getindata

HCatalog

Motivation

Inside a Data-Driven Company

- **Analysts use multiple tools for processing data**
  - Java MapReduce, Hive and Pig and more
- **Analysts create many (derivative) datasets**
  - Different formats e.g. CSV, JSON, Avro, ORC
  - Files in HDFS
  - Tables in Hive

Pig Analysts

- **To process data, they need to remember**
  - where a dataset is located
  - what format it has
  - what the schema is

```
songs = LOAD 'log' USING PigStorage(',') AS (artistName, songId, timestamp, user);
artists = FOREACH songs GENERATE artistName;
groupped = GROUP artists BY artistName;
counted = FOREACH groupped GENERATE group AS aritstName, COUNT(artists) AS cnt;
STORE counted INTO 'artist-count-pig';
```

Hive Analysts

- **They store popular datasets in (external) Hive tables**
- **To analyze datasets generated by Pig analysts, they need to know**
  - where a dataset is located
  - what format it has
  - what the schema is
  - how to load it into Hive table/partition

Changes, Changes And Changes

Let's start using more efficient format since today!

NO WAY!
We would have to re-write, re-test and re-deploy our applications!
This means a lot of engineering work for us!

MR, Pig, Hive And Data Storage

- **Hive reads data location, format and schema from metadata**
  - Managed by Hive Metastore
- **MapReduce encodes them in the application code**
- **Pig specifies them in the script**
  - Schema can be provided by the Loader

Conclusion

- **MapReduce and Pig are sensitive to metadata changes!**

Data Availability

Jeff: Is your dataset already generated?

Tom: Check in HDFS!

Jeff: What is the location?

Tom: /data/streams/20140101

Jeff: hdfs dfs -ls /data/streams/20140101

Not yet.... :(

Tom: Check it later!

HCatalog

- **Aims to solve these problems!**
- **First of all**
  - It knows the location, format and schema of our datasets!

HCatalog With Pig

- **Traditional way**

```
raw = load ‘/data/streams/20140101’ using MyLoader()
      as (time:chararray, host:chararray, userId:int, songId:int, duration:int);
…
store output into ‘/data/users/20140101’ using MyStorer();
```

HCatalog With Pig

- **Traditional way**

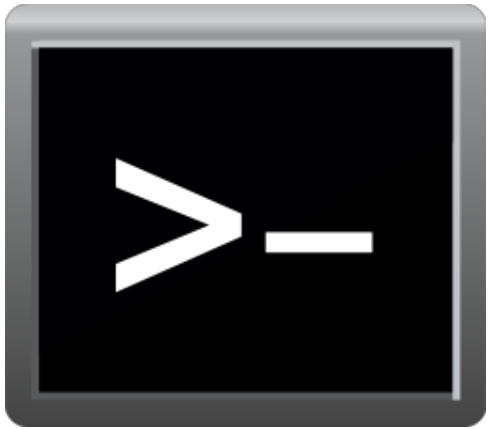
```
raw = load '/data/streams/20140101' using MyLoader()
      as (time:chararray, host:chararray, userId:int, songId:int, duration:int);
…
store output into '/data/users/20140101' using MyStorer();
```

- **HCatalog way**

```
raw = load 'streams' using HCatLoader();
…
store output into users using HCatStorer('date=20140101');
```

Demo

Interacting with HCatalog



Demo

1. Upload a dataset in HDFS
2. Add the dataset HCatalog
3. Access the dataset with Hive and Pig

HCatalog CLI

1. **Upload a dataset in HDFS**

```
$ hdfs dfs -put streams /data
```

Streams: timestamp, host, userId, songId, duration

```
2013-01-04 06:55:23     wa.stream.rock.net      224     4071    30
2013-01-07 06:45:32     ny.stream.rock.net      680     83      172
2013-01-08 18:22:22     phi.stream.rock.net     680     5534    72
2013-01-10 14:46:09     ny.stream.rock.net      30      4042    339
2013-01-10 23:08:34     ny.stream.rock.net      680     6284    306
2013-01-11 08:52:32     wa.stream.rock.net      869     2910    146
```

HCatalog CLI

2. **Add the dataset HCatalog**
   - A file with HCatalog DDL can be prepared

```
CREATE EXTERNAL TABLE streams(time string, host string, userId int, songId int, duration int)
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE
LOCATION '/data/streams';
```

   - And executed by `hcat -f`

```
$ hcat -f streams.hcatalog
```

HCatalog CLI

2. **Add the dataset HCatalog**
   - Describe the dataset

```
$ hcat -e "describe streams"

OK
time                    string                  None
host                    string                  None
userid                  int                     None
songid                  int                     None
duration                int                     None
```

HCatalog CLI

3. Access the dataset with Pig

```
raw_streams = LOAD 'streams' USING org.apache.hcatalog.pig.HCatLoader();
all_count = FOREACH (GROUP raw_streams ALL) GENERATE COUNT(raw_streams);
DUMP all_count;
```

```
$ pig -useHCatalog streams.pig
…
(93294)
```

HCatalog CLI

3. **Access the dataset with Hive**

```
$ hive -e "select count(*) from streams"
OK
93294
```

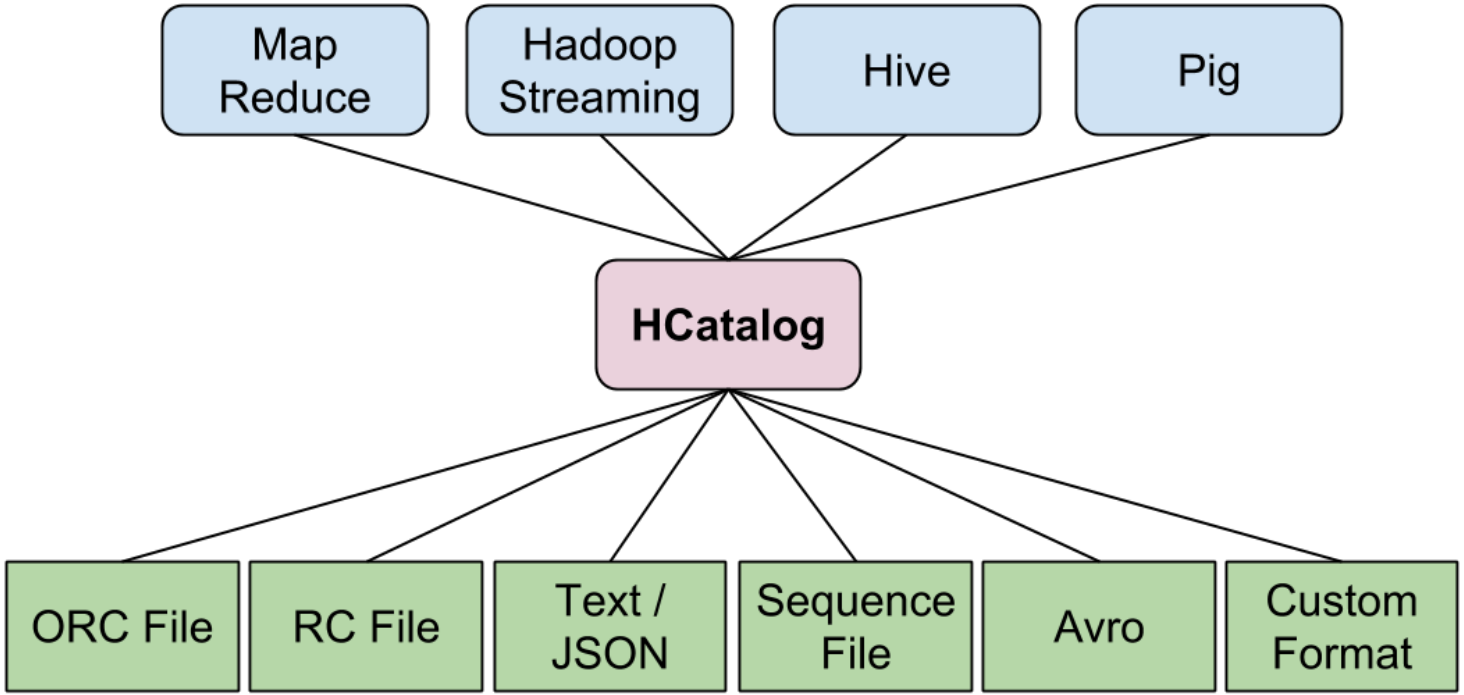
Goals

Three Main Goals

1. **Provide an abstraction on top of datasets stored in HDFS**
   - Just use the name of the dataset, not the path
2. **Enable data discovery**
   - Store all datasets (and their properties) in HCatalog
   - Integrate with Web UI
3. **Provide notifications for data availability**
   - Process new data immediately when it appears

Supported Formats and Projects

Supported Projects And Formats



Custom Formats

- **A custom format can be supported**
  - But `InputFormat`, `OutputFormat`, and Hive `SerDe` must be provided

Pig Interface - HCatLoader

- **Consists of HCatLoader and HCatStorer**
- **HCatLoader read data from a dataset**
  - Indicate which partitions to scan by following the load statement with a partition filter statement

```
raw = load 'streams' using HCatLoader();
valid = filter raw by date = '20140101' and isValid(duration);
```

Pig Interface - HCatStorer

- **HCatStorer writes data to a dataset**
  - A specification of partition keys can be also provided
  - Possible to write to a single partition or multiple partitions

```
store valid into 'streams_valid' using HCatStorer
('date=20110924');
```

MapReduce Interface

- **Consists of `HCatInputFormat` and `HCatOutputFormat`**
- **`HCatInputFormat` accepts a dataset to read data from**
  - Optionally, indicate which partitions to scan
- **`HCatOutputFormat` accepts a dataset to write to**
  - Optionally, indicated with partition to write to
  - Possible to write to a single partition or multiple partitions

Hive Interface

- **There is no Hive-specific interface**
  - Hive can read information from HCatalog directly
- **Actually, HCatalog is now a submodule of Hive**

Conclusion

- **HCatalog enables non-Hive projects to access Hive tables**

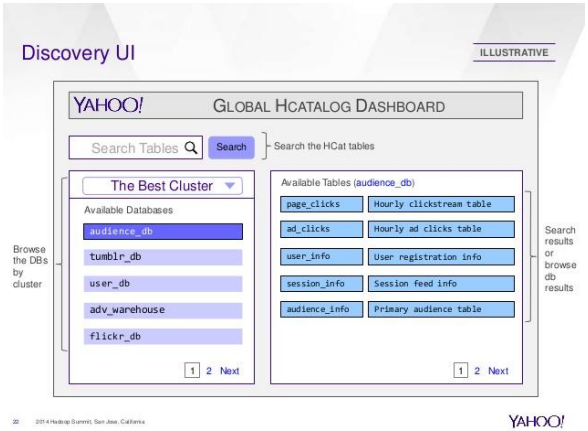
Components

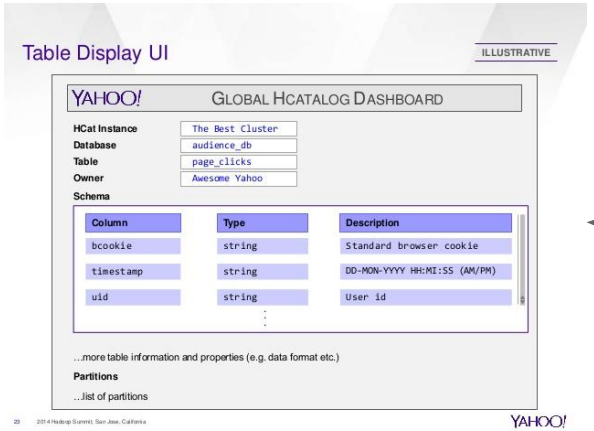
- **Hive Metastore to store information about datasets**
  - A table per dataset is created (the same as in Hive)
- **`hcat` CLI**
  - Create and drop tables, specify table parameters, etc
- **Programming interfaces**
  - For MapReduce and Pig
  - New ones can be implemented e.g. for Crunch
- **WebHCat server**
  - More about it later

Features

Data Discovery

- **A nice web UI can be build on top of HCatalog**
  - You have all Hive tables there for free!
  - See Yahoo!'s *illustrative* example below





Pictures come from Yahoo's presentation at Hadoop Summit San Jose 2014

Properties Of Datasets

- **Can store data life-cycle management information**
  - Cleaning, archiving and replication tools can learn which datasets are eligible for their services

```
ALTER TABLE intermediate.featured-streams SET TBLPROPERTIES ('retention' = 30);

SHOW TBLPROPERTIES intermediate.featured-streams;
SHOW TBLPROPERTIES intermediate.featured-streams("retention");
```

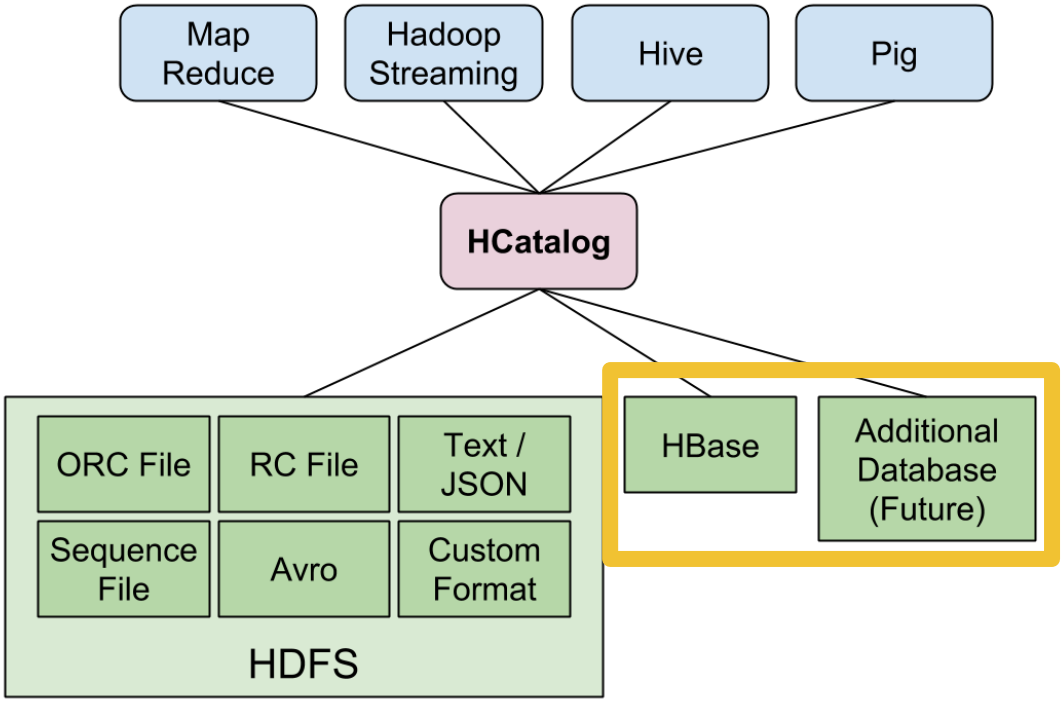
Notifications

- **Provides notifications for certain events**
  - e.g. Oozie or custom Java code can wait for those events and immediately schedule tasks depending on them
- **Multiple events can trigger notifications**
  - A database, a table or a partition is added
  - A set of partitions is added
  - A database, a table or a partition is dropped

Evolution Of Data

- **Allows data producers to change how they write data**
  - No need to re-write existing data
  - HCatalog can read old and new data
  - Data consumers don’t have to change their applications
- **Data location, format and schema can be changed**
  - In case of schema, new fields can be added

HCatalog Beyond HDFS



WebHCat Server

- **Provides a REST-like web API for HCatalog**
  - Send requests to get information about datasets

```
curl -s 'http://hostname:
port/templeton/v1/ddl/database/db_name/table/table_name?user.
name=adam'
```

  - Send requests to run Pig or Hive scripts

- **Previously called *Templeton***

There Is More!

getindata

- **Data engineers, architects and instructors**
- **+4 years of experience in Apache Hadoop**
  - Working with hundreds-node Hadoop clusters
  - Developing Hadoop applications in many cool frameworks
  - Delivering Hadoop trainings for +2 years
- **Passionate about data**
  - Speaking at conferences and meetups
  - Blogging and reviewing books