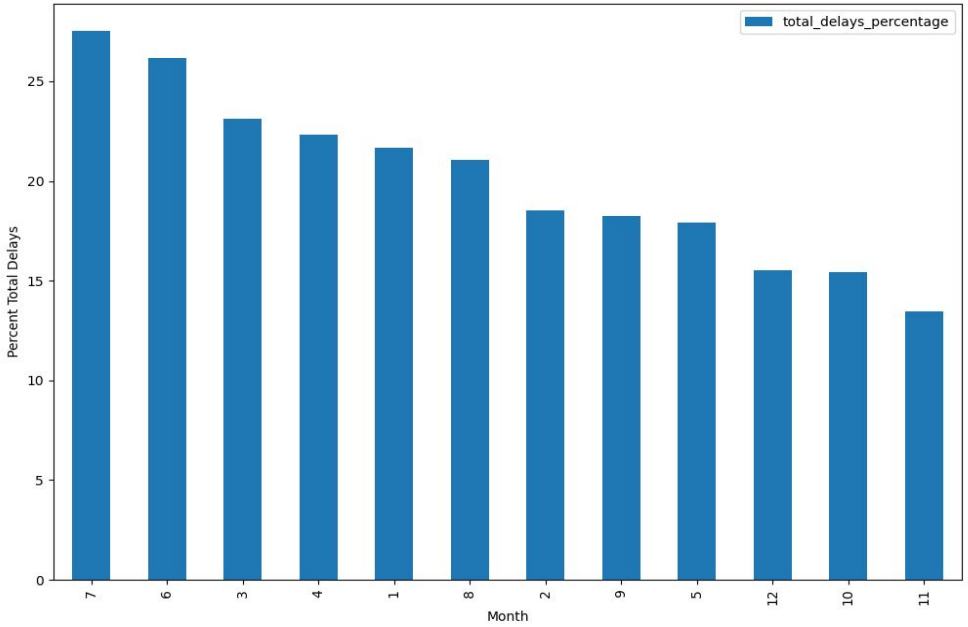
```
# Convert the result into a list of
dictionaries
result_dicts = []
for row in result:
    result_dict = {
        'airport': row[0],
        'city': row[1],
        'Total_Arrivals': row[2],
        'total_delays': row[3],
        'carrier_delays': row[4],
        'weather_delays': row[5],
        'nat_air_sys_delays': row[6],
        'late_aircraft_delays': row[7],
        'month': row[8]
    }
    result_dicts.append(result_dict)

# Close the connection
conn.close()

# Create a Pandas DataFrame from the
list of dictionaries
northEast = pd.DataFrame(result_dicts)

# Display the DataFrame
northEast
```

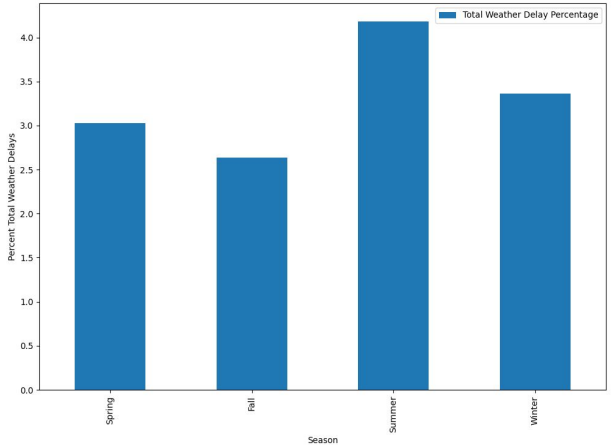


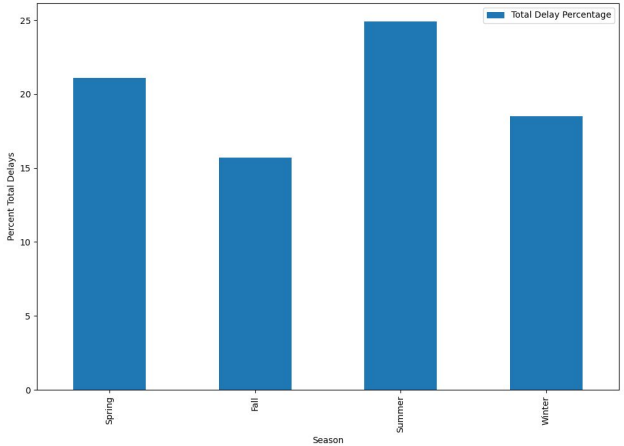


Seasonal Analysis

```
query = ''' SELECT * FROM flight_data
WHERE month IN ('3','4','5')'''

spring = pd.read_sql(query, engine)
```







Summary & Conclusions

Top 5 Analysis: Identifying the airlines with the highest number of delays by count is interesting, but not useful. In most "Top 5" analyses, the 5 airlines that had the most routes also had the most delays. The outliers are the interesting data points (e.g. Spirit Airlines ranks #1 in number of security delays but #9 in number of routes). A review by percentage would yield more useful information.

Regional & Seasonal Analysis: the regional delays due to weather for all analyzed regions was less than 1%. While the southwest region showed over 20% of all flights were delayed, roughly only .8% of those are attributed to weather. Summer showed to be the season with the highest total delays and delays due to weather with July being the month with the highest total delays.

Frequency vs Duration of Delays: There is a positive correlation between frequency (in #) of delays and total minutes delayed for a given airline, airport and month. This makes sense as more delays contribute to a longer total delay time. When graphing the distribution of frequency as the % of total flights delayed vs duration, there is a somewhat random distribution. This provides evidence against the hypothesis that airlines may be willing to endure more frequent delays if delays are shorter in duration and airlines with fewer delays may have longer delays. Lastly, when plotting % of flights delayed out of total flights against average duration of delay per flight this revealed specific airlines who perform poorly and well in both categories. This may be helpful to travelers in choosing high risk vs low risk or based on the variable they are more willing to accommodate.

Ethical Considerations: The data used does not qualify as human subjects research or analysis; therefore privacy is not a consideration. The data is publicly available, therefore ownership is not a consideration. While we are not aware of AI/machine-learning algorithms to generate data or train the data set given the data is generated and published monthly, we cannot rule out the possibility of other unknown biases such as ascertainment bias (e.g. if certain airlines do not contribute data or data related to specific variables). Possibilities we considered with regards to our data exploration related to intent and outcomes included the potential to perpetuate a negative stereotype about a specific airline or airport based on preliminary and unsubstantiated findings from our data exploration, which could unfairly prejudice individuals against a specific airline.