Monthly Airline Delays by US Airport (2003 - 2016)

Table of Contents

- **5. Lessons Learned**

1. Project Description

Travelling by plane, either for business or pleasure, has become a very popular means of transportation for most people around the world. As a result, the airline industry has grown rapidly and airports nowadays manage thousands of flights, which also means thousands of passengers per year.
Unfortunately, where there is a lot of traffic, traffic jams are not far. In the airline industry, delays can quickly become expensive if customers miss their connecting flight, additional planes have to be delayed or a flight might even have to be rerouted.

For this project, the goal is to analyse the data set which contains all the delays for several US airports for every month from 2003 until 2016. To achieve this goal, a first overview over the data set along with a description of the different data classes will be provided. Second, the data set will be downloaded and imported in to a MongoDB. Third, the data will be transformed to be analysed properly. Fourth, some aggregation statements are run to get a better understanding of the data and the flight delays over the years. And finally, a short conclusion will summarize the project outcome along with the lessons learned.

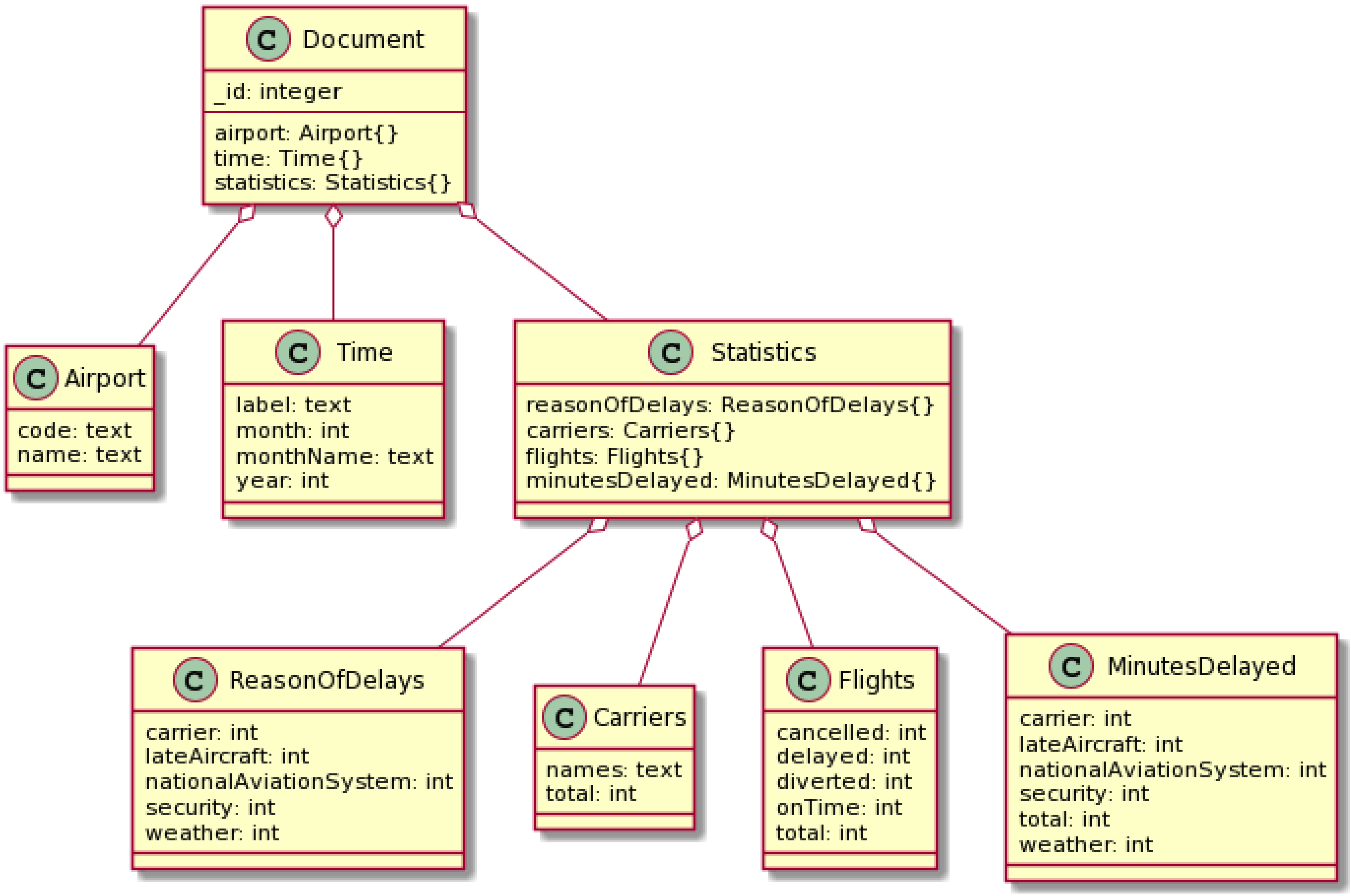
1.1. Research Questions

As mentioned above, the project tries to answer several questions with the available data.
The questions to be answered are the following:

1. Show the Airport with lowest total number of flights delayed in one month
2. Show the Airport with highest total number of flights delayed in one month
3. Which Airports had less than 500 delayed flights per month?
4. Which Airport had the most flights cancelled in one month over all years?
5. Which year had the most flights delayed?
6. How many flights had to be diverted per year?
7. Which are the top 5 airports that had the most flights delayed in 2015 due to weather condiditons?
8. How many minutes were all flights in a month delayed on average, per year for Chicago O'Hare International (ORD)?
9. Which were the top 10 months (months with most delays) due to weather for flight delays at JFK?
10. How many flight delays were there per category for 2005 and 2015?
11. Which Airport has the most delayed flights for a given year and how many?

1.2. Data Structure and Class Diagramms

The data in this project was retrieved form the following URL:
[https://think.cs.vt.edu/corgis/datasets/json/airlines/airlines.json (https://think.cs.vt.edu/corgis/datasets/json/airlines/airlines.json)](https://think.cs.vt.edu/corgis/datasets/json/airlines/airlines.json)
It contains the information about flight delays by airport and month starting in 2003 until 2016. It consists of the information about the airport, the time of the recording and several statistical recordings. The structure of the data is displayed in the class diagram below.



Source:
//www.plantuml.com/plantuml/png/jLAxhjim3Dtz5HpRmJ_WwLuawDIaGDDk0GfMPcAXUWGIdTO8ykzL9D1E1A
HOP0u_MDApP46m6xRp_0qVtS_SLDWPZXBrdPEtTqNswf6IDnaoVcqRbuDQhqP9Zk3K0ltJVWNPCPme9jcrwK8
HaRQc37-dDC3u88ciLEpZCTWhTXQOK5nRb-rJCz8yJS6W5I7B-
DsSQ_4rcJhNUyP3x4x1VYW4cmb2Yyj5JB_cxG1Yz-
tRQai4fjVhz1UT958QWAQJWfbZ5VJBwF73oVYb6TbhjHgQOgoa9jtl1JzDNH71RY6GSWqn98nNmnE9pUMjetgyw
X0PX7TTasQ-Tl1j2tALbRlSN-LylgLciKkmNZ30IewiOYqckgyLLh8rR-JwqPh
(http://www.plantuml.com/plantuml/png/jLAxhjim3Dtz5HpRmJ_WwLuawDIaGDDk0GfMPcAXUWGIdTO8ykzL9D
HOP0u_MDApP46m6xRp_0qVtS_SLDWPZXBrdPEtTqNswf6IDnaoVcqRbuDQhqP9Zk3K0ltJVWNPCPme9jcrwK8
HaRQc37-dDC3u88ciLEpZCTWhTXQOK5nRb-rJCz8yJS6W5I7B-
DsSQ_4rcJhNUyP3x4x1VYW4cmb2Yyj5JB_cxG1Yz-
tRQai4fjVhz1UT958QWAQJWfbZ5VJBwF73oVYb6TbhjHgQOgoa9jtl1JzDNH71RY6GSWqn98nNmnE9pUMjetgyw
X0PX7TTasQ-Tl1j2tALbRlSN-LylgLciKkmNZ30IewiOYqckgyLLh8rR-JwqPh)_

In order for the reader to better understand the meaning of values in "ReasonOfDelays" and "MinutesDelayed", the official description has been provided below:

- 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.

1.3. Requirements and Configuration

In [411]:

```
# Import all needed libraries
import pymongo
from pprint import pprint
import pandas as pd
import requests
import json
import datetime
import matplotlib.pyplot as plt
```

In [412]:

```
# Pandas configuration
pd.set_option('precision', 2)
pd.set_option('max_rows', 30)
pd.set_option('max_colwidth', 50)
# pd.describe_option('max_rows')
# pd.describe_option('precision')
# pd.describe_option('max_colwidth')
```

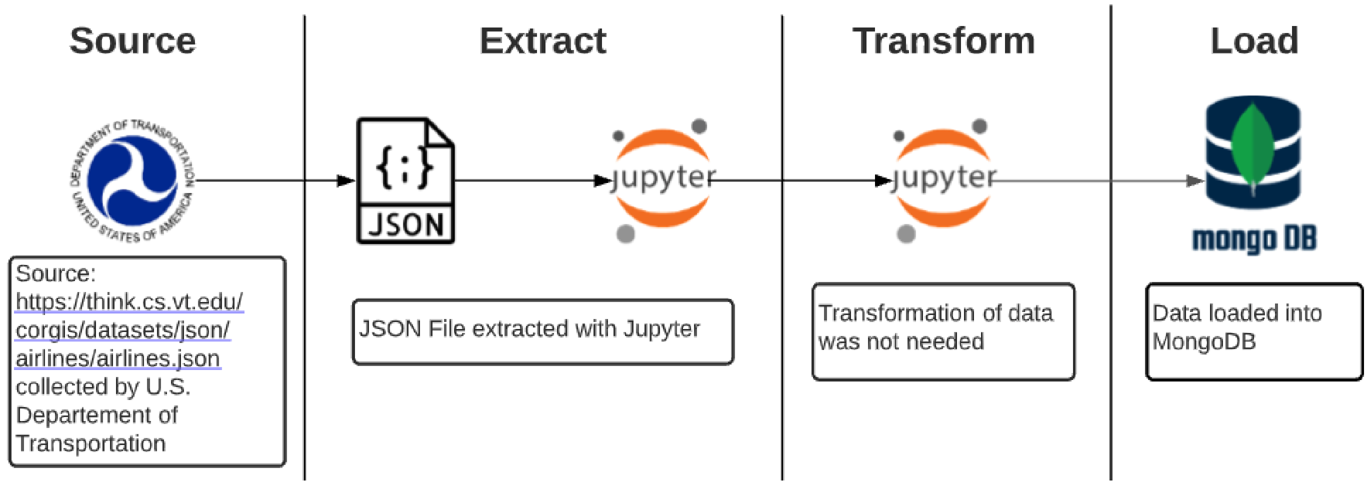
In [413]:

```
# API and Database details
API = "https://think.cs.vt.edu/corgis/datasets/json/airlines/airlines.json"
CNX_STR = "mongodb+srv://Chrigi:SuperStrongPW39%21@main.io4fa.mongodb.net"
DB_NAME = "project"
COLL_NAME = "airlines"
```

In [414]:

```
# Create connection to MongoDB
client = pymongo.MongoClient(CNX_STR)
db = client[DB_NAME]
airlines = db[COLL_NAME]
```

2. Extract, Transform, Load



2.1. Clear existing collections

In [415]:

```
# All collections in the DB are dropped to ensure a clear environment
airlines.drop()
airlines.count_documents({})
```

Out[415]:

0

2.2. Fetch Data

In [416]:

```
# Fetch JSON from API_URL and print part of the result
r = requests.get(API)
data = json.loads(r.text)
print(r.text[:1200])
```


```
[
  {
    "Airport": {
      "Code": "ATL",
      "Name": "Atlanta, GA: Hartsfield-Jackson Atlanta International"
    },
    "Time": {
      "Label": "2003/06",
      "Month": 6,
      "Month Name": "June",
      "Year": 2003
    },
    "Statistics": {
      "# of Delays": {
        "Carrier": 1009,
        "Late Aircraft": 1275,
        "National Aviation System": 3217,
        "Security": 17,
        "Weather": 328
      },
      "Carriers": {
        "Names": "American Airlines Inc.,JetBlue Airways,Continental A
ir Lines Inc.,Delta Air Lines Inc.,Atlantic Southeast Airlines,AirTran
Airways Corporation,America West Airlines Inc.,Northwest Airlines In
c.,ExpressJet Airlines Inc.,United Air Lines Inc.,US Airways Inc.",
        "Total": 11
      },
      "Flights": {
        "Cancelled": 216,
        "Delayed": 5843,
        "Diverted": 27,
        "On Time": 23974,
        "Total": 30060
      },
      "Minutes Delayed": {
        "Carrier": 61606,
        "Late Aircraft": 68335,
        "National Aviation System": 118831,
        "Security": 518,
        "Total": 268764,
        "Weather": 19474
      }
    }
  },
  {
    "Airport": {
      "Code": "BOS",
      "Name": "Boston, MA:
```

In [417]:

```
# Check amount of documents retrieved from API
len(data)
```

Out[417]:

```
4408
```

In [418]:

```
# Pretty print one document
pprint(data[20])
```

```
{'Airport': {'Code': 'ORD',
             'Name': "Chicago, IL: Chicago O'Hare International"},
 'Statistics': {'# of Delays': {'Carrier': 801,
                                'Late Aircraft': 982,
                                'National Aviation System': 2947,
                                'Security': 11,
                                'Weather': 157},
                'Carriers': {'Names': 'American Airlines Inc.,Alaska A
irlines '
                                      'Inc.,Continental Air Lines '
                                      'Inc.,Atlantic Coast Airlines,De
lta Air '
                                      'Lines Inc.,Atlantic Southeast '
                                      'Airlines,America West Airlines
'
                                      'Inc.,American Eagle Airlines '
                                      'Inc.,Northwest Airlines Inc.,Ex
pressJet '
                                      'Airlines Inc.,United Air Lines
Inc.,US '
                                      'Airways Inc.',
                             'Total': 12},
                'Flights': {'Cancelled': 261,
                            'Delayed': 4899,
                            'Diverted': 38,
                            'On Time': 24997,
                            'Total': 30195},
                'Minutes Delayed': {'Carrier': 54296,
                                    'Late Aircraft': 55855,
                                    'National Aviation System': 13585
1,
                                    'Security': 604,
                                    'Total': 257007,
                                    'Weather': 10401}},
 'Time': {'Label': '2003/06', 'Month': 6, 'Month Name': 'June', 'Yea
r': 2003}}
```

2.3. Insert into MongoDB

In [419]:

```
# Insert the list of airline delays documents in "data" into MongoDB collection "ai
airlines.insert_many(data);
```

In [420]:

```
# count number of documents inserted and check if all have been imported
airlines.count_documents({})
```

Out[420]:

```
4408
```

In [421]:

```
result = airlines.aggregate([
      {"$limit": 5},
])

pd.DataFrame(result)
```

Out[421]:

	_id	Airport	Time	Statistics
0	6182de5e01a3eff37d7b54e2	{'Code': 'ATL', 'Name': 'Atlanta, GA: Hartsfie...	{'Label': '2003/06', 'Month': 6, 'Month Name':...	{'# of Delays': {'Carrier': 1009, 'Late Aircra...
1	6182de5e01a3eff37d7b54e3	{'Code': 'BOS', 'Name': 'Boston, MA: Logan Int...	{'Label': '2003/06', 'Month': 6, 'Month Name':...	{'# of Delays': {'Carrier': 374, 'Late Aircraf...
2	6182de5e01a3eff37d7b54e4	{'Code': 'BWI', 'Name': 'Baltimore, MD: Baltim...	{'Label': '2003/06', 'Month': 6, 'Month Name':...	{'# of Delays': {'Carrier': 296, 'Late Aircraf...
3	6182de5e01a3eff37d7b54e5	{'Code': 'CLT', 'Name': 'Charlotte, NC: Charlo...	{'Label': '2003/06', 'Month': 6, 'Month Name':...	{'# of Delays': {'Carrier': 300, 'Late Aircraf...
4	6182de5e01a3eff37d7b54e6	{'Code': 'DCA', 'Name': 'Washington, DC: Ronal...	{'Label': '2003/06', 'Month': 6, 'Month Name':...	{'# of Delays': {'Carrier': 283, 'Late Aircraf...

2.4. DB Overview

In [422]:

```
# Check all DB names in the cluster
client.list_database_names()
```

Out[422]:

```
['project', 'admin', 'local']
```

In [423]:

```
# Count the documents in the airlines collection
airlines.count_documents({})
```

Out[423]:

```
4408
```

In [424]:

```
coll = pd.DataFrame(client['project'].list_collection_names(),
                    columns=["collections"])
coll
```

Out[424]:

	collections
0	airlines

3. Data Analysis

3.1. Show Airport with lowest total number of flights delayed in one month

In [425]:

```
flatten = {'$addFields': {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_delayed": "$Statistics.Flights.Delayed",
    "total_flights": "$Statistics.Flights.Total"

}}

project = {"$project": {
    "_id": 0,
    "Airport": 0,
    "Time": 0,
    "Statistics": 0
}}

sort = {'$sort': {'total_flights_delayed': 1}}

limit = {'$limit': 1}

pipeline = [flatten, project, sort, limit]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[425]:

	airport_code	airport_name	month	year	total_flights_delayed	total_flights
0	PDX	Portland, OR: Portland International	November	2009	283	3982

3.2. Show Airport with highest total number of flights delayed in one month

In [426]:

```
flatten = {'$addFields': {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_delayed": "$Statistics.Flights.Delayed",
    "total_flights": "$Statistics.Flights.Total"
}}

project = {"$project": {
    "_id": 0,
    "Airport": 0,
    "Time": 0,
    "Statistics": 0
}}

sort = {'$sort': {'total_flights_delayed': -1}}

limit = {'$limit': 1}

pipeline = [flatten, project, sort, limit]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[426]:

	airport_code	airport_name	month	year	total_flights_delayed	total_flights
0	ATL	Atlanta, GA: Hartsfield-Jackson Atlanta Intern...	July	2005	13699	37584

3.3. Which Airports had less than 500 delayed flights per month?

In [427]:

```
flatten = {"$addFields": {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_delayed": "$Statistics.Flights.Delayed",
    "total_flights": "$Statistics.Flights.Total"
}}

match = {"$match": {
    "total_flights_delayed": {"$lte": 500}
}}

project = {"$project": {
    "_id": 0,
    "Airport": 0,
    "Time": 0,
    "Statistics": 0
}}

sort = {"$sort": {"total_flights_delayed": 1}}

pipeline = [flatten, match, project, sort]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[427]:

	airport_code	airport_name	month	year	total_flights_delayed	total_flights
0	PDX	Portland, OR: Portland International	November	2009	283	3982
1	IAD	Washington, DC: Washington Dulles International	January	2016	285	2533
2	IAD	Washington, DC: Washington Dulles International	November	2015	320	2684
3	IAD	Washington, DC: Washington Dulles International	October	2015	352	3000
4	PDX	Portland, OR: Portland International	September	2009	373	4266
5	IAD	Washington, DC: Washington Dulles International	September	2015	397	2925
6	PDX	Portland, OR: Portland International	September	2003	428	4617
7	PDX	Portland, OR: Portland International	Febuary	2012	435	3741
8	IAD	Washington, DC: Washington Dulles International	April	2015	438	2955

	airport_code	airport_name	month	year	total_flights_delayed	total_flights
9	IAD	Washington, DC: Washington Dulles International	December	2015	443	2799
10	PDX	Portland, OR: Portland International	Febuary	2013	459	3566
11	PDX	Portland, OR: Portland International	September	2011	462	4431
12	PDX	Portland, OR: Portland International	September	2015	464	4465
13	PDX	Portland, OR: Portland International	April	2010	471	4223
14	PDX	Portland, OR: Portland International	September	2008	476	4535
15	PDX	Portland, OR: Portland International	Febuary	2010	478	3733
16	PDX	Portland, OR: Portland International	April	2012	482	4206
17	PDX	Portland, OR: Portland International	September	2012	488	4484
18	PDX	Portland, OR: Portland International	October	2015	489	4553

3.4. Which Airport had the most flights cancelled in one month over all years?

In [428]:

```
flatten = {"$addFields": {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_cancelled": "$Statistics.Flights.Cancelled",
    "total_flights": "$Statistics.Flights.Total"
}}

project = {"$project": {
    "_id": 0,
    "Airport": 0,
    "Time": 0,
    "Statistics": 0
}}

sort = {"$sort": {"total_flights_cancelled": -1}}

limit = {"$limit": 1}

pipeline = [flatten, project, sort, limit]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[428]:

	airport_code	airport_name	month	year	total_flights_cancelled	total_flights
0	ORD	Chicago, IL: Chicago O'Hare International	January	2014	3680	21529

3.5. Which year had the most flights delayed?

In [429]:

```
group = {"$group": {
    "_id": "$Time.Year",
    "total_flights_delayed": {"$sum": "$Statistics.Flights.Delayed"}
}}

project = {"$project": {
    "_id": 0,
    "year": "$_id",
    "total_flights_delayed": "$total_flights_delayed"
}}

sort = {"$sort": {"year": 1}}

pipeline = [group, project, sort]

result = airlines.aggregate(pipeline)
df = pd.DataFrame(result)
df
```

Out[429]:

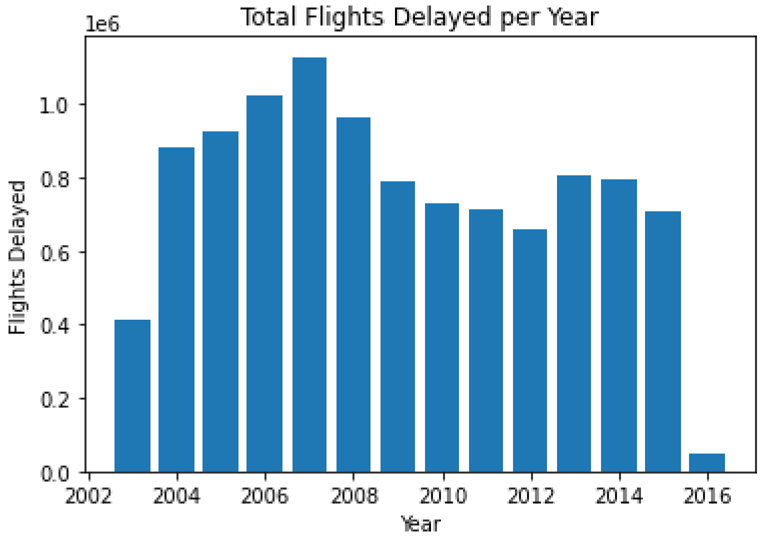
	year	total_flights_delayed
0	2003	411570
1	2004	880677
2	2005	925578
3	2006	1024612
4	2007	1129439
5	2008	965136
6	2009	787472
7	2010	732445
8	2011	715560
9	2012	658326
10	2013	805063
11	2014	796314
12	2015	707800
13	2016	48026

In [430]:

```
plt.bar(df['year'],df['total_flights_delayed'])
plt.title('Total Flights Delayed per Year')
plt.xlabel('Year')
plt.ylabel('Flights Delayed')
plt.show
```

Out[430]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



3.6. How many flights had to be diverted per year?

In [431]:

```
group = {"$group": {
    "_id": "$Time.Year",
    "total_flights_diverted": {"$sum": "$Statistics.Flights.Diverted"}
}}

project = {"$project": {
    "_id": 0,
    "year": "$_id",
    "total_flights_diverted": "$total_flights_diverted"
}}

sort = {"$sort": {"year": 1}}

pipeline = [group, project, sort]

result = airlines.aggregate(pipeline)
df = pd.DataFrame(result)
df
```

Out[431]:

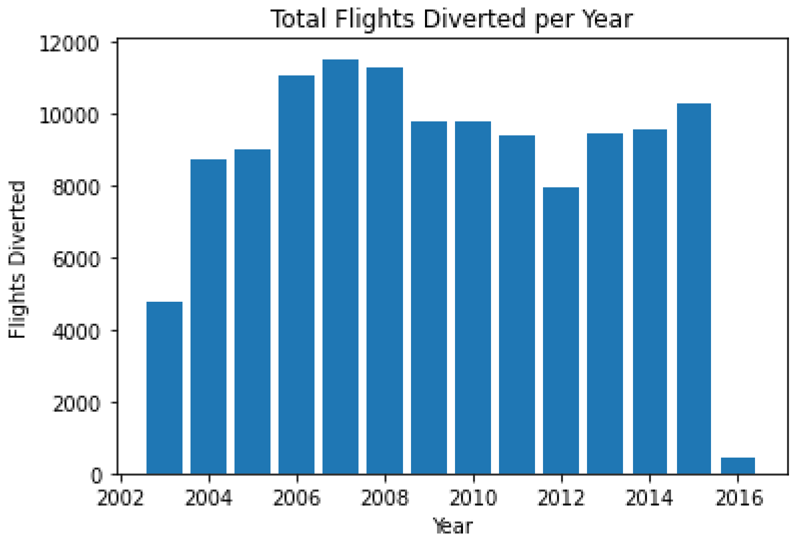
	year	total_flights_diverted
0	2003	4791
1	2004	8731
2	2005	8978
3	2006	11058
4	2007	11510
5	2008	11245
6	2009	9780
7	2010	9783
8	2011	9364
9	2012	7941
10	2013	9454
11	2014	9559
12	2015	10256
13	2016	449

In [432]:

```
plt.bar(df['year'],df['total_flights_diverted'])
plt.title('Total Flights Diverted per Year')
plt.xlabel('Year')
plt.ylabel('Flights Diverted')
plt.show
```

Out[432]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



The result for the year 2016 seems to be pretty low, lets see how many months were actually recorded in the data set for the year 2016.

In [433]:

```
# Check to see how many months were recorded for 2016
airlines.distinct("Time.Month", {"Time.Year": {"$eq": 2016}})
```

Out[433]:

```
[1]
```

There was only one month recorded in 2016. As a result, the number of total flights diverted is much smaller compared to previous years.

3.7. Which are the top 5 airports that had the most flights delayed in 2015 due to weather condititons?

In [434]:

```
flatten = {"$addFields": {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_delayed_weather": "$Statistics.# of Delays.Weather",
    "total_flights": "$Statistics.Flights.Total"
}}

match = {"$match": {
    "year": {"$gte": 2015}
}}

project = {"$project": {
    "_id": 0,
    "Airport": 0,
    "Time": 0,
    "Statistics": 0
}}

sort = {"$sort": {"Statistics.Flights.Delayed": -1}}

limit = {"$limit": 5}

pipeline = [flatten, match, project, sort, limit]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[434]:

	airport_code	airport_name	month	year	total_flights_delayed_weather	total_flights
0	CLT	Charlotte, NC: Charlotte Douglas International	January	2015	41	9351
1	DCA	Washington, DC: Ronald Reagan Washington National	January	2015	52	6680
2	BOS	Boston, MA: Logan International	January	2015	83	8841
3	ATL	Atlanta, GA: Hartsfield-Jackson Atlanta Intern...	January	2015	133	29492
4	BWI	Baltimore, MD: Baltimore/Washington Internatio...	January	2015	31	6913

3.8. How many minutes were all flights in a month delayed on average, per year for Chicago O'Hare International (ORD)?

In [435]:

```
flatten = {"$addFields": {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_delay_minutes": "$Statistics.Minutes Delayed.Total",
    "total_flights": "$Statistics.Flights.Total"
}}

match = {"$match": {
    "airport_name": {"$regex": "^Chic"}
}}

group = {"$group": {
    "_id": "$year",
    "avg_month_delay_minutes_ORD": {"$avg": "$Statistics.Minutes Delayed.Total"}
}}

project_2 = {"$project": {
    "_id": 0,
    "year": "$_id",
    "avg_month_delay_minutes_ORD": "$avg_month_delay_minutes_ORD",
}}

sort = {"$sort": {"year": 1}}

pipeline = [flatten, match, group, project_2, sort]

result = airlines.aggregate(pipeline)
df = pd.DataFrame(result)
df
```

Out[435]:

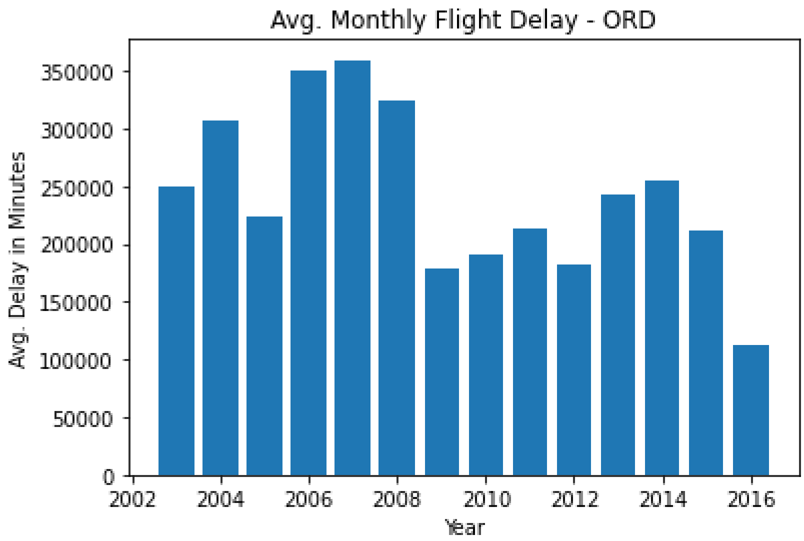
	year	avg_month_delay_minutes_ORD
0	2003	249004.71
1	2004	306344.79
2	2005	224445.42
3	2006	350345.67
4	2007	359680.38
5	2008	323957.04
6	2009	177962.00
7	2010	191019.88
8	2011	212510.25
9	2012	182360.08
10	2013	242996.04
11	2014	254212.46
12	2015	211790.29
13	2016	111977.50

In [436]:

```
plt.bar(df['year'],df['avg_month_delay_minutes_ORD'])
plt.title('Avg. Monthly Flight Delay - ORD')
plt.xlabel('Year')
plt.ylabel('Avg. Delay in Minutes')
plt.show
```

Out[436]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



3.9. Which were the top 10 months (months with most delays) due to weather for flight delays for JFK?

In [437]:

```
flatten = {"$addFields": {
    'airport_code': '$Airport.Code',
    "airport_name": "$Airport.Name",
    "month": "$Time.Month Name",
    "year": "$Time.Year",
    "total_flights_delayed_weather": "$Statistics.# of Delays.Weather",
    "total_flights": "$Statistics.Flights.Total"
}}

match = {"$match": {
    "airport_code": {"$eq": "JFK"}
}}

project = {"$project": {
    "_id": 0,
    "Airport": 0,
    "Time": 0,
    "Statistics": 0
}}

sort = {"$sort": {"total_flights_delayed_weather": -1}}

limit = {"$limit": 10}

pipeline = [flatten, match, project, sort, limit]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[437]:

	airport_code	airport_name	month	year	total_flights_delayed_weather	total_flights
0	JFK	New York, NY: John F. Kennedy International	July	2007	196	10964
1	JFK	New York, NY: John F. Kennedy International	October	2006	194	10296
2	JFK	New York, NY: John F. Kennedy International	August	2007	190	11064
3	JFK	New York, NY: John F. Kennedy International	July	2008	189	11499
4	JFK	New York, NY: John F. Kennedy International	November	2006	184	10097
5	JFK	New York, NY: John F. Kennedy International	August	2008	176	11420
6	JFK	New York, NY: John F. Kennedy International	December	2008	163	9419
7	JFK	New York, NY: John F. Kennedy International	July	2004	163	8308

	airport_code	airport_name	month	year	total_flights_delayed_weather	total_flights
8	JFK	New York, NY: John F. Kennedy International	March	2007	161	11014
9	JFK	New York, NY: John F. Kennedy International	June	2008	159	9930

It seems as if there is a clear tendency for weather delays in 2006 - 2008. Lets check if actually all years were recorded for JFK in the data set.

In [438]:

```
# Check to see if all years are recorded for JFK
airlines.distinct("Time.Year", {"Airport.Code": {"$eq": "JFK"}})
```

Out[438]:

```
[2003,
 2004,
 2005,
 2006,
 2007,
 2008,
 2009,
 2010,
 2011,
 2012,
 2013,
 2014,
 2015,
 2016]
```

Apperantly, all years were recorded. Either the pilots or the planes got more used to rough weather or the data collection has changed over the years.

3.10. How many flight delays were there per category for 2013 and 2016?

In [439]:

```
match = {"$match": {
    "Time.Year": {"$in": [2005, 2015]}
}}

group = {"$group": {
    "_id": "$Time.Year",
    "total_delay_carrier": {"$sum": "$Statistics.# of Delays.Carrier"},
    "total_delay_late_aircraft": {"$sum": "$Statistics.# of Delays.Late Aircraft"},
    "total_delay_national_aviation_system": {"$sum": "$Statistics.# of Delays.Natior
    "total_delay_security": {"$sum": "$Statistics.# of Delays.Security"},
    "total_delay_weather": {"$sum": "$Statistics.# of Delays.Weather"}
}}

pipeline = [match, group]

result = airlines.aggregate(pipeline)
pd.DataFrame(result)
```

Out[439]:

	_id	total_delay_carrier	total_delay_late_aircraft	total_delay_national_aviation_system	total_de
0	2005	218705	269560	402032	
1	2015	194976	247441	240811	

3.11. Which Airport has the most delayed flights for a given year and how many?

In [440]:

```
years = [2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,2014,2015,2016]
count = 0

for y in years:
    match = {"$match": {
        "Time.Year": {"$eq": y}
    }}

    group = {"$group": {
        "_id": {
            "year": "$Time.Year",
            "airport_code": "$Airport.Code"
        },
        "total_flights_delayed": {"$sum": "$Statistics.Flights.Delayed"}
    }}

    project = {"$project": {
        "_id": 0,
        "year": "$_id.year",
        "airport_code": "$_id.airport_code",
        "total_flights_delayed": "$total_flights_delayed"
    }}

    sort = {"$sort": {"total_flights_delayed": -1}}

    limit = {"$limit": 1}

    pipeline = [match, group, project, sort, limit]
    result = airlines.aggregate(pipeline)
    if count == 0:
        df = pd.DataFrame(result)
    df_append = pd.DataFrame(result)
    df = df.append(df_append, ignore_index=True)
    count =+ 1

df
```

Out[440]:

	year	airport_code	total_flights_delayed
0	2003	ORD	51716
1	2004	ATL	103842
2	2005	ATL	105351
3	2006	ATL	105673
4	2007	ORD	109765
5	2008	ORD	95961
6	2009	ATL	107439
7	2010	ATL	77891
8	2011	ATL	69197
9	2012	ATL	56370
10	2013	ATL	72292

	year	airport_code	total_flights_delayed
11	2014	ORD	77824
12	2015	ORD	62123
13	2016	ATL	3984

4. Conclusion

Obviously, smaller airports have smaller delays in absolute numbers. However, the data shows that even smaller airports have a lot of delays. In the whole data set, there are only 18 months out of 157 recorded, where an airport had less than 500 delays. Hence, better prepare yourself for delays!

But there is good news too. The delays seemed to reduce in the years closer to 2016. Also, average delay time for Chicago Airport reduced steadily from 2003 till 2016. Maybe we are improving. Unfortunately, flights diverted stayed the same, but there are much fewer incidents compared to total delays.
However, if you are flying to the east coast in January, prepare for weather delays, because there is no month with as many weather delays as January. Unfortunately, it does not stop there, if you fly to JFK, you will have a chance to encounter weather delays all year long since there has been a different month with the most weather delays almost every year. And overall, chances that you are delayed if you fly to Chicago or Atlanta are the highest.

To end it on a positive note, if we compare the reasons for delay in 2005 to 2015, we can see a drastic decrease of delays due to the national aviation system. Maybe they have analysed the data for their flight plans and adjusted it accordingly, or maybe it was just coincidence. Let us assume the first one is true.

5. Lessons learned

During this project several new skills and methods have been learned by reading, practicing and applying them. The hands on activities throughout the course enforced repetition, which, in my opinion, lead to a steep learning curve.
During the project, I encountered the following challanges:

- Creating a MongoDB environment and accessing it through Jupyter
- Accessing an API (with authentication, even though it is not used in the final report)
- Finding suitable JSON data with an interesting structure
- Creating a class diagramm for a NoSQL database
- Using different aggregation statements

The work during the project helped me to create a better understanding of how MongoDB works in combination with Python.