Module 6

Understanding Pig and Impala

Thanachart Numnonda, Executive Director, IMC Institute

Thanisa Numnonda, Faculty of Information Technology,

King Mongkut's Institute of Technology Ladkrabang

Introduction

A high-level platform for creating MapReduce programs Using Hadoop

Pig is a platform for analyzing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs. The salient property of Pig programs is that their structure is amenable to substantial parallelization, which in turns enables them to handle very large data sets.

Pig Components

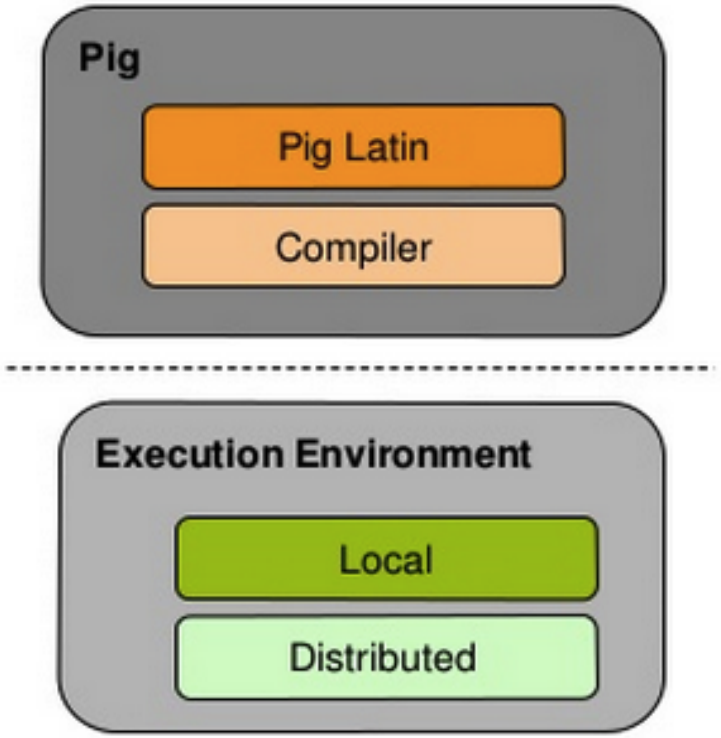
- **Two Components**
  - Language (Pig Latin)
  - Compiler

- **Two Execution Environments**
  - **Local**

    pig -x local

  - **Distributed**

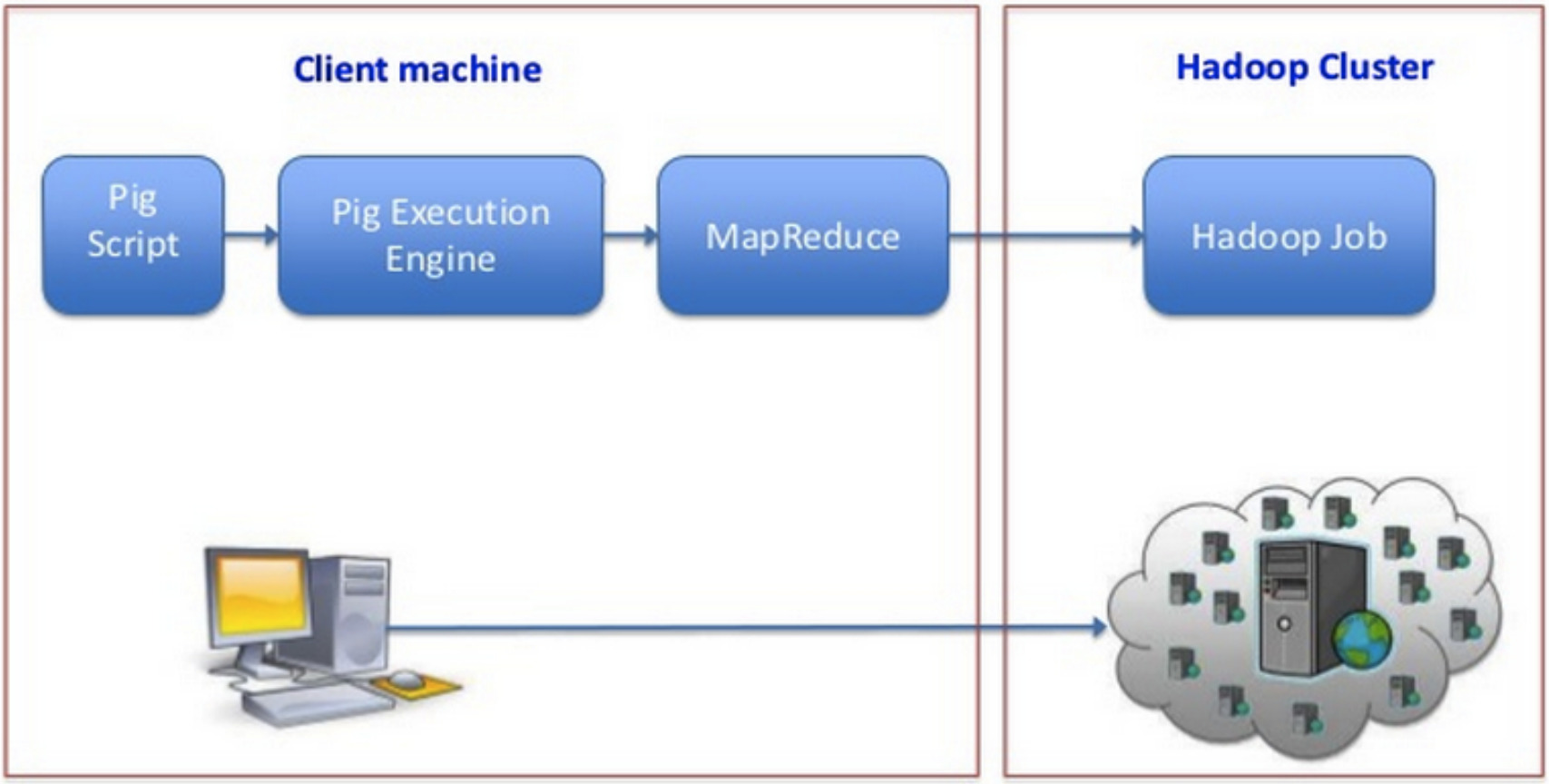
    pig -x mapreduce



Pig Latin

```
Users = load 'users' as (name, age);
Fltrd = filter Users by age >= 18 and age <= 25;
Pages = load 'pages' as (user, url);
Jnd = join Fltrd by name, Pages by user;
Grpd = group Jnd by url;
Smmd = foreach Grpd generate group, COUNT(Jnd) as clicks;
Srtd = order Smmd by clicks desc;
Top5 = limit Srtd 5;
store Top5 into 'top5sites';
```

Pig Execution Stages



Why Pig?

- **Makes writing Hadoop jobs easier**
  - 5% of the code, 5% of the time
  - You don't need to be a programmer to write Pig scripts

- **Provide major functionality required for DatawareHouse and Analytics**
  - *Load, Filter, Join, Group By, Order, Transform*

- **User can write custom UDFs (User Defined Function)**

Pig v.s. Hive

Characteristic	Pig	Hive
Developed by	Yahoo!	Facebook
Language name	Pig Latin	HiveQL
Type of language	Data flow	Declarative (SQL dialect)
Data structures it operates on	Complex, nested	
Schema optional?	Yes	No, but data can have many schemas
Relational complete?	Yes	Yes
Turing complete?	Yes when extended with Java UDFs	Yes when extended with Java UDFs

Hands-On: Running a Pig script

Starting Pig Command Line

```
$  pig -x mapreduce
2013-08-01 10:29:00,027 [main] INFO  org.apache.pig.Main - Apache Pig
version 0.11.1 (r1459641) compiled Mar 22 2013, 02:13:53
2013-08-01 10:29:00,027 [main] INFO  org.apache.pig.Main - Logging error
messages to: /home/hdadmin/pig_1375327740024.log
2013-08-01 10:29:00,066 [main] INFO  org.apache.pig.impl.util.Utils -
Default bootup file /home/hdadmin/.pigbootup not found
2013-08-01 10:29:00,212 [main] INFO
org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting
to hadoop file system at: file:///
grunt>
```

Writing a Pig Script for wordcount

```
A = load '/user/cloudera/input/*';
B = foreach A generate flatten(TOKENIZE((chararray)$0)) as
word;
C = group B by word;
D = foreach C generate COUNT(B), group;
store D into '/user/cloudera/output/wordcountPig';
```

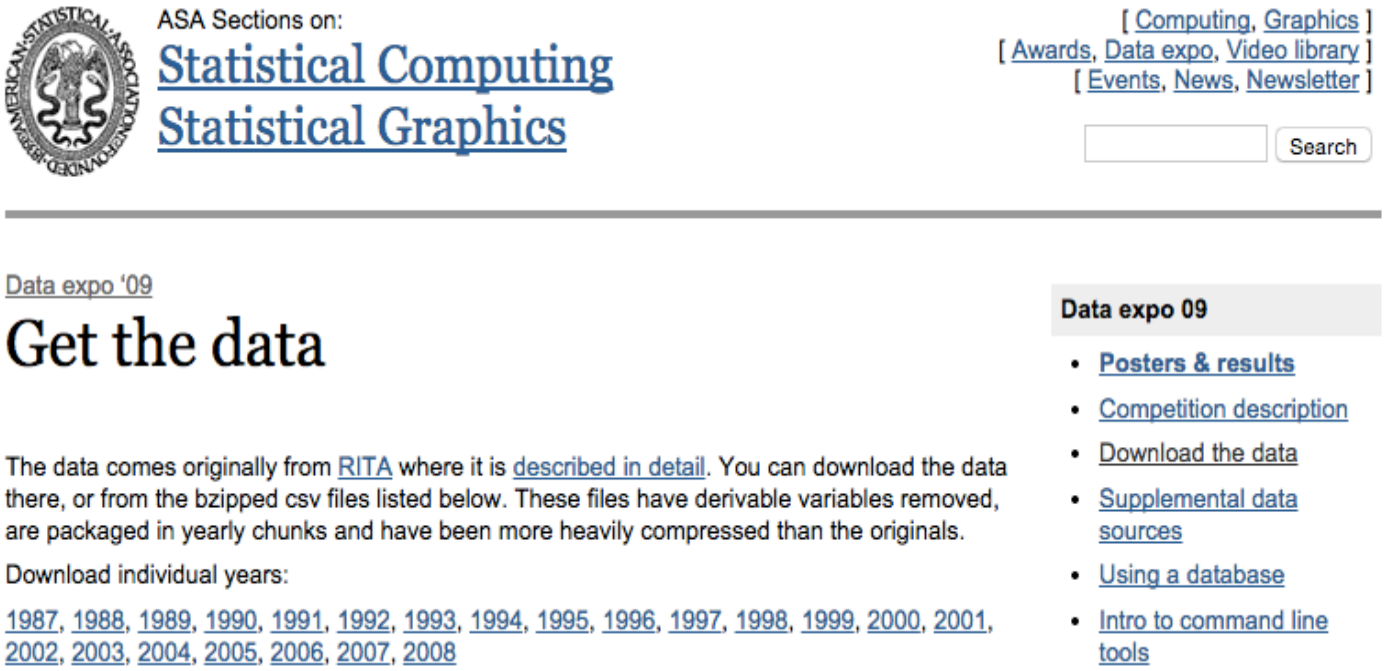
```
2016-06-14 08:29:25,835 [main] INFO  org.apache.pig.backend.hadoop.executionengine.m
apReduceLayer.MapReduceLauncher - More information at: http://localhost:50030/jobdet
ails.jsp?jobid=job_1465875170640_0004
2016-06-14 08:29:25,871 [main] INFO  org.apache.pig.backend.hadoop.executionengine.m
apReduceLayer.MapReduceLauncher - 0% complete
2016-06-14 08:29:41,625 [main] INFO  org.apache.pig.backend.hadoop.executionengine.m
apReduceLayer.MapReduceLauncher - 50% complete
2016-06-14 08:29:51,359 [main] INFO  org.apache.pig.backend.hadoop.executionengine.m
apReduceLayer.MapReduceLauncher - 100% complete
2016-06-14 08:29:51,362 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStats -
Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.7.0  0.12.0-cdh5.7.0 root    2016-06-14 08:29:18     2016-06-14 08:29:51G
ROUP_BY

Success!
```

Home / user / cloudera / output / wordcountPig / **part-r-00000**

```
70      '
21      -
37      .
20      1
41      2
19      3
6       4
3       5
2       6
2       7
2       8
1       9
2       :
10      ;
2       =
447     A
4       B
2       C
2       D
4026    I
42      O
34      V
```

Flight Details Data

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



ASA Sections on:
Statistical Computing
Statistical Graphics

[Computing, Graphics]
[Awards, Data expo, Video library]
[Events, News, Newsletter]

Search

Data expo '09

Get the data

The data comes originally from RITA where it is described in detail. You can download the data there, or from the bzipped csv files listed below. These files have derivable variables removed, are packaged in yearly chunks and have been more heavily compressed than the originals.

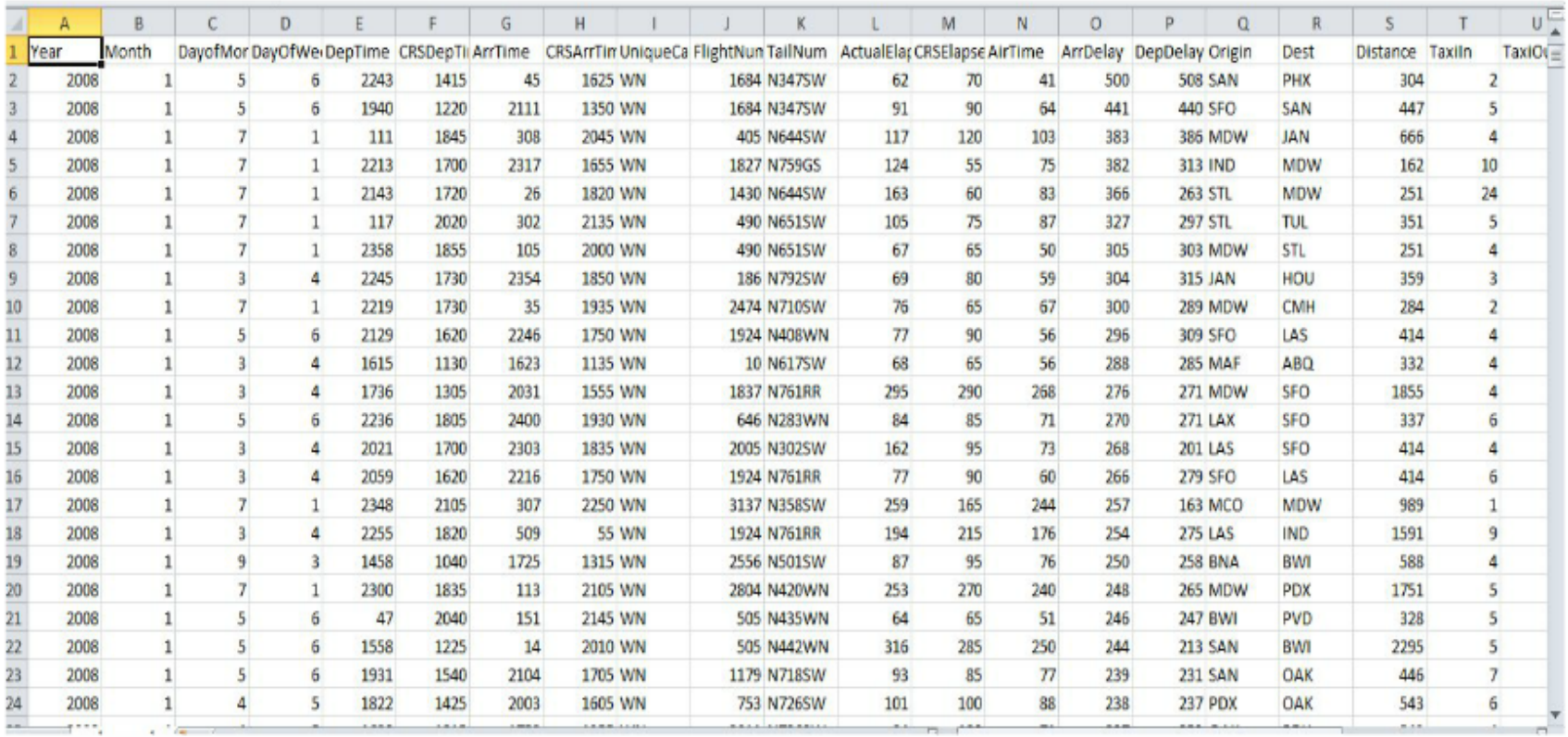
Download individual years:

1987, 1988, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008

Data expo 09

- **Posters & results**
- Competition description
- Download the data
- Supplemental data sources
- Using a database
- Intro to command line tools

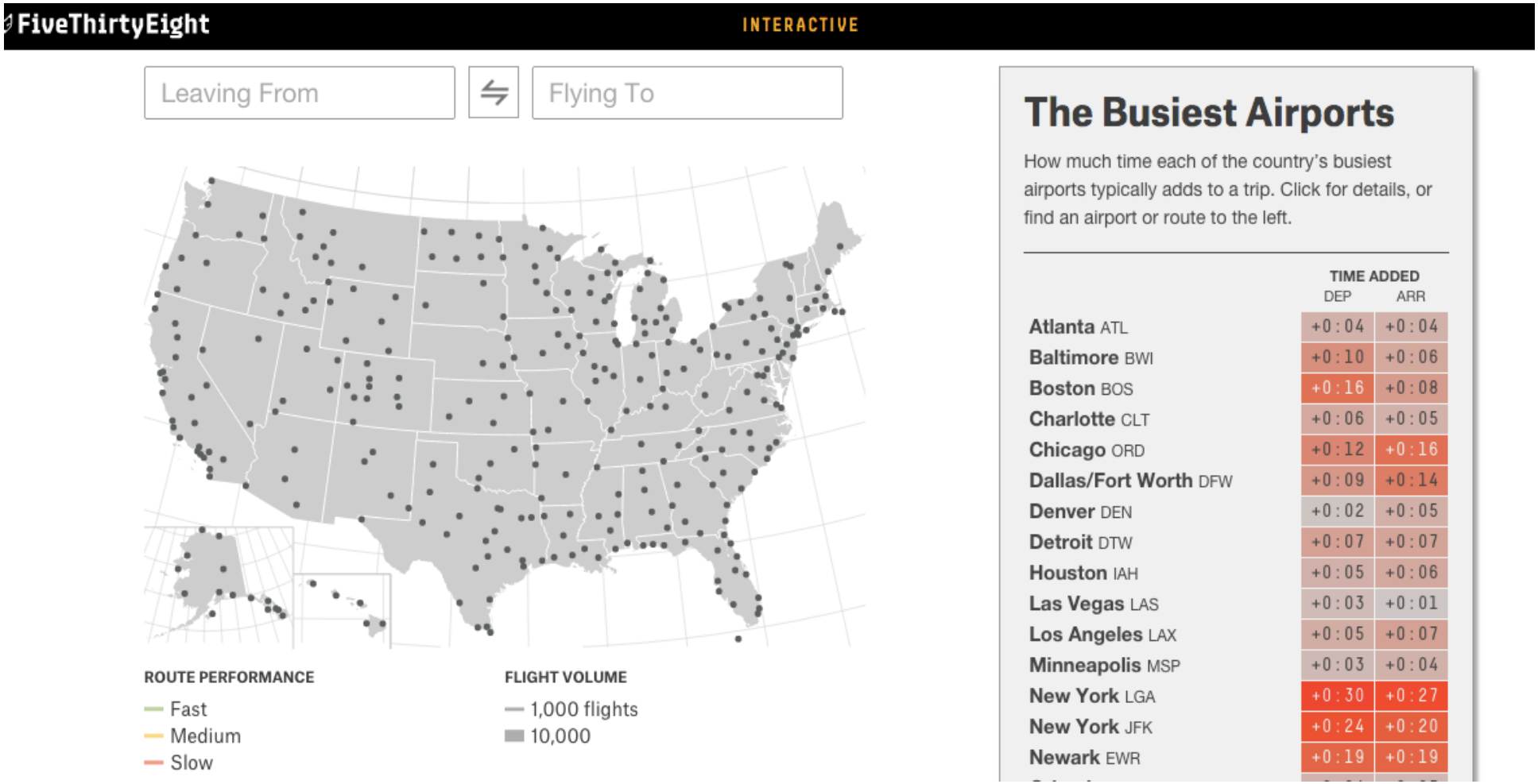
Snapshot of Dataset



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Year	Month	DayofMor	DayOfWe	DepTime	CRSDepTi	ArrTime	CRSArrTin	UniqueCa	FlightNun	TailNum	ActualEla	CRSElapse	AirTime	ArrDelay	DepDelay	Origin	Dest	Distance	TaxiIn	TaxiO
2	2008	1	5	6	2243	1415	45	1625	WN	1684	N347SW	62	70	41	500	508	SAN	PHX	304	2	
3	2008	1	5	6	1940	1220	2111	1350	WN	1684	N347SW	91	90	64	441	440	SFO	SAN	447	5	
4	2008	1	7	1	111	1845	308	2045	WN	405	N644SW	117	120	103	383	386	MDW	JAN	666	4	
5	2008	1	7	1	2213	1700	2317	1655	WN	1827	N759GS	124	55	75	382	313	IND	MDW	162	10	
6	2008	1	7	1	2143	1720	26	1820	WN	1430	N644SW	163	60	83	366	263	STL	MDW	251	24	
7	2008	1	7	1	117	2020	302	2135	WN	490	N651SW	105	75	87	327	297	STL	TUL	351	5	
8	2008	1	7	1	2358	1855	105	2000	WN	490	N651SW	67	65	50	305	303	MDW	STL	251	4	
9	2008	1	3	4	2245	1730	2354	1850	WN	186	N792SW	69	80	59	304	315	JAN	HOU	359	3	
10	2008	1	7	1	2219	1730	35	1935	WN	2474	N710SW	76	65	67	300	289	MDW	CMH	284	2	
11	2008	1	5	6	2129	1620	2246	1750	WN	1924	N408WN	77	90	56	296	309	SFO	LAS	414	4	
12	2008	1	3	4	1615	1130	1623	1135	WN	10	N617SW	68	65	56	288	285	MAF	ABQ	332	4	
13	2008	1	3	4	1736	1305	2031	1555	WN	1837	N761RR	295	290	268	276	271	MDW	SFO	1855	4	
14	2008	1	5	6	2236	1805	2400	1930	WN	646	N283WN	84	85	71	270	271	LAX	SFO	337	6	
15	2008	1	3	4	2021	1700	2303	1835	WN	2005	N302SW	162	95	73	268	201	LAS	SFO	414	4	
16	2008	1	3	4	2059	1620	2216	1750	WN	1924	N761RR	77	90	60	266	279	SFO	LAS	414	6	
17	2008	1	7	1	2348	2105	307	2250	WN	3137	N358SW	259	165	244	257	163	MCO	MDW	989	1	
18	2008	1	3	4	2255	1820	509	55	WN	1924	N761RR	194	215	176	254	275	LAS	IND	1591	9	
19	2008	1	9	3	1458	1040	1725	1315	WN	2556	N501SW	87	95	76	250	258	BNA	BWI	588	4	
20	2008	1	7	1	2300	1835	113	2105	WN	2804	N420WN	253	270	240	248	265	MDW	PDX	1751	5	
21	2008	1	5	6	47	2040	151	2145	WN	505	N435WN	64	65	51	246	247	BWI	PVD	328	5	
22	2008	1	5	6	1558	1225	14	2010	WN	505	N442WN	316	285	250	244	213	SAN	BWI	2295	5	
23	2008	1	5	6	1931	1540	2104	1705	WN	1179	N718SW	93	85	77	239	231	SAN	OAK	446	7	
24	2008	1	4	5	1822	1425	2003	1605	WN	753	N726SW	101	100	88	238	237	PDX	OAK	543	6	

FiveThirtyEight

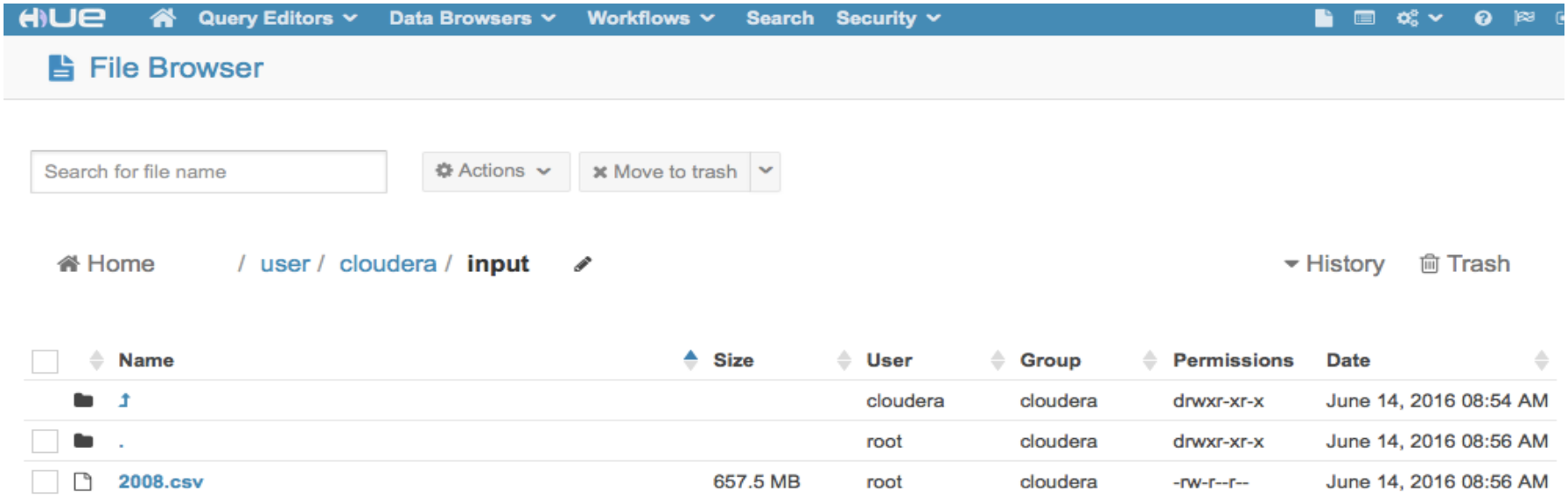
http://projects.fivethirtyeight.com/flights/



Upload Flight Delay Data

Upload a data to HDFS

```
$ wget https://s3.amazonaws.com/imcbucket/data/flights/2008.csv
$ hadoop fs -mkdir /user/cloudera/input
$ hadoop fs -put 2008.csv /user/cloudera/input
```

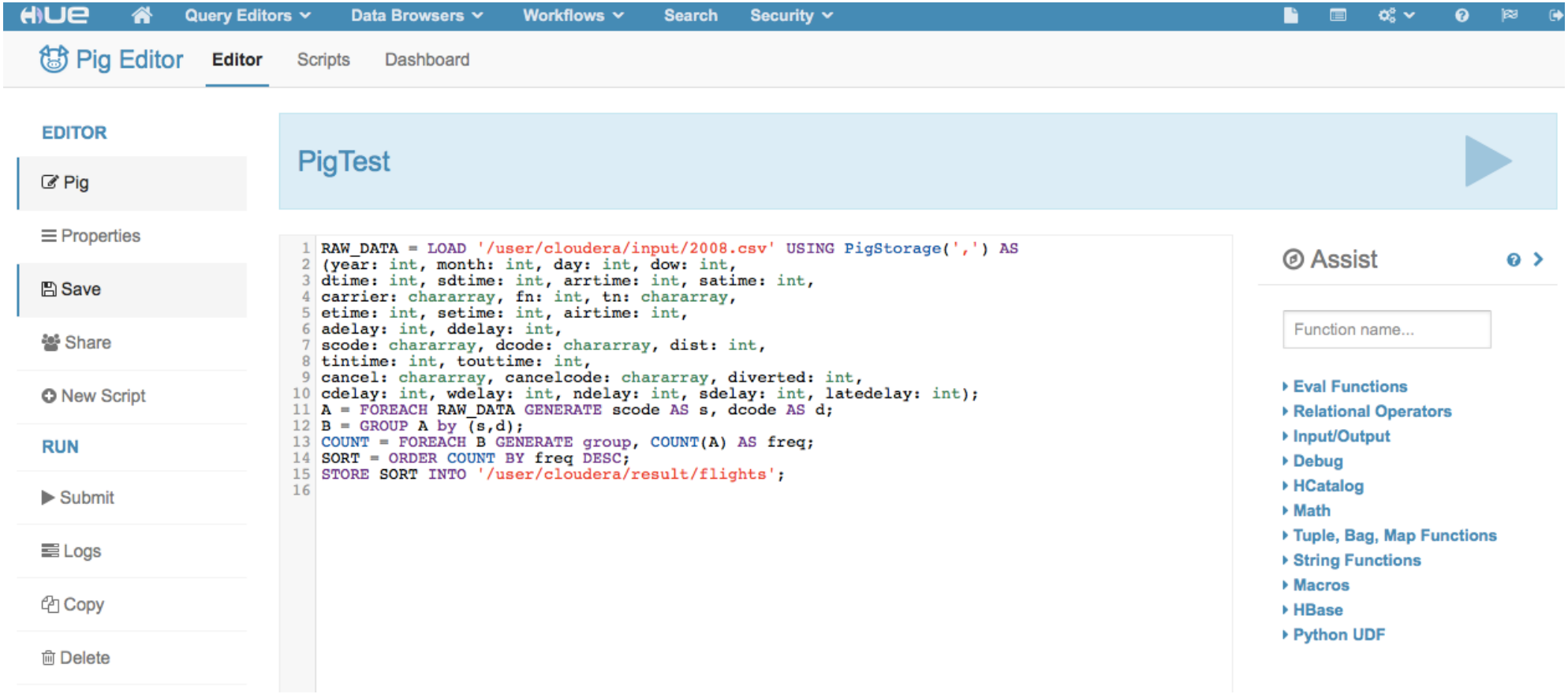


Writing a Pig Script to find 10 most frequent routes

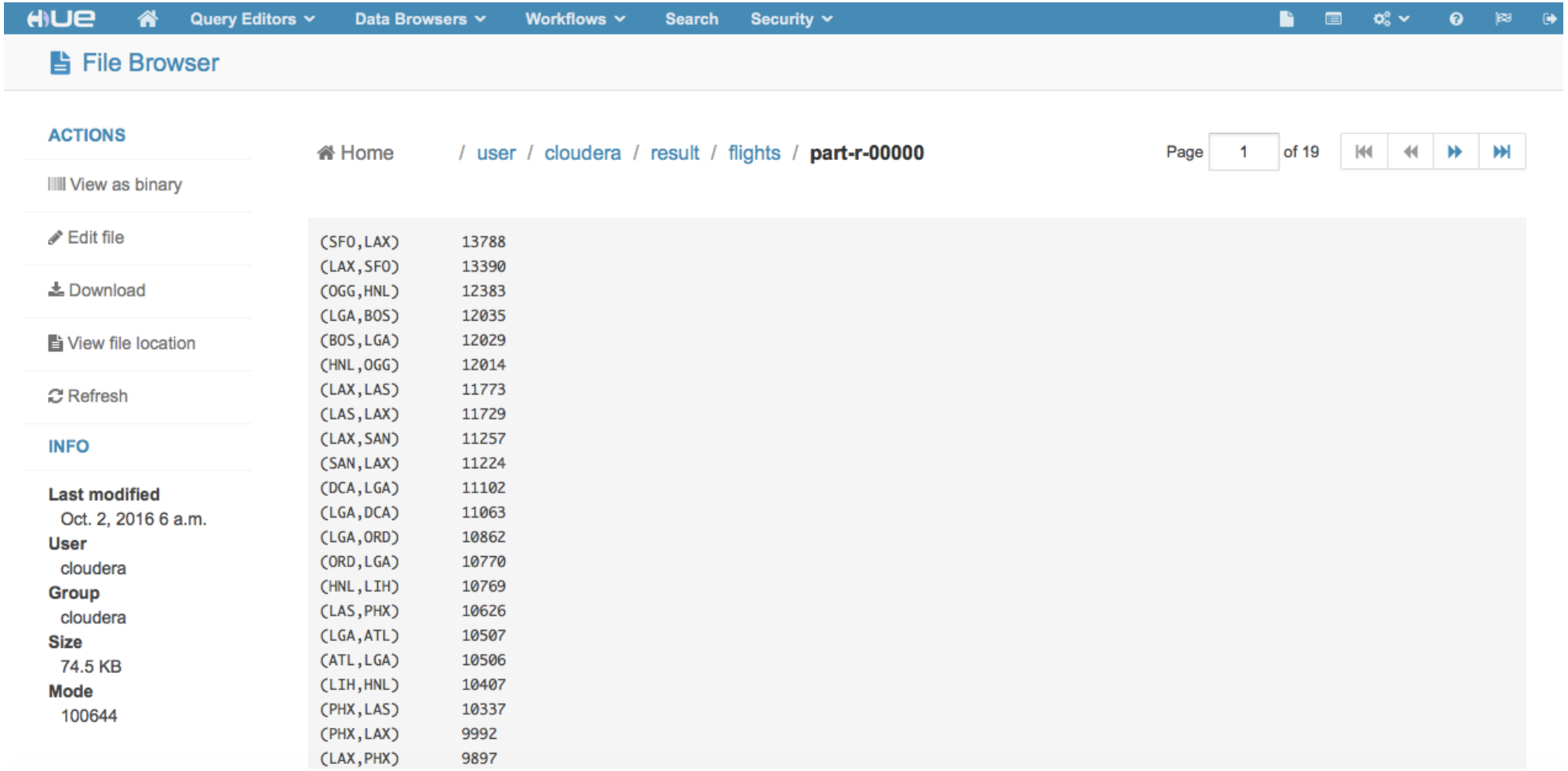
```
RAW_DATA = LOAD '/user/cloudera/input/2008.csv' USING
PigStorage(',') AS
(year: int, month: int, day: int, dow: int,
dtime: int, sdtime: int, arrtime: int, satime: int,
carrier: chararray, fn: int, tn: chararray,
etime: int, setime: int, airtime: int,
adelay: int, ddelay: int,
scode: chararray, dcode: chararray, dist: int,
tintime: int, touttime: int,
cancel: chararray, cancelcode: chararray, diverted: int,
cdelay: int, wdelay: int, ndelay: int, sdelay: int,
latedelay: int);
```

Writing a Pig Script to find 10 most frequent routes

```
A = FOREACH RAW_DATA GENERATE scode AS s, dcode AS d;
B = GROUP A by (s,d);
COUNT = FOREACH B GENERATE group, COUNT(A) AS freq;
SORT = ORDER COUNT BY freq DESC;
STORE SORT INTO '/user/cloudera/result/flights';
```


Query Editors
Data Browsers
Workflows
Search
Security
Pig Editor
Editor
Scripts
Dashboard
EDITOR
Pig
Properties
Save
Share
New Script
RUN
Submit
Logs
Copy
Delete
PigTest
1 RAW_DATA = LOAD '/user/cloudera/input/2008.csv' USING PigStorage(',') AS
2 (year: int, month: int, day: int, dow: int,
3 dtime: int, sdtime: int, arrtime: int, satime: int,
4 carrier: chararray, fn: int, tn: chararray,
5 etime: int, setime: int, airtime: int,
6 adelay: int, ddelay: int,
7 scode: chararray, dcode: chararray, dist: int,
8 tintime: int, touttime: int,
9 cancel: chararray, cancelcode: chararray, diverted: int,
10 cdelay: int, wdelay: int, ndelay: int, sdelay: int, latedelay: int);
11 A = FOREACH RAW_DATA GENERATE scode AS s, dcode AS d;
12 B = GROUP A by (s,d);
13 COUNT = FOREACH B GENERATE group, COUNT(A) AS freq;
14 SORT = ORDER COUNT BY freq DESC;
15 STORE SORT INTO '/user/cloudera/result/flights';
16
Assist
Function name...
Eval Functions
Relational Operators
Input/Output
Debug
HCatalog
Math
Tuple, Bag, Map Functions
String Functions
Macros
HBase
Python UDF

View a result



Impala

Why do we need it?

SPEED

Current Adhoc-Query using Pig

1min, 30sec

- Do count book in each category on “books” (15 M. Records)
  - RAW_DATA = LOAD '/data/books/books' USING PigStorage('|') AS (id: int, isbn: chararray, category: chararray, publish_date :chararray, publisher: chararray, price: float);
  - A = FOREACH RAW_DATA GENERATE category;
  - B = GROUP A by (category);
  - COUNT = FOREACH B GENERATE group, COUNT(A) AS Freq;
  - SORT = ORDER COUNT By Freq DESC;
  - dump SORT

51 Categories – Max is the “REFERENCE”, 314194.

Using Impala

- SELECT category, count(*) cnt
  FROM books
  GROUP BY category
  ORDER BY cnt DESC;

1.75s

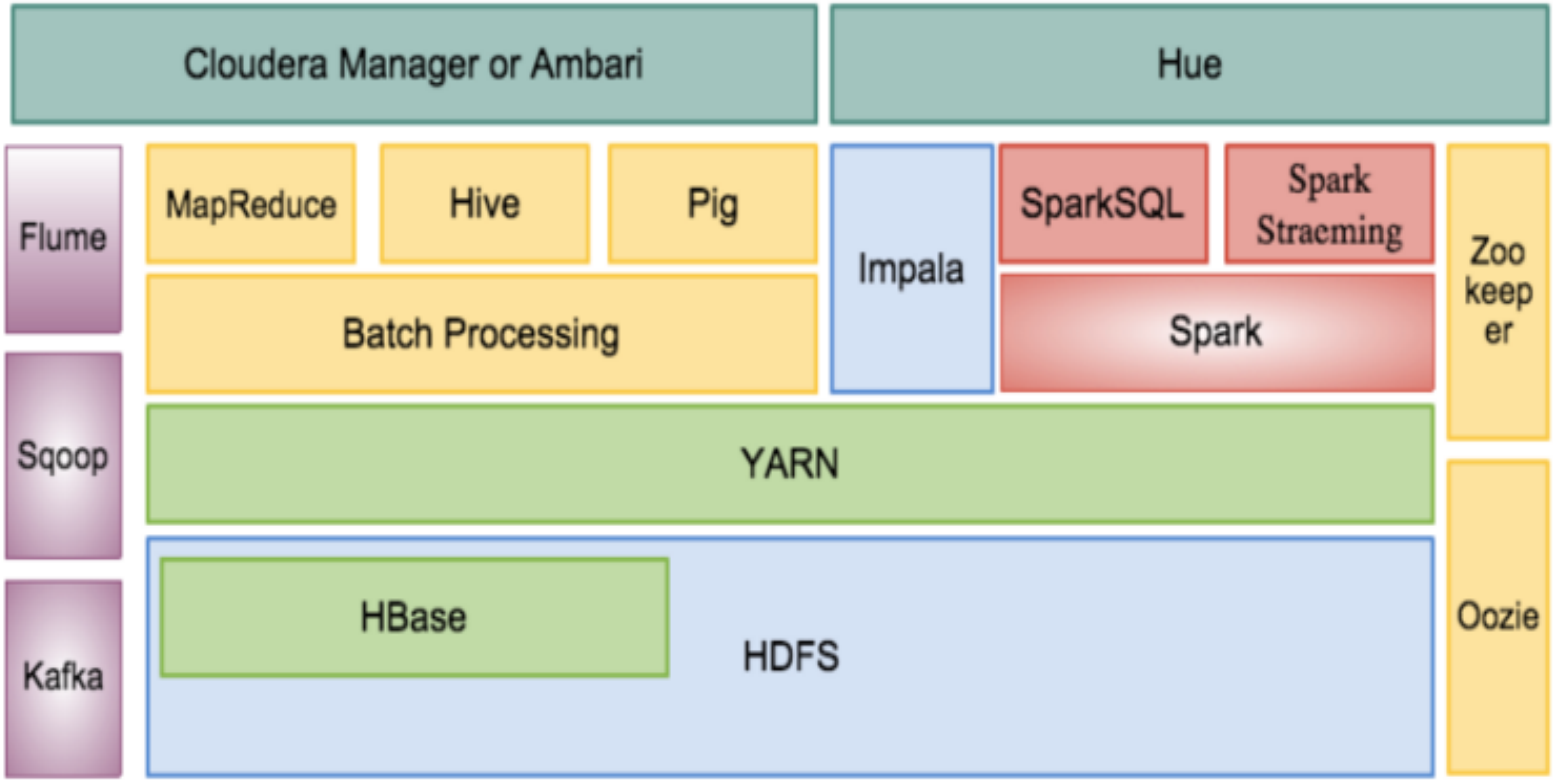
category	cnt
REFERENCE	314194
JUVENILE-NONFICTION	314180
SELF-HELP	314039
PHILOSOPHY	314021
TRANSPORTATION	313973
COOKING	313940
PETS	313917
TRUE-CRIME	313902
JUVENILE-FICTION	313889
SCIENCE	313871
SOCIAL-SCIENCE	313773
FAMILY-RELATIONSHIPS	313617
ARCHITECTURE	313572
LAW	313547
COMPUTERS	313543
COMICS-GRAPHIC-NOVELS	313502
HEALTH-FITNESS	313468
BIBLES	313463
POETRY	313350
LANGUAGE-ARTS-DISCIPLINES	313277
SPORTS-RECREATION	313144
ANTIQUES-COLLECTIBLES	313130
BODY-MIND-SPIRIT	313129
ART	313075
MATHEMATICS	313034
BUSINESS-ECONOMICS	312999
HISTORY	312998
TECHNOLOGY-ENGINEERING	312955
LITERARY-COLLECTIONS	312905
GARDENING	312894
PHOTOGRAPHY	312883
HUMOR	312853
PSYCHOLOGY	312846
GAMES	312831
STUDY-AIDS	312830
DESIGN	312815
PERFORMING-ARTS	312812
BIOGRAPHY-AUTOBIOGRAPHY	312793
HOUSE-HOME	312787
MEDICAL	312721
EDUCATION	312662
DRAMA	312654
FOREIGN-LANGUAGE-STUDY	312605
NATURE	312565
LITERARY-CRITICISM	312523
MUSIC	312459
RELIGION	312369
POLITICAL-SCIENCE	312361
FICTION	312265
TRAVEL	312201
CRAFTS-HOBBIES	311926

Fetched 51 row(s) in 1.75s

About Impala

- Developed by Cloudera
- Open source under Apache License
- Current version is 2.6. (Available in July 2016)
- Connect via ODBC/JDBC/hue/impala-shell

Hadoop Ecosystem

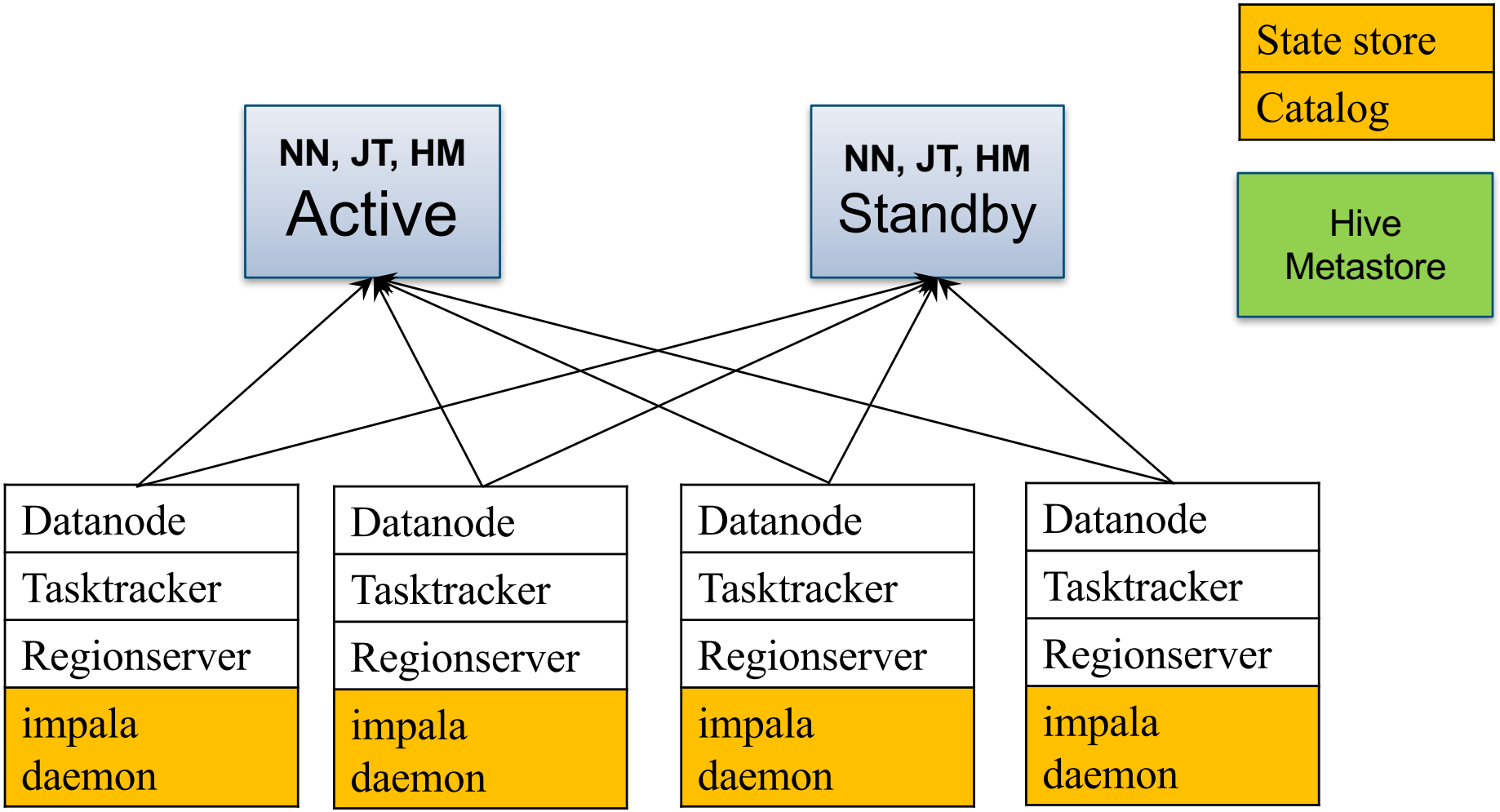


Support queries takes from milliseconds to hours (near real-time)

Benefits

- High Performance
  - C++
  - Direct access to data (No JVM, No MapReduce).
  - In-memory query execution
- Flexibility
  - Query across existing data (No Duplication)
  - Support multiple Hadoop file format
- Scalable
  - Scale out by adding nodes

Impala Architecture



Components

- Impala daemon
  - Runs on every node.
  - Collocate with data nodes.
  - Handle client requests related to query execution.
    - User can submit request to impala daemon running on any node and that node serve as coordinator node
  - Handle query planning & execution.

Components (Cont.)

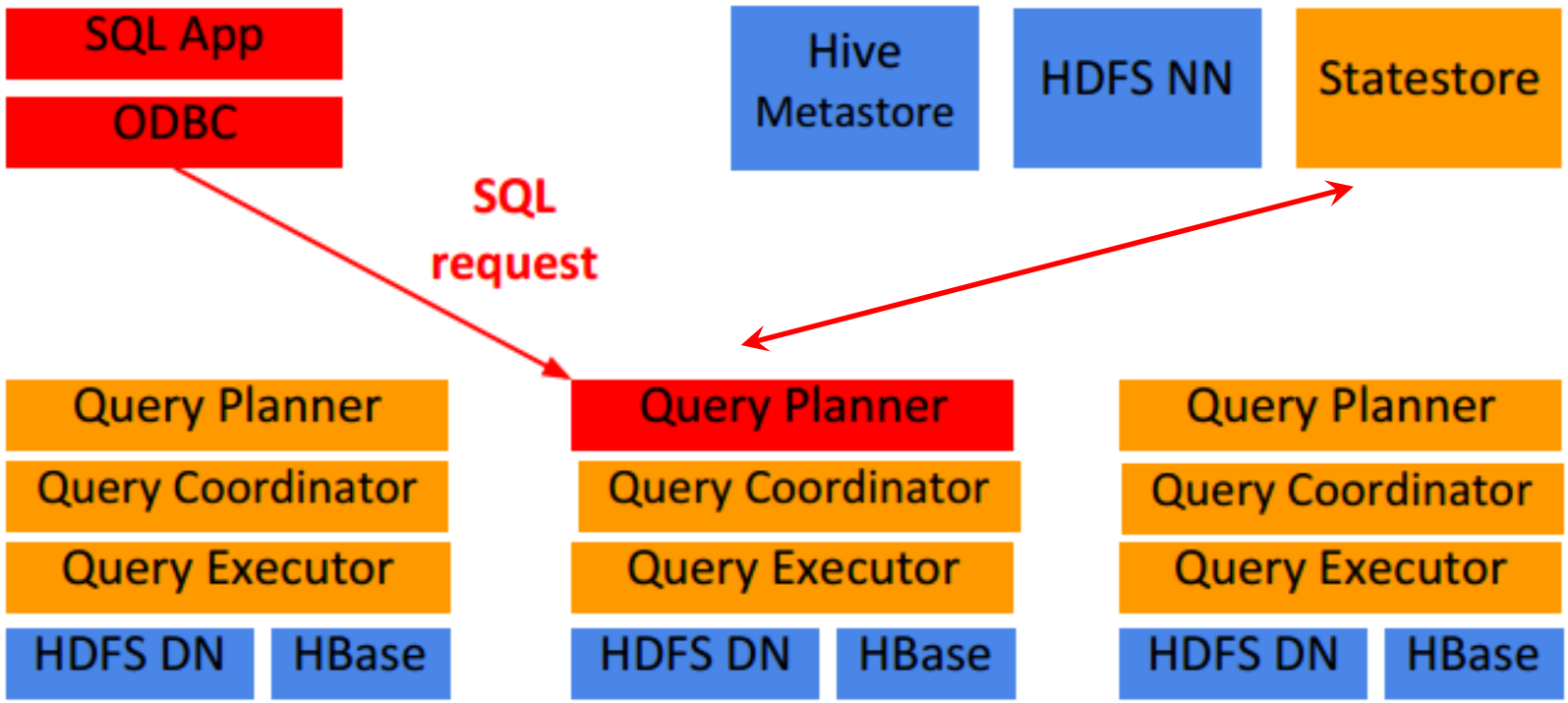
- State store daemon
  - Provides name service
  - Metadata distribution
    - Used for finding data.
  - Communicates with impala daemons to confirm which node is healthy and can accept new work
- Catalog daemon
  - broadcast metadata changes from impala SQL statements to all the impala daemons
    - via the statestored.

Fault tolerance

- No fault tolerance for impala daemons
  - A node failed, the query failed
- State-store offline
  - query execution still function normally
  - can not update metadata(create, alter…)
  - if another impala daemon goes down, then entire cluster can not execute any query
- Catalog offline
  - can not update metadata

Impala Architecture: Query Execution

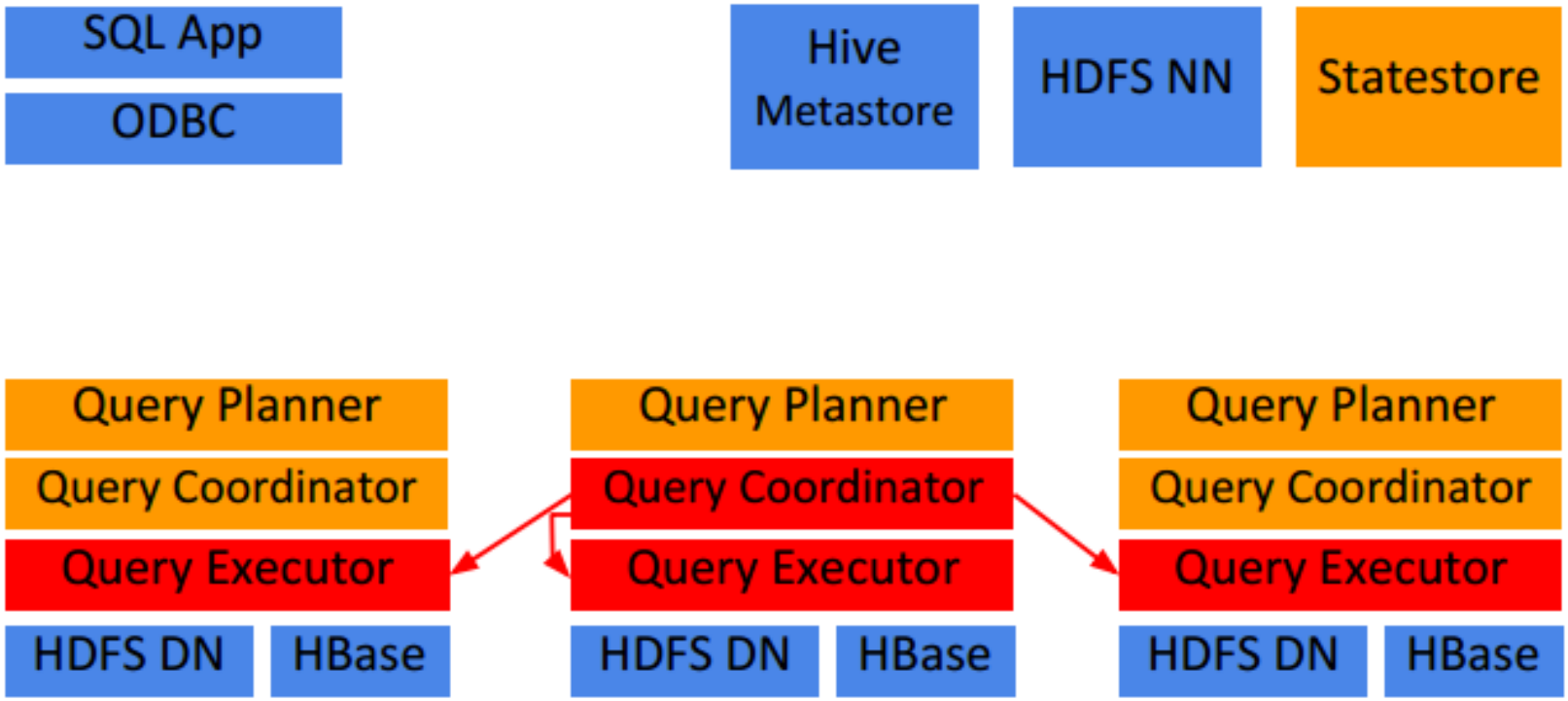
Request arrives via odbc/jdbc



Impala Architecture: Query Execution

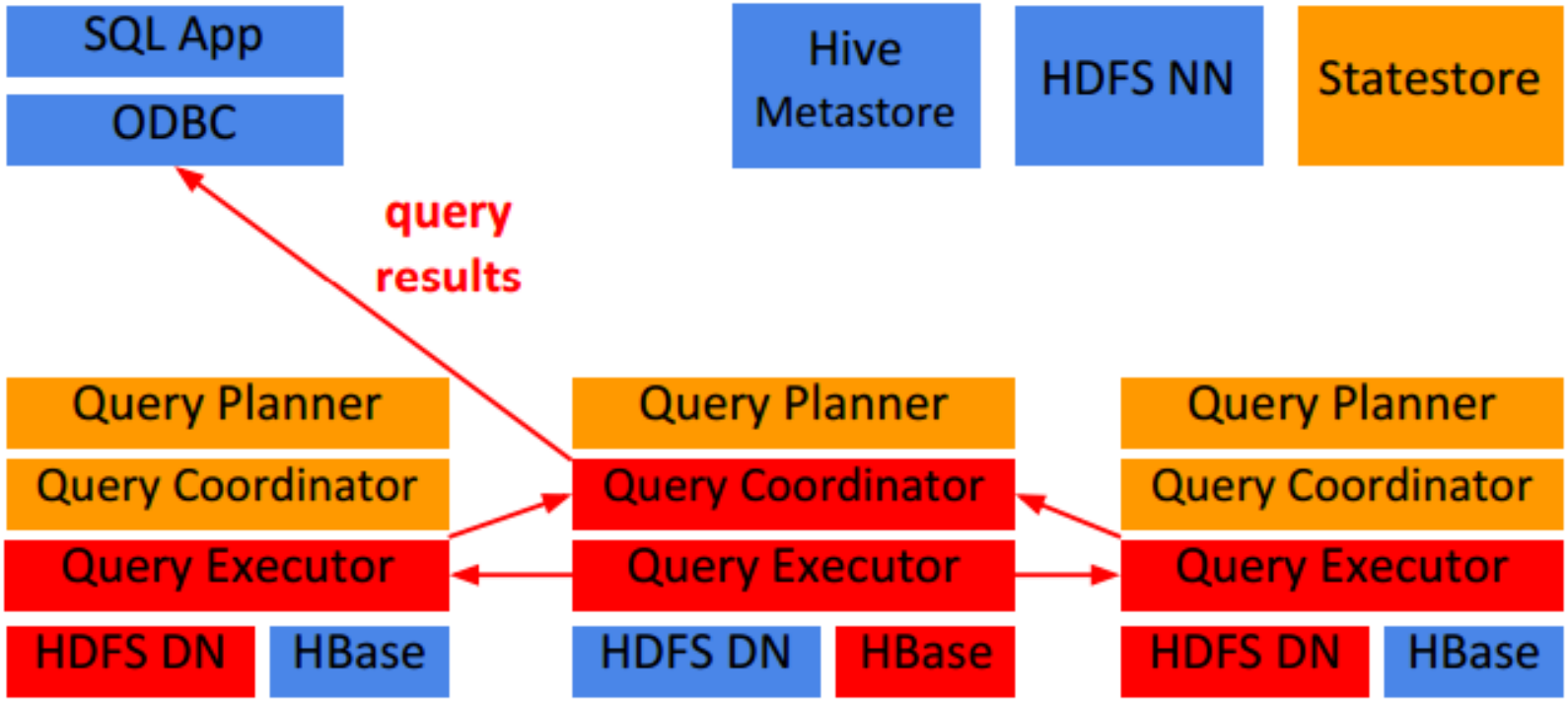
Planner turns request into collections of plan fragments

Coordinator initiates execution on remote impalad's



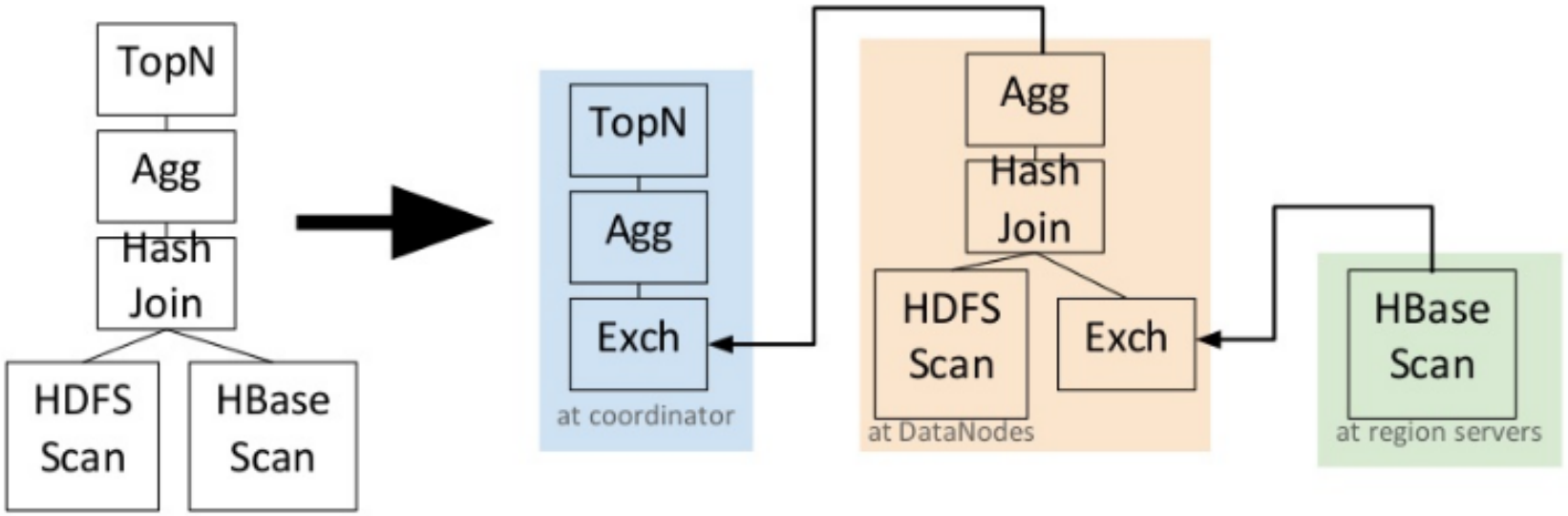
Impala Architecture: Query Execution

Intermediate results are streamed between impalad's Query results are streamed back to client



Impala Architecture: Planner

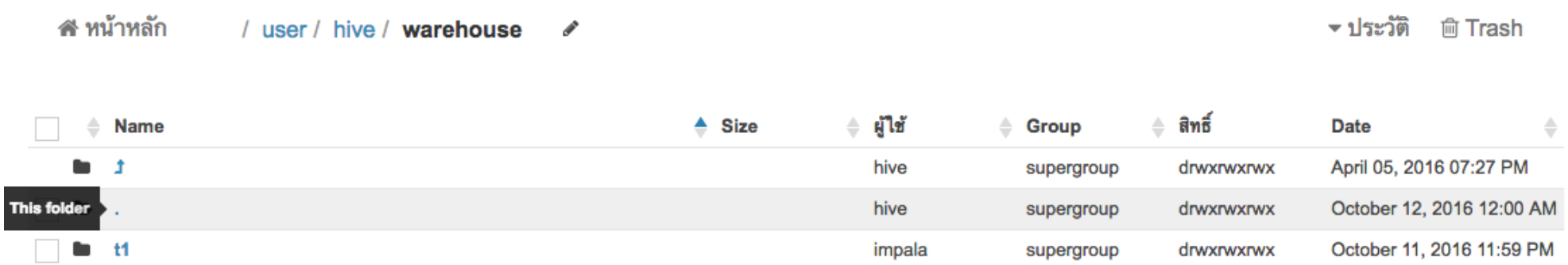
- Example: query with join and aggregation
  SELECT state, SUM(revenue)
  FROM HdfsTbl h JOIN HbaseTbl b ON (...)
  GROUP BY 1 ORDER BY 2 desc LIMIT 10



LAB: Create and insert

```
# impala-shell
> create table t1 (col1 string, col2 int);
> insert into t1 values ('foo', 10);
```

- only supports writing to TEXT and PARQUET tables
- every insert creates 1 tiny hdfs file
- by default, the file will be stored under /user/hive/warehouse/t1/



More information:

http://www.tutorialspoint.com/impala/

http://www.cloudera.com/documentation/enterprise/5-6-x/topics/impala_joins.html