Zeppelin

FINISHED

```
// Imports
import org.apache.spark.sql.functions._
import org.joda.time.format.DateTimeFormat

// Load data - adjust the path to the location of your data
val inputPath =  "/Users/Rhon/Desktop/Airline"
val airTraffic = sqlContext.read
        .format("com.databricks.spark.csv")
        .option("header", "true") // Use first line of all files as header
        .option("delimiter", ",")
        .option("inferSchema", "true") // Automatically infer data types
        .load(inputPath)
```

```
import org.apache.spark.sql.functions._
import org.joda.time.format.DateTimeFormat
inputPath: String = /Users/Rhon/Desktop/Airline
airTraffic: org.apache.spark.sql.DataFrame = [Year: int, Month: int ... 27 more fields]
```

FINISHED

```
  // Add extra features
val calcDayOfYear = udf(
    (dayOfMonth: Int, month: Int, year: Int) => {
        val dateFormat = DateTimeFormat.forPattern("dd/MM/yyyy")
        dateFormat.parseDateTime(s"$dayOfMonth/$month/$year").getDayOfYear()
    }
)
val calcRoute = udf(
    (origin: String, dest: String) => s"$origin - $dest"
)
val calcHourOfArrival = udf(
    (arrTime: String) => arrTime.slice(0,arrTime.size-2)
)
val featuredTraffic = airTraffic
    .withColumn("DayOfYear", calcDayOfYear(airTraffic("DayOfMonth"), airTraffic("Month"), a
    .withColumn("HourOfArr", calcHourOfArrival(airTraffic("ArrTime")))
    .withColumn("Route", calcRoute(airTraffic("Origin"), airTraffic("Dest")))
```

```
warning: Class org.joda.convert.FromString not found - continuing with a stub.
warning: Class org.joda.convert.ToString not found - continuing with a stub.
warning: Class org.joda.convert.ToString not found - continuing with a stub.
calcDayOfYear: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<
function3>,IntegerType,Some(List(IntegerType, IntegerType, IntegerType)))
calcRoute: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<func
tion2>,StringType,Some(List(StringType, StringType)))
calcHourOfArrival: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFuncti
on(<function1>,StringType,Some(List(StringType)))
featuredTraffic: org.apache.spark.sql.DataFrame = [Year: int, Month: int ... 30 more fields
]
```

FINISHED

```
// Register as Spark SQL Table
featuredTraffic.registerTempTable("air_traffic")
// sqlContext.cacheTable("air_traffic")
```

```
warning: there was one deprecation warning; re-run with -deprecation for details
```

FINISHED

```
%sql
select DayOfYear, count(*) as NrOfFlights, avg(DepDelay) as AvgDepDelay, avg(ArrDelay) as A
```

DayOfYear	NrOfFlights	AvgDepDelay
148	325,680	5.98483
243	334,157	6.48174
31	333,830	6.77604
85	336,519	7.22741
137	335,712	8.40268
251	334,225	4.09414
65	338,508	9.01227
53	336,107	11.65395
255	336,123	4.81319

FINISHED

```
%pyspark
# helper function to display in Zeppelin

import StringIO
def show(p):
  img = StringIO.StringIO()
  p.savefig(img, format='svg')
```

```
  img.seek(0)
  print "%html " + img_buf
```

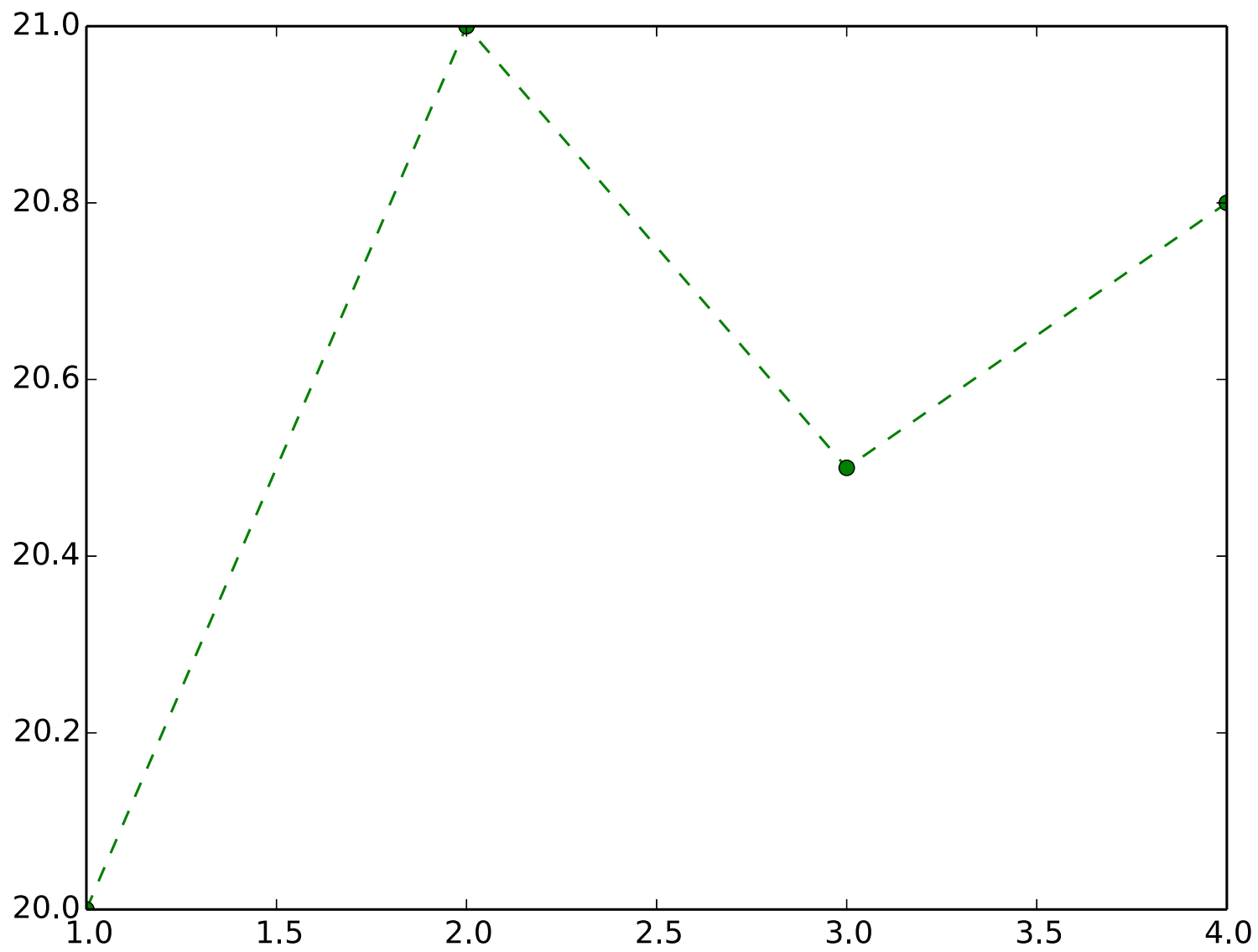
FINISHED

```
%pyspark
import matplotlib.pyplot as plt

#define some data
x = [1,2,3,4]
y = [20, 21, 20.5, 20.8]

#plot data
plt.plot(x, y, linestyle="dashed", marker="o", color="green")
```

```
[<matplotlib.lines.Line2D object at 0x1041252d0>]
```

FINISHED

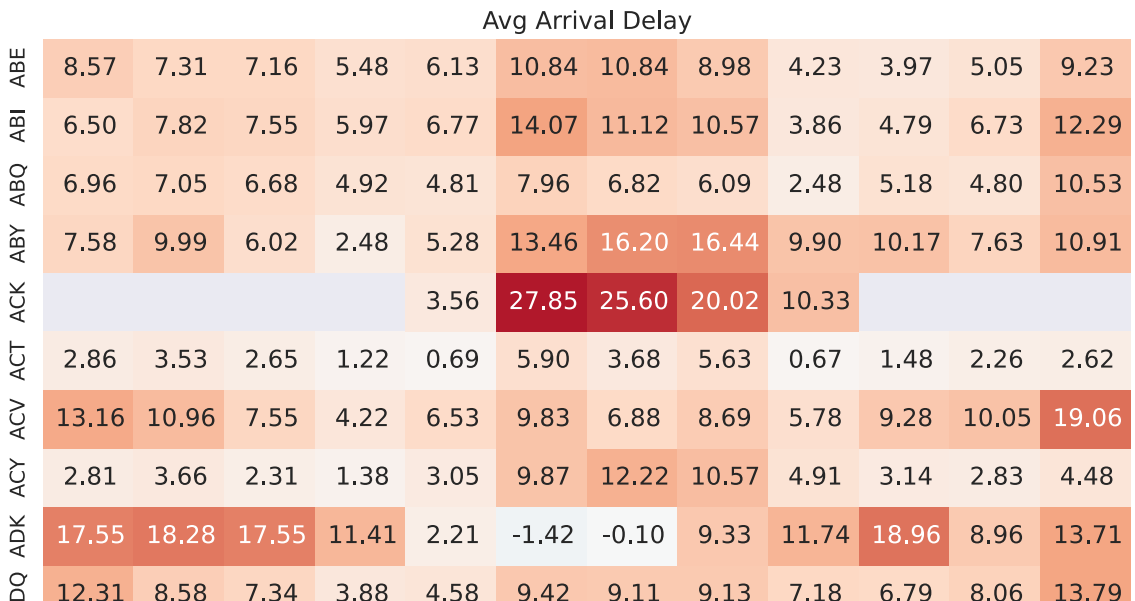
```
%pyspark
show(plt)
```



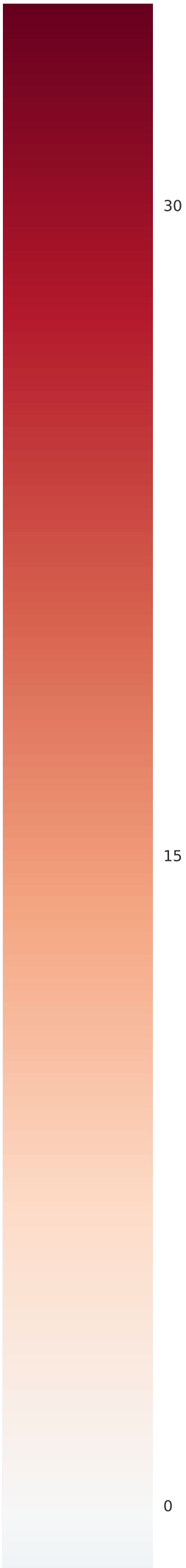
FINISHED

```
%pyspark
```

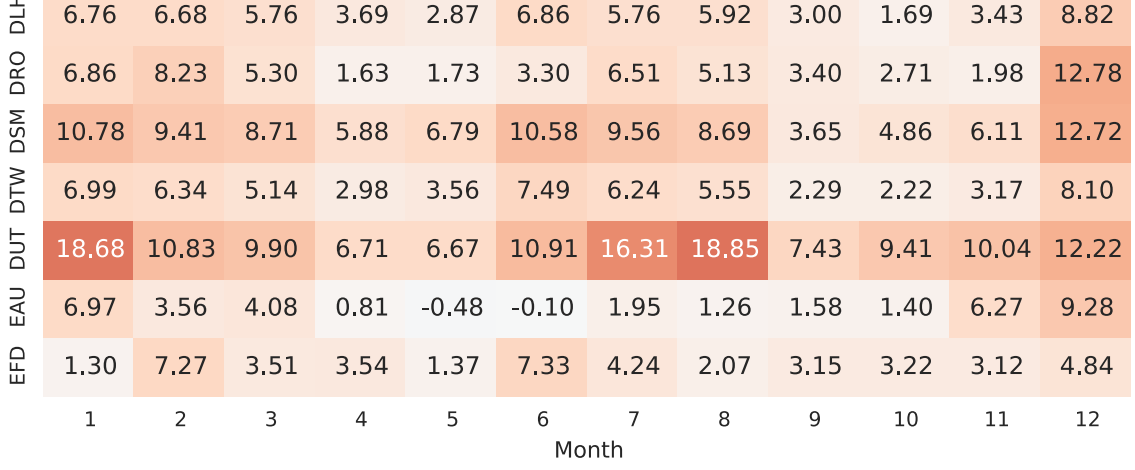
```
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns

import StringIO
def show(p):
  img = StringIO.StringIO()
  p.savefig(img, format='svg')
  img.seek(0)
  print "%html " + img.buf

df = sqlContext.sql("SELECT Dest, Month, count(*) as NrOfFlights, avg(ArrDelay) as AvgArrD
data = df.toPandas()

value = "AvgArrDelay"
x = "Dest"
grouping = ["Month"]

heatmap_data = data.pivot_table(values=value, index=x, columns=grouping)
heatmap_data = heatmap_data[0:100]

a4_dims = (len(heatmap_data.columns),50)
fig, ax = plt.subplots(figsize=a4_dims)
ax.set_title("Avg Arrival Delay")
sns.heatmap(heatmap_data, ax=ax, annot=True, fmt=".02f")

show(plt)
```



Dest	AEX	AGS	AKN	ALB	ALO	AMA	ANC	ANI	APF	ASE	ATL	ATW	AUS	AVL	AVP	AZO	BDL	BET	BFF	BFI	BFL	BGM	BGR	BHM	BIL	BIS	BJI	BLI	BMI	BNA	BOI	BOS	BPT	BQK	BQN	BRO	BRW	BTM	BTR	BTV	BUF
AEX	4.66	7.86	5.55	4.29	4.32	14.71	12.41	11.78	4.76	6.44	6.79	9.97																														
AGS	7.64	8.17	8.01	7.91	7.15	12.44	15.24	11.71	7.20	6.56	7.04	8.79																														
AKN	11.55	17.78	15.10	13.59	7.56	12.74	8.44	9.98	9.66	12.31	7.90	18.34																														
ALB	8.79	8.41	9.02	6.61	7.43	12.26	12.89	11.19	4.86	5.30	5.38	10.19																														
ALO	4.59	4.08	1.38	2.96	-4.20	7.27	5.55	5.94	-3.33	0.65	6.28	15.09																														
AMA	6.80	7.71	8.43	6.57	7.63	11.47	7.76	6.96	4.08	6.86	6.19	11.43																														
ANC	12.11	9.75	8.84	3.82	6.55	9.65	10.65	12.14	7.83	7.97	10.45	16.70																														
ANI	21.11	15.52	9.07	15.88	9.36	15.80	12.26	5.81	4.60	5.69	17.92	14.14																														
APF	9.19	9.87	9.06	4.91	3.89	12.84	19.89	17.18	7.57	5.73	7.21	9.63																														
ASE	17.90	15.62	10.85	1.45	0.28	7.24	8.76	7.03	7.92	5.25	-1.43	24.19																														
ATL	9.42	9.81	8.28	5.77	5.58	10.68	11.69	9.58	5.65	6.60	6.59	11.17																														
ATW	12.93	13.86	9.46	7.07	5.28	11.55	11.83	10.38	6.13	6.34	6.61	18.32																														
AUS	6.72	6.62	6.70	5.04	5.47	8.76	7.56	6.28	2.19	4.74	4.72	9.46																														
AVL	5.97	5.43	5.34	4.67	5.40	12.23	13.23	11.48	5.79	6.76	4.90	8.76																														
AVP	10.36	8.68	8.69	7.01	7.86	13.94	13.84	12.49	5.98	5.78	4.61	11.36																														
AZO	10.75	9.42	7.29	4.45	4.23	7.24	6.14	5.72	3.15	2.99	5.11	11.93																														
BDL	7.10	6.83	7.98	5.64	5.88	11.06	10.55	9.00	3.36	3.51	4.06	8.21																														
BET	10.04	9.62	7.41	4.35	5.61	8.45	9.04	14.29	6.17	8.36	11.81	14.23																														
BFF																																									
BFI																																									
BFL	7.47	5.95	4.71	3.15	3.11	5.80	4.30	3.77	2.29	4.59	2.80	9.80																														
BGM	7.42	6.43	5.73	3.94	5.35	9.47	7.69	8.05	4.49	3.10	3.78	8.94																														
BGR	11.80	10.68	10.93	8.98	9.84	14.66	14.22	13.33	5.67	5.15	5.34	12.70																														
BHM	6.81	6.39	6.37	4.92	5.87	9.67	9.23	6.84	3.05	4.36	4.76	9.17																														
BIL	8.11	6.91	5.55	2.65	2.98	7.33	8.04	6.18	2.41	3.24	4.13	12.72																														
BIS	6.49	6.36	5.79	1.86	2.14	7.22	4.35	3.85	1.04	1.57	3.72	11.29																														
BJI						2.80	0.91	4.14	9.46	2.00	-7.00																														
BLI	6.42	4.65	2.43	1.66	1.74	3.73	2.13	6.65	3.10	6.59	8.86	14.54																														
BMI	16.15	17.10	14.75	8.36	7.48	15.43	15.75	13.13	8.11	8.18	10.32	18.37																														
BNA	5.56	5.17	4.87	3.39	4.05	7.94	7.28	5.70	1.87	3.25	3.35	7.55																														
BOI	10.94	9.11	7.49	4.39	4.67	8.36	8.15	8.27	3.60	5.73	6.39	15.51																														
BOS	8.65	8.05	9.26	7.41	7.75	12.53	11.75	10.59	5.34	5.45	5.73	9.00																														
BPT	1.30	3.12	2.30	1.07	1.05	10.48	5.37	3.82	0.38	0.17	4.82	1.57																														
BQK	4.76	9.24	8.77	5.23	9.13	16.03	19.26	19.80	11.96	12.37	8.23	12.42																														
BQN	7.88	8.99	10.32	9.59	8.57	18.56	20.01	13.79	4.30	1.14	5.86	12.97																														
BRO	4.71	6.41	7.15	3.87	4.51	10.51	7.18	4.12	-1.51	3.87	5.68	9.56																														
BRW	7.83	9.27	7.81	2.73	7.54	9.46	12.64	13.26	5.55	5.49	6.13	11.71																														
BTM	10.29	5.82	4.87	1.16	0.26	4.33	2.74	2.78	-0.26	0.44	3.52	15.56																														
BTR	7.22	8.61	7.89	6.49	6.03	10.73	9.68	7.73	4.13	4.95	6.18	9.84																														
BTV	9.42	9.49	11.18	7.41	7.82	12.89	15.24	13.12	4.81	4.85	5.18	12.76																														
BUF	9.12	8.81	8.31	7.02	6.52	12.01	12.50	10.62	4.72	5.04	5.76	11.34																														



BUR	7.43	8.04	6.07	5.05	3.90	6.23	5.54	6.47	3.54	5.92	6.38	10.35
BWI	6.62	5.88	6.67	4.67	5.85	11.26	10.97	8.73	2.88	3.31	4.41	7.77
BZN	11.55	9.19	6.23	1.05	1.83	6.87	6.11	5.61	1.55	1.44	3.28	14.89
CAE	10.05	9.74	9.20	7.34	7.42	12.42	13.29	11.08	5.78	5.82	6.99	12.07
CAK	9.24	10.42	7.47	4.54	5.09	10.64	11.86	10.04	3.77	5.12	5.91	12.52
CBM				0.00								
CCR	2.64	1.13	1.95	-1.32	0.49	2.03	0.75	5.94	2.77	7.50	7.05	7.49
CDC	5.11	6.13	2.56	0.02	0.33	1.34	2.66	0.59	1.39	0.24	2.64	6.99
CDV	11.89	10.54	8.35	7.67	9.40	13.43	14.23	19.31	14.81	8.43	8.77	17.78
CEC	11.64	7.51	6.84	1.88	2.92	6.85	5.46	8.28	4.31	6.88	11.16	16.20
CHA	6.93	6.44	5.59	4.52	5.05	9.76	10.34	7.73	4.62	4.52	5.09	8.75
CHO	6.60	5.66	3.69	3.16	3.76	7.55	9.79	8.55	4.47	4.06	5.00	4.88
CHS	7.89	8.21	8.52	6.78	6.63	10.63	11.26	8.80	4.68	5.58	5.76	10.19
CIC	14.35	12.15	7.26	6.39	8.80	12.11	10.92	7.83	8.24	11.36	14.37	18.27
CID	11.40	8.95	8.77	6.04	7.07	10.39	9.21	7.99	3.06	3.48	5.03	12.64
CKB	7.83											
CLD	3.11	3.14	2.50	0.60	1.19	2.26	1.96	0.71	1.97	3.39	0.50	1.85
CLE	7.95	6.92	6.62	4.45	5.27	9.57	8.83	7.44	2.72	3.60	4.53	9.56
CLL	3.57	4.42	2.42	1.19	2.11	7.55	4.39	6.11	-0.72	1.04	2.26	3.36
CLT	5.85	5.44	5.52	3.97	3.77	7.86	7.73	6.28	2.19	2.67	3.62	7.11
CMH	9.53	8.03	8.16	6.25	6.99	11.46	11.01	9.25	4.28	4.74	5.94	11.03
CMI	18.43	13.91	13.38	9.01	10.80	13.61	14.16	12.84	5.83	7.15	10.47	20.84
CMX	12.46	11.20	-6.45	10.49	1.79	5.06	24.55	13.74	12.11	-0.40	6.33	34.69
COD	9.18	10.99	3.60	-0.82	0.28	2.08	3.57	1.66	-0.13	4.24	0.38	13.77
COS	8.43	7.79	7.08	4.82	5.69	10.20	8.96	7.47	3.67	4.49	5.01	11.64
CPR	7.98	6.33	3.13	0.91	1.79	3.05	4.52	3.44	1.92	2.38	1.54	11.80
CRP	6.54	7.41	6.83	5.79	5.68	9.55	6.79	5.34	2.76	5.49	5.14	9.06
CRW	7.51	7.28	6.80	4.82	5.79	10.87	10.84	9.33	5.15	5.14	5.38	10.05
CSG	7.47	10.11	7.05	6.11	6.59	11.51	15.14	13.90	9.68	8.60	7.27	8.43
CVG	6.65	5.40	4.34	2.63	3.56	7.02	6.78	5.57	1.35	2.53	3.68	7.85
CWA	18.80	19.46	17.10	12.73	6.35	14.57	10.47	9.42	9.30	7.11	6.90	28.45
CYS												
DAB	9.67	9.99	9.11	6.44	5.52	10.05	10.99	10.16	5.38	6.40	7.41	11.31
DAL	4.02	5.29	6.46	5.05	6.42	8.51	5.65	4.75	2.87	5.65	4.86	7.34
DAY	8.67	8.55	7.92	5.52	6.40	9.74	9.55	7.90	3.68	4.40	5.92	11.04
DBQ	16.84	12.96	14.83	8.37	12.13	11.55	10.95	8.01	4.23	6.36	7.73	19.73
DCA	5.87	5.03	5.81	4.18	4.98	9.44	8.48	6.45	3.19	3.14	3.73	6.61
DEN	8.83	8.53	7.45	4.41	5.48	8.73	7.78	6.33	2.19	3.81	4.31	12.16
DET	5.63	5.26	5.33	3.53	4.26	5.69	4.20	6.33	4.26	5.79	5.38	7.09
DFW	6.79	5.97	5.39	4.47	4.49	8.72	6.68	5.50	1.37	4.12	3.10	8.86
DHN	9.70	12.85	10.81	7.31	8.50	19.61	25.23	22.63	9.79	12.52	11.68	12.98
DLG	12.38	14.74	10.48	8.52	6.78	13.66	12.78	11.60	8.59	5.85	9.79	16.28



FINISHED

```
%pyspark
from wordcloud import WordCloud

import StringIO
def show(p):
  img = StringIO.StringIO()
  p.savefig(img, format='svg')
  img.seek(0)
  print "%html " + img.buf

# Create route frequencies
routes = sqlContext.sql("SELECT Route, count(*) as Count FROM air_traffic GROUP BY Route")
route_freq = [(x[0],x[1]) for x in routes]

# Generate word cloud image
wordcloud = WordCloud().generate_from_frequencies(route_freq)
image = wordcloud.to_image()
image.show()
```

READY