**Big Data – Cast Study**

As a Big Data engineer you are expected to build/test & support large-scale data processing systems and be an expert in data ingestion, modelling & processing solutions to meet business needs of the organisation. You should be able to work with the latest Big Data and NoSQL database technologies to deliver cutting edge solutions. You should be creative problem solvers, resourceful in getting things done and productive working independently or collaboratively and well versed in Agile methodology and behaviours.

**Use-Case: Airline on-time performance**

**Reference Link:** <http://stat-computing.org/dataexpo/2009/>

Have you ever been stuck in an airport because your flight was delayed or cancelled and wondered if you could have predicted it if you'd had more data? This is your chance to find out.

**The data:**

The data consists of flight arrival and departure details for all commercial flights within the USA, from October 1987 to April 2008. This is a large dataset: there are nearly 120 million records in total, and takes up 1.6 gigabytes of space compressed and 12 gigabytes when uncompressed.

**Environment:**

* **Option1:** We recommend using Google Cloud Big Data platform (dataproc) to address this uses cases. The attached document has the guidelines on how to create Google cloud account and provision a new Hadoop cluster. When you first sign-in, you will be provided with a 300$ credit to use for 365 days, which will be highly sufficient to develop the use cases mentioned.
* **Option2:** Use your own environment



1. **Batch Ingestion & Processing**
   1. **Hadoop directory Structure to be created.**

|  |  |  |
| --- | --- | --- |
| **Layer** | **Directory Path** | **File Format** |
| RAW | /data/raw/ | As is (e.g. TXT, CSV, XML, JSON, etc.,) |
| Decomposed | /data/decomposed/ | Avro |
| Modelled | /data/modelled/ | Parquet |
| Schema (Meta data) | /data/schema/ | AVSC schema |

* 1. **Source data details**

Download the stats created for year 2008 & 2007.

<http://stat-computing.org/dataexpo/2009/the-data.html>

**Supplemental Data:** <http://stat-computing.org/dataexpo/2009/supplemental-data.html>

* 1. **Data preparation**
  + Create a Kafka cluster
  + Create the following Topics in Kafka
    - Airports
    - Carriers
    - Planedate
    - OTP
  + Download the stats created for year 2008 & 2007 and load the data into a Kafka cluster under the relevant topics. Use any options of your choice to load the data to Kafka topics.
  1. **Batch Ingestion (HDFS)**
* **Raw layer (Store data AS-IS)**
  + Use *Apache Flume* to consume messages from Airports & Planedate Kafka Topic to HDFS Raw folder
  + Use *Spark Streaming* to consume messages from Carriers and OTP Kafka Topic to HDFS Raw folder
* **Decomposed layer (Append UUID and timestamp to the AS-IS data)**
  + For each message in the Airports & Planedate data from raw directory, append UUID and timestamp using Pig Latin.
  + For each message in the Carriers & OTP data from raw directory, append UUID and timestamp using Pig Latin.
  1. **Modelling and processing**
* Cleanse the data (trim, null, removing duplicates) and load it in Parquet format as modelled using Spark/Scala
  1. **Develop a solution in Hive, Pig and Spark/Scala to answer the following questions. Demonstrate your expertise in each tools.**
* Which carrier performs better?
* When is the best time of day/day of week/time of year to fly to minimise delays?
* Do older planes suffer more delays?
* Can you detect cascading failures as delays in one airport create delays in others? Are there critical links in the system? Airtime ,arrtime , dest,depdelay, origin

1. **Speed Ingestion & Processing**

**2.1 Data preparation**

* Use the data already loaded as part of the section 1.3
  1. **Speed Ingestion and Modelling**
* Model in Elasticsearch (Index, Type, Partition)
* Consume the data from Kafka topic and load in to speed (Elasticsearch) using Spark streaming

**2.3 Develop Restful service (Read data from Elasticsearch Index), Answer any three of the following questions…**

* Which carrier performs better?
* When is the best time of day/day of week/time of year to fly to minimise delays?
* Do older planes suffer more delays?
* Can you detect cascading failures as delays in one airport create delays in others? Are there critical links in the system?

**Note:**

Prepare a technical presentation to explain the solution and also share the code & configuration.

**Solution:**

**Hadoop Environment: Cloudera 5.12 single node cluster**

**Ingestion Process:**

Flume configuration file for data file ingestion process from local Flume dir to HDFS flume\_target:

# Flume configuration: **spoolsrcagent\_hdfs.conf**

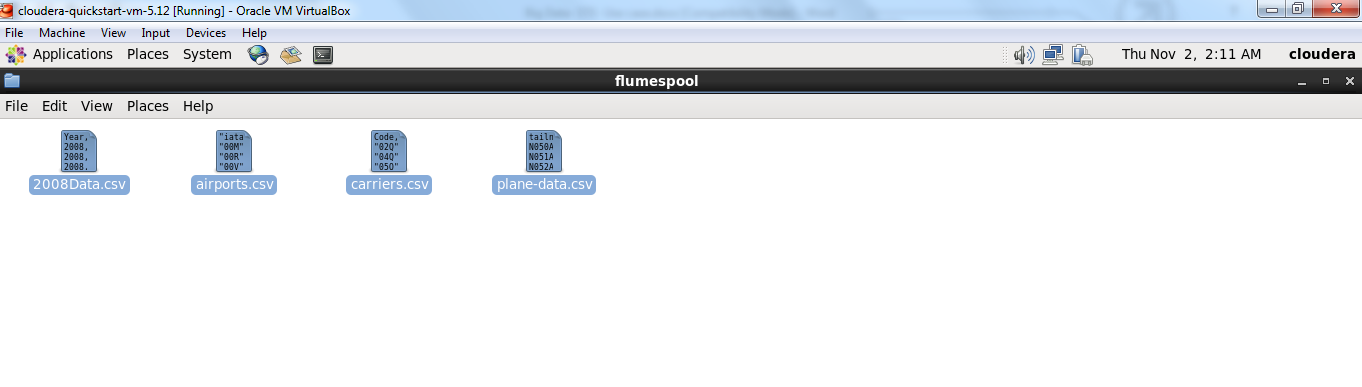
# Name the components on this agenta1.sources = src1a1.sinks = snk1a1.channels = ch1# Describe/configure the sourcea1.sources.src1.type = spooldira1.sources.src1.spoolDir = /home/cloudera/flumespool

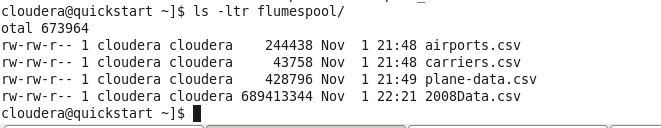
# Describe the sinka1.sinks.snk1.type = hdfsa1.sinks.snk1.channel = ch1

a1.sinks.snk1.hdfs.path=/user/cloudera/flume\_target# Use a channel which buffers events in memorya1.channels.ch1.type = memorya1.channels.ch1.capacity = 1000a1.channels.ch1.transactionCapacity = 100

# Bind the source and sink to the channelda1.sources.src1.channels = ch1a1.sinks.snk1.channels = ch1

**Copy file from download dir to flumespool dir:**





Load airport.csv data from local flumespool dir to HDFS using Flume ingestion process.

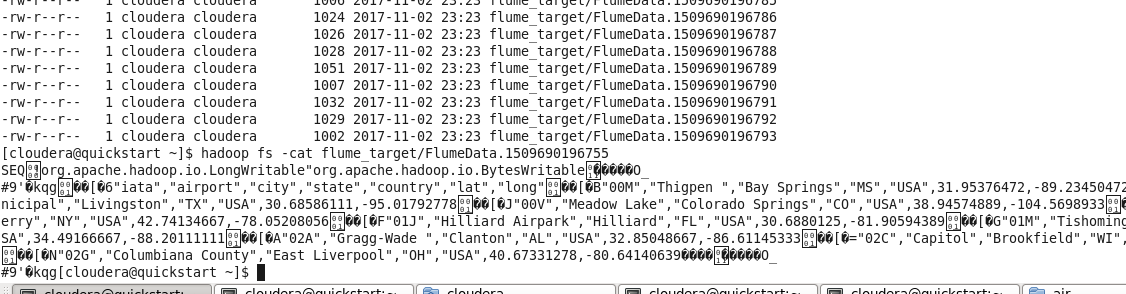
**Run flume-ng command @ cloudera**

flume-ng agent --conf-file spoolsrcagent\_hdfs.conf --name a1 -Dflume.root.logger=INFO,console

**air/airport/airports.csv**

Airport data @Flume\_target dir at hdfs:

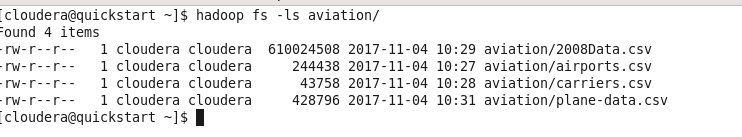
**Data transpose in the form of AVRO file format using Flume-ng agent as mention above.**



**@pig grunt: if I am using a Flumedata.xxxxxxx file at pig grunt shell**

a= load 'flume\_target/' using PigStorage (',') as (a1:chararray,a2:chararray, a3:chararray, a4:chararray,a5:chararray,a6:chararray,a7:chararray);

Due to space and a memory crunch, I am copying file from local to HDFS using shell scripts.



**Loading all four file data from pig:**

**Q1: Which carrier performs better?** Considering a top 5 best carrier who made a maximum round trip.

**F = load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/2008Data.csv' using PigStorage (',') as (p1:chararray,p2:chararray, p3:chararray, p4:chararray,p5:chararray,p6:chararray,p7:chararray, p8:chararray,p9:chararray, p10:chararray, p11:chararray, p12:chararray, p13:chararray, p14:chararray, p15:chararray, p16:chararray, p17:chararray, p18:chararray, p19:chararray, p20:chararray, p21:chararray, p22:chararray, p23:chararray, p24:chararray, p25:chararray, p26:chararray, p27:chararray, p28:chararray, p29:chararray);**

F1 = FILTER F BY (p9 is not null);

groupF1 = GROUP F1 by p9;

countF1 = FOREACH groupF1 GENERATE group, COUNT(F1) as carr\_count;

F2 = order countF1 by carr\_count DESC;

**carr= load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/carriers.csv' using PigStorage (',') as (c1:chararray,c2:chararray);**

C = FILTER carr BY c1 is not null and c2 is not null;

C1 = foreach C generate (chararray)$0, (chararray)$1;

C2 = join F2 by $0, C1 by $0;

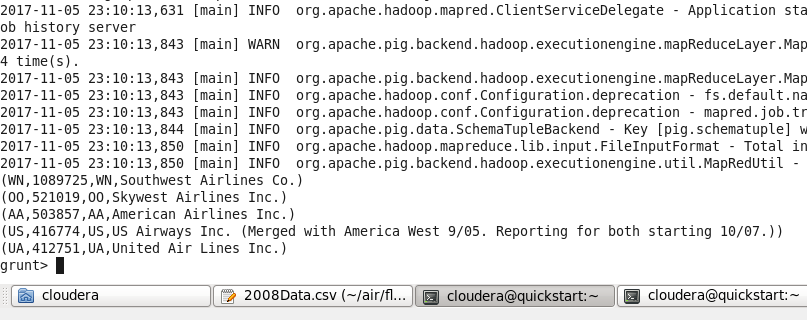
Final = foreach C2 generate $0,$1,$3;

Final\_Result = ORDER Final by $1 DESC;

C3 = limit Final\_Result 5;

dump C3;

Note: Find out the group count of UniqueCarrier (P9)



**Analysis: For better performance we can considered maximum round trip made by any carriers. As per result WN “Southwest Airlines” made a remarkable performance.**

**Q2. When is the best time of day/day of week/time of year to fly to minimise delays?**

**F = load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/2008Data.csv' using PigStorage (',') as (**p1:chararray,p2:chararray, p3:chararray, p4:chararray,p5:chararray,p6:chararray,p7:chararray, p8:chararray,p9:chararray, p10:chararray, p11:chararray, p12:chararray, p13:chararray, p14:chararray, p15:chararray, p16:chararray, p17:chararray, p18:chararray, p19:chararray, p20:chararray, p21:chararray, p22:chararray, p23:chararray, p24:chararray, p25:chararray, p26:chararray, p27:chararray, p28:chararray, p29:chararray);

BB = foreach F generate (int)p3 as daymonth , (int)p4 as dayweek , (int)p5 as dep\_time ,(int)p16 as dep\_delay, (chararray)p17 as origin;

C2 = filter BB by (dep\_delay is not null) AND (origin is not null);

D2 = group C2 by origin;

E2 = foreach D2 generate group,AVG(C2.dep\_delay),MIN(C2.daymonth),MIN(C2.dayweek),MIN(C2.dep\_time);

F = order E2 by $1 DESC;

G = limit F 10;

**air = load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/airports.csv' using PigStorage (',') as** (a1:chararray,a2:chararray, a3:chararray, a4:chararray,a5:chararray,a6:chararray,a7:chararray);

A1 = foreach **air** generate (chararray)$0 as origin, (chararray)$2 as city, (chararray)$4 as country;

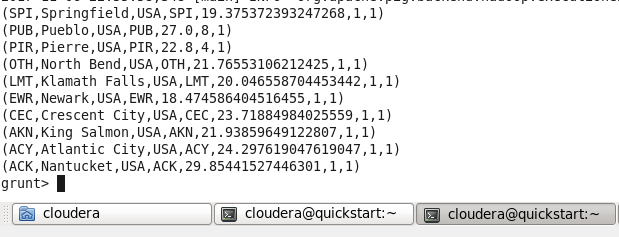
Joined = join A1 by origin, G by $0;

Final = foreach Joined generate $0,$1,$2,$3,$4,$5,$6;

Final\_Result = ORDER Final by $3 DESC;

dump Final\_Result;

Ans:



**Analysis: As per analysis delay might be possible through the year but if we avoid 1st week of every mount that would be good for because majority of delay happen during this time. That is my perceptions based on data analysis.**

**Q3. Do older planes suffer more delays? Year and tellnum**

**FD = load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/2008Data.csv' using PigStorage (',') as (**p1:chararray,p2:chararray, p3:chararray, p4:chararray,p5:chararray,p6:chararray,p7:chararray, p8:chararray,p9:chararray, p10:chararray, p11:chararray, p12:chararray, p13:chararray, p14:chararray, p15:chararray, p16:chararray, p17:chararray, p18:chararray, p19:chararray, p20:chararray, p21:chararray, p22:chararray, p23:chararray, p24:chararray, p25:chararray, p26:chararray, p27:chararray, p28:chararray, p29:chararray);

X = foreach FD generate (chararray) p11 as tailnum, (int) p15 as arr\_delay, (int) p16 as dep\_delay, (chararray) p17 as origin, (chararray) p18 as dest;

X1 = filter X by (tailnum is not null) AND (dep\_delay is not null) AND (origin is not null);

X2 = group X1 by (tailnum);

X3 = foreach X2 generate group, AVG(X1.dep\_delay),MIN(X1.origin);

X4 = order X3 by $1 DESC;

**plane = load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/plane-data.csv' using PigStorage (',') as (p1:chararray,p2:chararray, p3:chararray, p4:chararray,p5:chararray,p6:chararray,p7:chararray, p8:chararray,p9:chararray);**

K = filter **plane** by (p1 is not null) AND (p5 is not null) AND (p9 is not null);

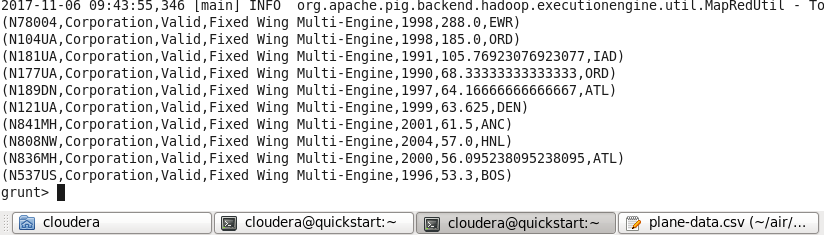
Joined = join K by $0, X4 by $0;

Final = foreach Joined generate $0,$1,$5,$6,$8,$10,$11;

Final\_Result = ORDER Final by $5 DESC;

Z1 = limit Final\_Result 10;

Dump Z1;



**Analysis: As per analysis older planes took more time in departure. As per result, average delay time is more than 60 min.**

**Q4: Can you detect cascading failures as delays in one airport create delays in others? Are there critical links in the system?**

**D = load 'hdfs://quickstart.cloudera:8020/user/cloudera/aviation/2008Data.csv' using PigStorage (',') as (**p1:chararray,p2:chararray, p3:chararray, p4:chararray,p5:chararray,p6:chararray,p7:chararray, p8:chararray,p9:chararray, p10:chararray, p11:chararray, p12:chararray, p13:chararray, p14:chararray, p15:chararray, p16:chararray, p17:chararray, p18:chararray, p19:chararray, p20:chararray, p21:chararray, p22:chararray, p23:chararray, p24:chararray, p25:chararray, p26:chararray, p27:chararray, p28:chararray, p29:chararray);

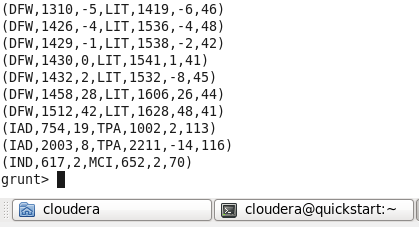
D1 = foreach D generate (chararray)p17 as origin,(int)p5 as dep\_time, (int)p16 as dep\_delay, (chararray)p18 as dest, (int)p7 as arr\_time, (int)p15 as arr\_delay, (int)p14 as air\_time;

C2 = filter D1 by (dep\_delay is not null) AND (origin is not null) AND (arr\_time is not null) AND (dest is not null);

G1 = limit C2 10;

Dump G1;

**Origin, DepTime, Departure\_Delay, Dest, Arrivaltime, Arrival\_Delay, Airtime**



**Analysis: delay arrival and delay departure are dependent variable if departure is delay due to any reason that may impact arrival at destination. Recover time may be in a couple of min but airtime have very less variation. As per the figure if average airtime is 45 min from DEW to LIT, hardly 1 to 5 min variation.**