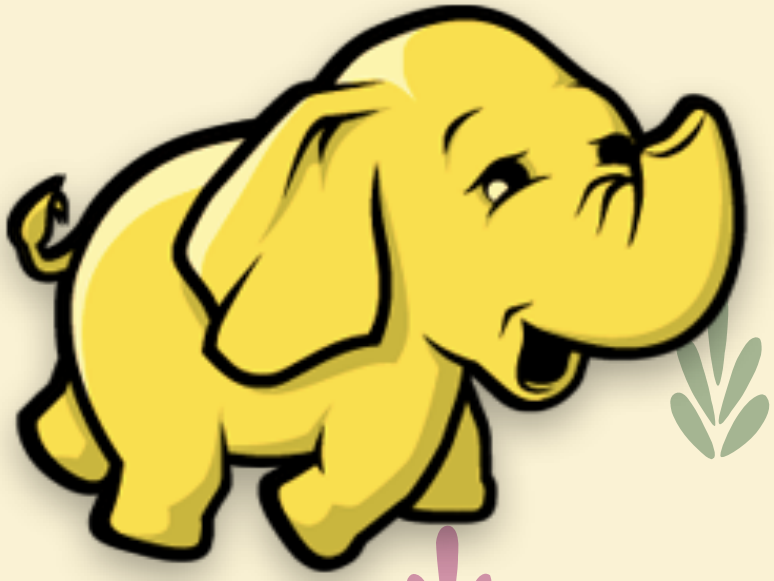
Hadoop for the Complete Beginner

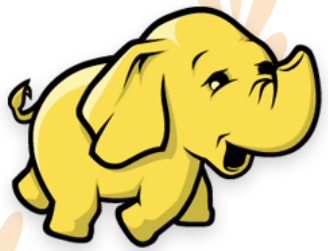


Charity Hilton

Dev U, *24-Mar-2016*

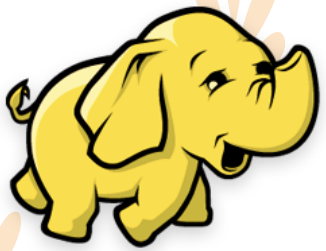
What is Hadoop?
(in 4 points)

1. Open source software (Apache)
2. Reliable
3. Scalable
4. Distributed

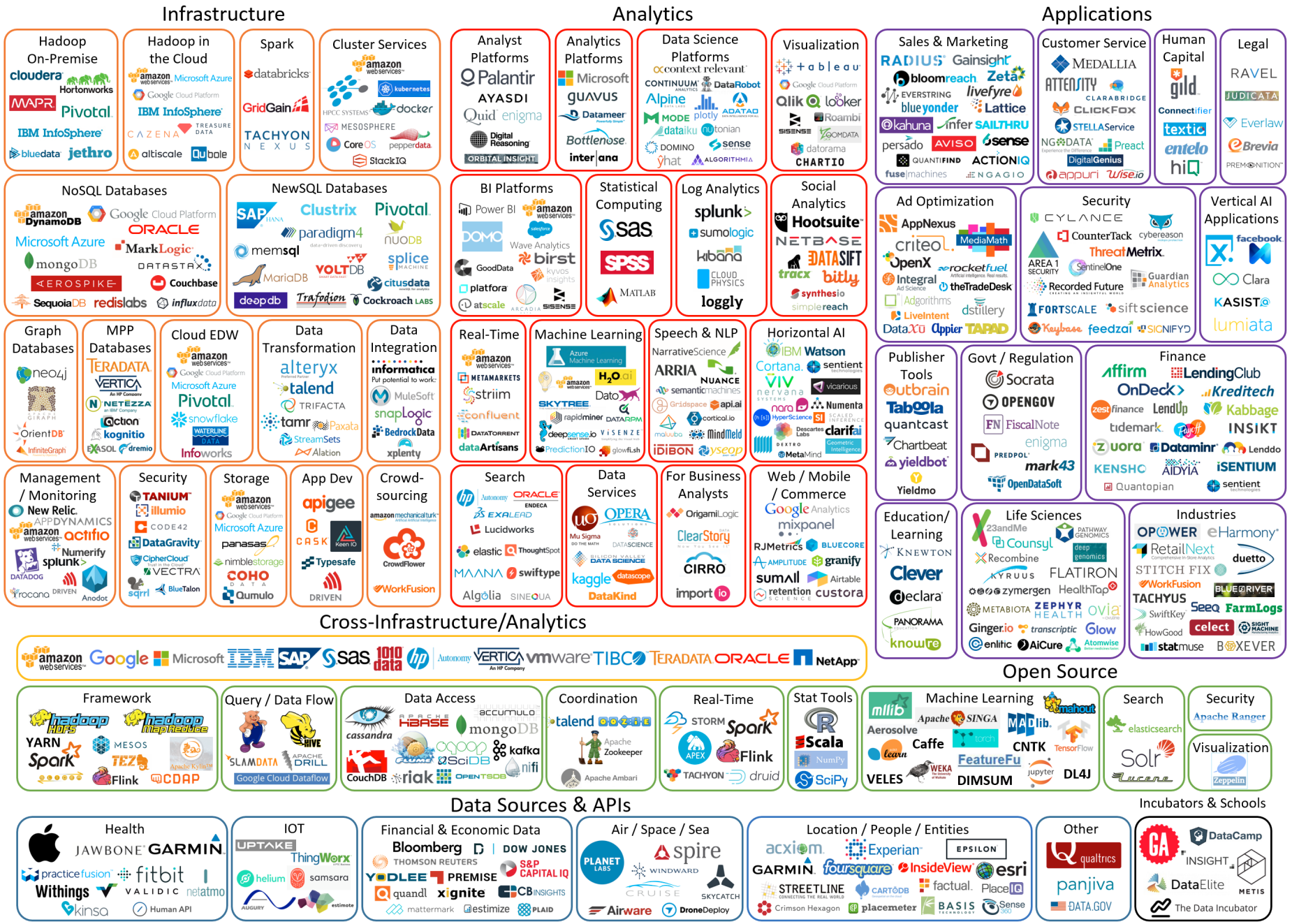


What is Hadoop?

- Enables distributed processing of large data sets across clusters of commodity servers.
- Hadoop is (relatively) low cost, scalable, fault-tolerant, and flexible.
- Built with the assumption that hardware fails
- Allows for keeping all data without strict schema rules
- Written in Java – but supports other languages
- …and the ecosystem is rapidly evolving



Big Data Landscape 2016 (Version 2.0)



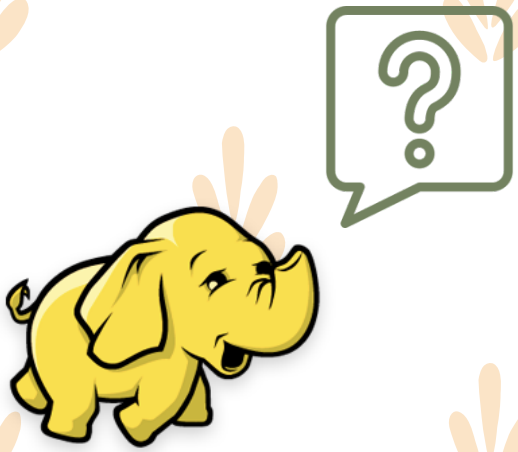
Last Updated 2/12/2016

© Matt Turck (@mattturck), Jim Hao (@jimrhao), & FirstMark Capital (@firstmarkcap)

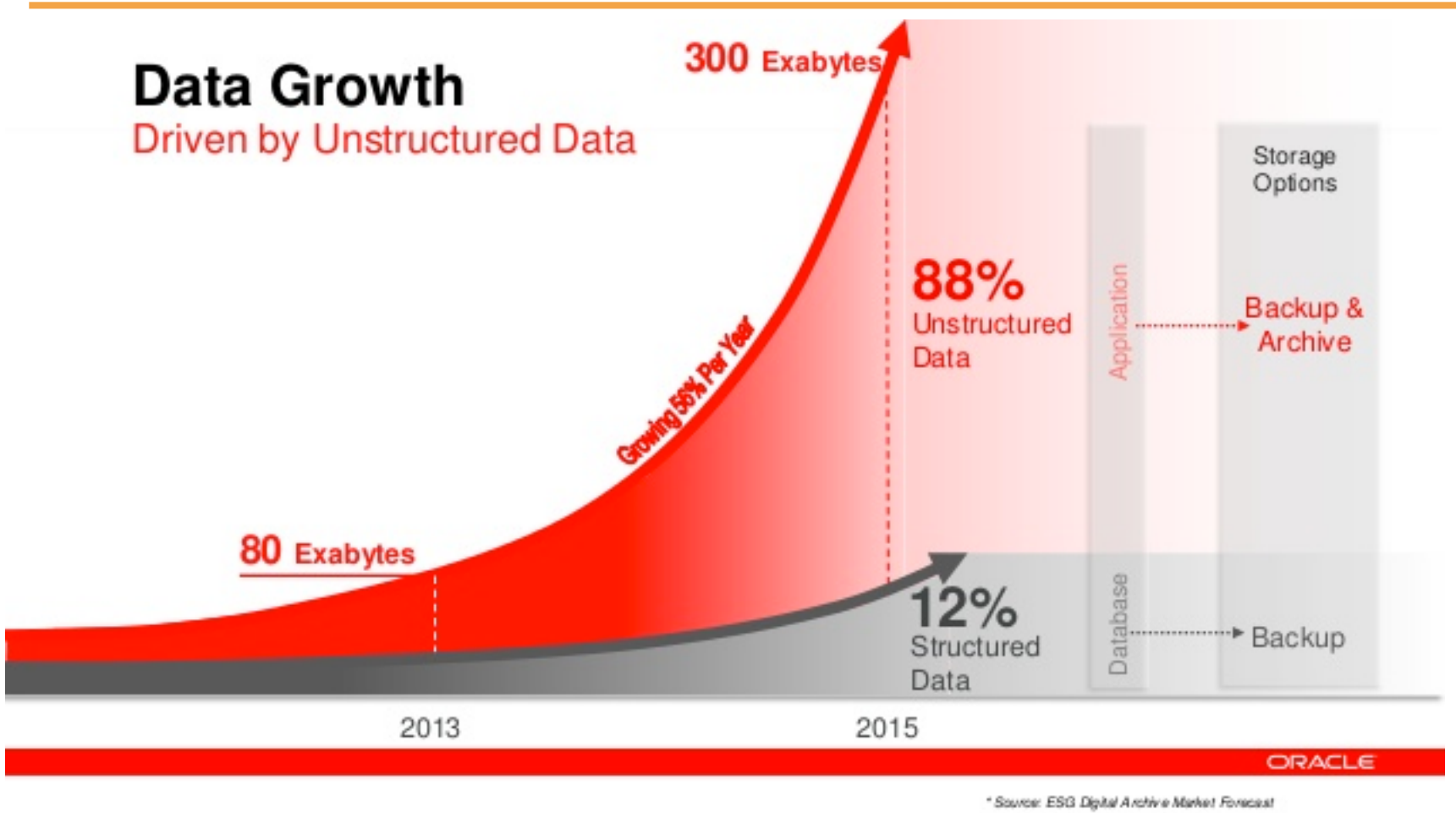
FIRSTMARK

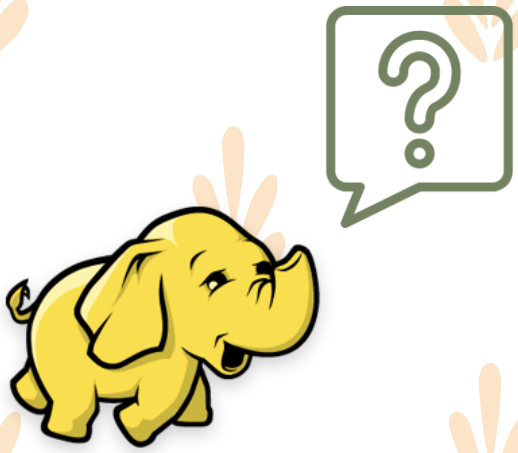
Why Hadoop?

- We have massive amounts of data, growing all the time. In 2013, it was estimated that 90% of the world's data had been created in the previous 2 years, and that number appears to be growing. https://www.sciencedaily.com/releases/2013/05/130522085217.htm
- We have massive amounts that unstructured data, with more growing each year. Unstructured data doesn't traditionally work as well in standard relational databases
- That includes many new data sources, click stream data, personal health data, location data, etc. (but still happily consumes data from relational databases)
- We need to do predictive analytics, to sift through text data, identify patterns of use, etc.
- Big Data!



Why Hadoop? – Data is big and getting bigger and less structured





What happens in one internet minute?

Can you guess?

By the way, in the 63 seconds you've been on this page, approximately 1422162 GB of data was transferred over the internet.

What happens in INPCR in one hour?

(specifically new data added in the 8AM hour on 22-Mar-2016)

3,411
Text
Documents

7,623
Encounter
Accounts

56,942
Clinical
Variables

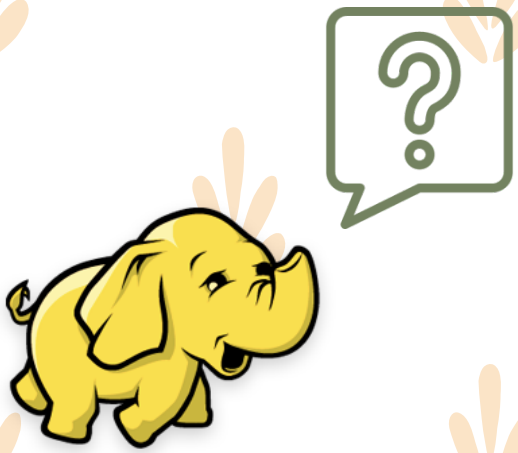
325
Patients
Added

When do we use Hadoop?

And how are other people using Hadoop?

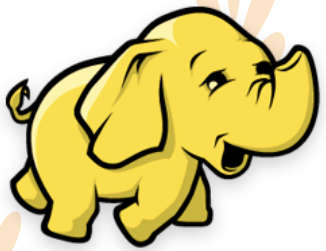
When do we solve our problem with Hadoop?
(in 3 steps)

1. We have complex, diverse data
2. We have large amounts data (if we're concerned about data costs of storing in a relational database)
3. We need advanced analytics (sum, count, average, aren't enough; we need predictive analytics, natural language processing, pattern recognition, machine learning)



What are sample Hadoopable problems?

1. Identifying patterns relating to fraud on the web
2. Analyzing log data
3. Predicting what people buy, like
4. Spam classification
5. Facebook
   - *We use Apache Hadoop to store copies of internal log and dimension data sources and use it as a source for reporting/analytics and machine learning.*
   - *Currently we have 2 major clusters:*
     - *A 1100-machine cluster with 8800 cores and about 12 PB raw storage.*
     - *A 300-machine cluster with 2400 cores and about 3 PB raw storage.*
     - *Each (commodity) node has 8 cores and 12 TB of storage.*
     - *We are heavy users of both streaming as well as the Java APIs. We have built a higher level data warehousing framework using these features called Hive (see the http://hadoop.apache.org/hive/). We have also developed a FUSE implementation over HDFS.*

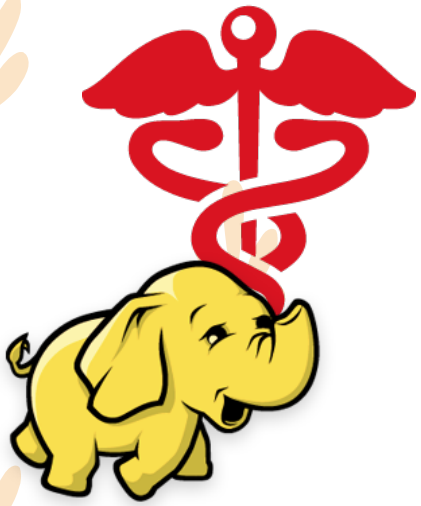


What are sample Hadoopable problems (in Healthcare)?

1. Analyzing genomic data to detect conditions

'Matching a particular drug to a specific genomic profile is a big data problem. Each individual's genome is about 1.5 gigabytes of data. Massive data storage and processing power is required to analyze data on a drug's interactions with different genetic combinations. For example, just focusing on 20 genes is a 20,000-choose-20 calculation, with 4.3 x 10^67 possible combinations.'

http://hortonworks.com/industry/healthcare/



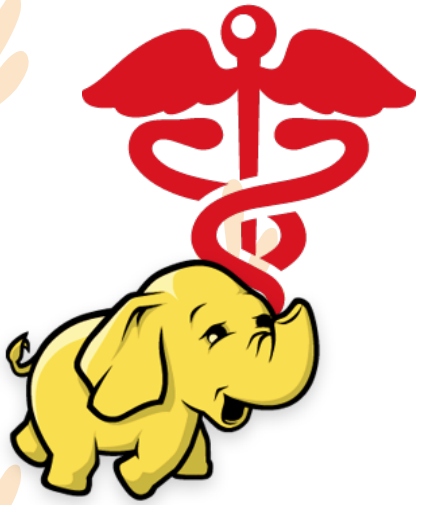
What are sample Hadoopable problems (in Healthcare) ?

2. Unifying complex data sets

Hadoop is uniquely capable of storing a wide range of healthcare environment data not matter the type or amount of structure. For us, this includes:

- all ancillary HL7 feeds (without the need for modification),
- EMR generated data,
- genomic data,
- financial data,
- RTLS data from assets,
- patient and caregiver data,
- smart pump data,
- incremental physiological monitoring measurements (across one minute increments),
- ventilator data in one minute or less increments
- and temperature and humidity data.

http://hortonworks.com/blog/improving-quality-of-care-with-apache-hadoop-at-uc-irvine-medical-center/



Saritor: A Hadoop Ecosystem to Advance Clinical Research and Practice

Charles Boicey, MS, RN-BC[1], Lisa Dahm, PhD[1], David Gonzalez[1], Mahesh Rangarajan[2], Rushipriya Panda[2], Jeff Markham[3]

[1]University of California, Irvine, [2]CMC Americas, [3]Hortonworks

Introduction

Facebook, Twitter, LinkedIn and Yahoo share the same underlying infrastructure, Apache Hadoop. All three of these applications consume, process and store millions of records consisting of structured, unstructured, image and video data. As healthcare data shares many of the characteristics of the data found in Facebook, Twitter, LinkedIn and Yahoo, Hadoop should be an ideal environment for the ingestion, storing and utilization of healthcare data.

Methods

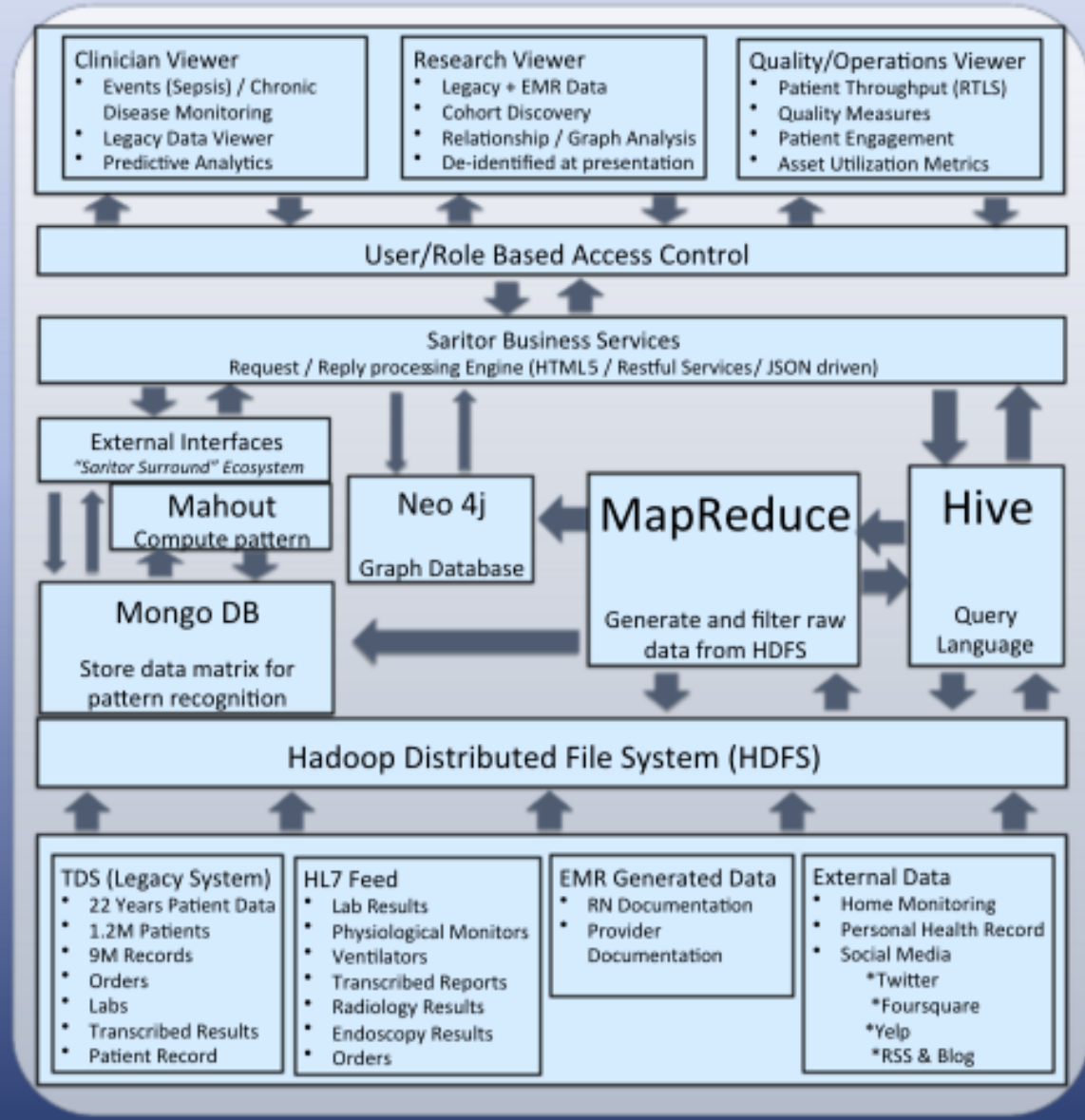
A virtual Apache Hadoop version 1.0 infrastructure consisting of a single NameNode server and four Task Node servers was set up within the UCI Medical Center data center. Ubuntu Linux running on VMware was the chosen OS. The Hadoop modules utilized were: Hadoop Common, Hadoop Distributed File System (HDFS), MapReduce, Pig, Mahout and Zookeeper. Java scripted routines processed the legacy data. Mirth HL7 listener and a java scripted routine processed the HL7 data.

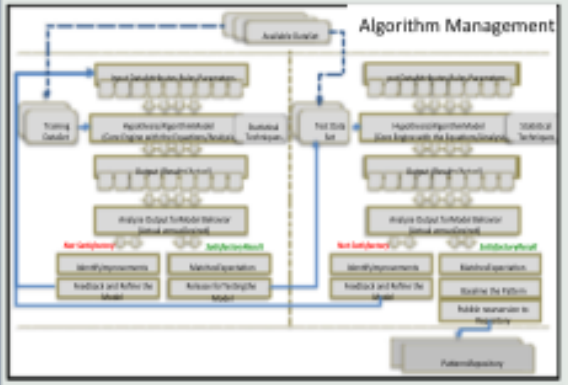
Results

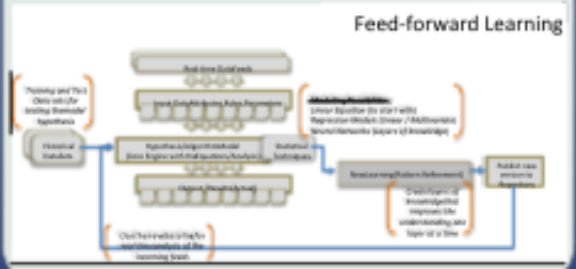
The legacy data of 1.2 million patients, contained in 9 million patient medical records was successfully ingested into the Saritor Hadoop Distributed File System. For researchers the drag and drop query and visualization tool allowed for the visualization of the legacy data. For clinicians in patient care complete patient records were retrieved via a web browser. HL7 messages from all source systems, physiological monitoring data in one-minute intervals, and ventilator data in one-minute intervals and EMR generated data was ingested and stored. Algorithms for sepsis, hospital acquired conditions and 30 day readmits are able to be built into Mahout for real time surveillance.

Discussion

Our initial findings demonstrated the Hadoop ecosystem is well suited for the ingestion, storage and retrieval of both legacy EMR data and runtime EMR data. Minimal programing is required to process legacy data and the processing of runtime EMR data requires the cloning of existing interfaces. The functionality of real time clinical surveillance presents unlimited use cases. Hadoop is an ecosystem that is affordable, scalable, highly available, allows for clinical research and clinical practice to coexist in the same system.





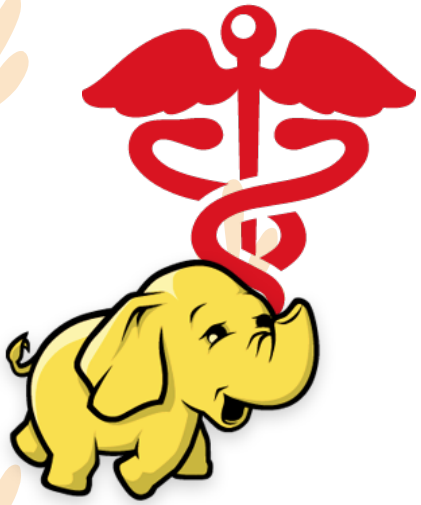


What are sample Hadoopable problems (in Healthcare)?

3. Shorten screening times with machine learning

Dr. Dennis Wall is Director of the Computational Biology Initiative at the Harvard Medical School and a process his team developed for low-cost, mobile screening for autism. It takes less than five minutes and relies on the ability to store large volumes of semi-structured data from brief in-home tests administered and submitted by parents. Wall's lab also used Facebook to capture user-reported information on autism.

https://www.youtube.com/watch?v=01i3vQKB3vw



What are sample Hadoopable problems (in Healthcare) ?

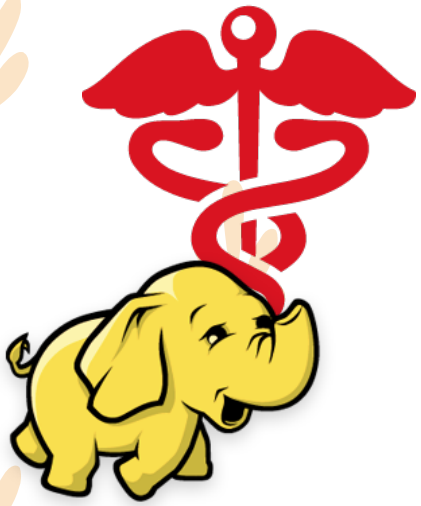
4. Linking with the Internet of Things to treat and cure

[Ido Karvany of Intel's Advanced Analytics group] explained that the research team has built an IoT Big Analytics platform (on Amazon Cloud Drive) based on open source technologies, such as Cloudera Distribution for Hadoop, to enable collection and processing of high data streams (up to 1 GB per patient per day). Mr. Karavney noted that the platform has been successfully used in multiple clinical trials and the project has started ramping up to connect thousands of patients 24/7 by the end of 2015.

The platform uses HBase & HDFS as its main scalable storage layer. The analytics batch layer leverages Apache Spark (over HBase & HDFS) and includes a set of complex machine learning algorithms, sophisticated event-based rule engine, an automatic change detection engine and a variety of PD-related measurements.

Examples for those are activity recognition, patients' sleep quality, tremor detection, PD gait recognition, and others. Mr. Karavney's presentation included an explanation of the way researchers are using Spark for implementing their machine learning algorithms.

http://parkinsonsnewstoday.com/2015/11/16/might-parkinsons-unlocked-via-big-data-internet-things/



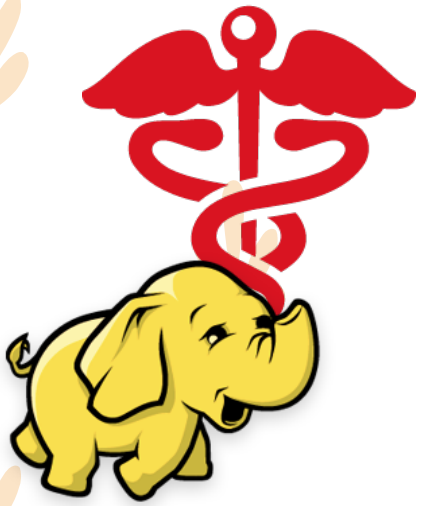
What are sample Hadoopable problems (in Healthcare) ?

5. Increasing participation in clinical trials

According to the National Institutes of Health, over 80% of clinical trials fail due to complications with enrollment timelines and troubles with patient retention. These problems lead to insufficient data sets, stalled drug development, and ultimately billions of lost dollars spent by pharmaceutical companies.

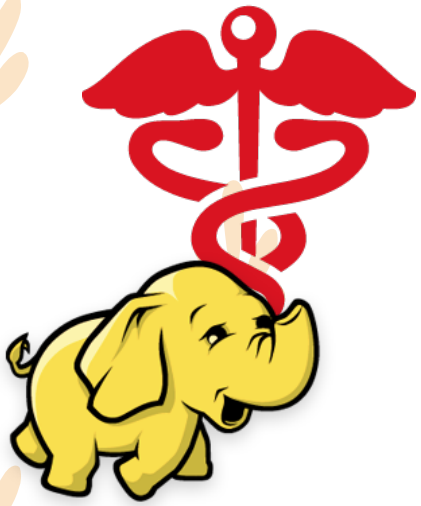
"Traditional recruitment and enrollment into clinical trials is unfocused, laborious and expensive," Lisa Griffin Vincent, president of PatientPoint Outcomes Research Solutions recently told InformationWeek. "Recruitment methods typically include clinic staff poring through paper clinical charts page by page to identify patients that may meet the trial eligibility criteria. Access to big data can transform and accelerate the clinical trials process if leveraged in the right ways."

http://www.datanami.com/2013/10/08/big_data_to_give_clinical_trials_a_big_boos/



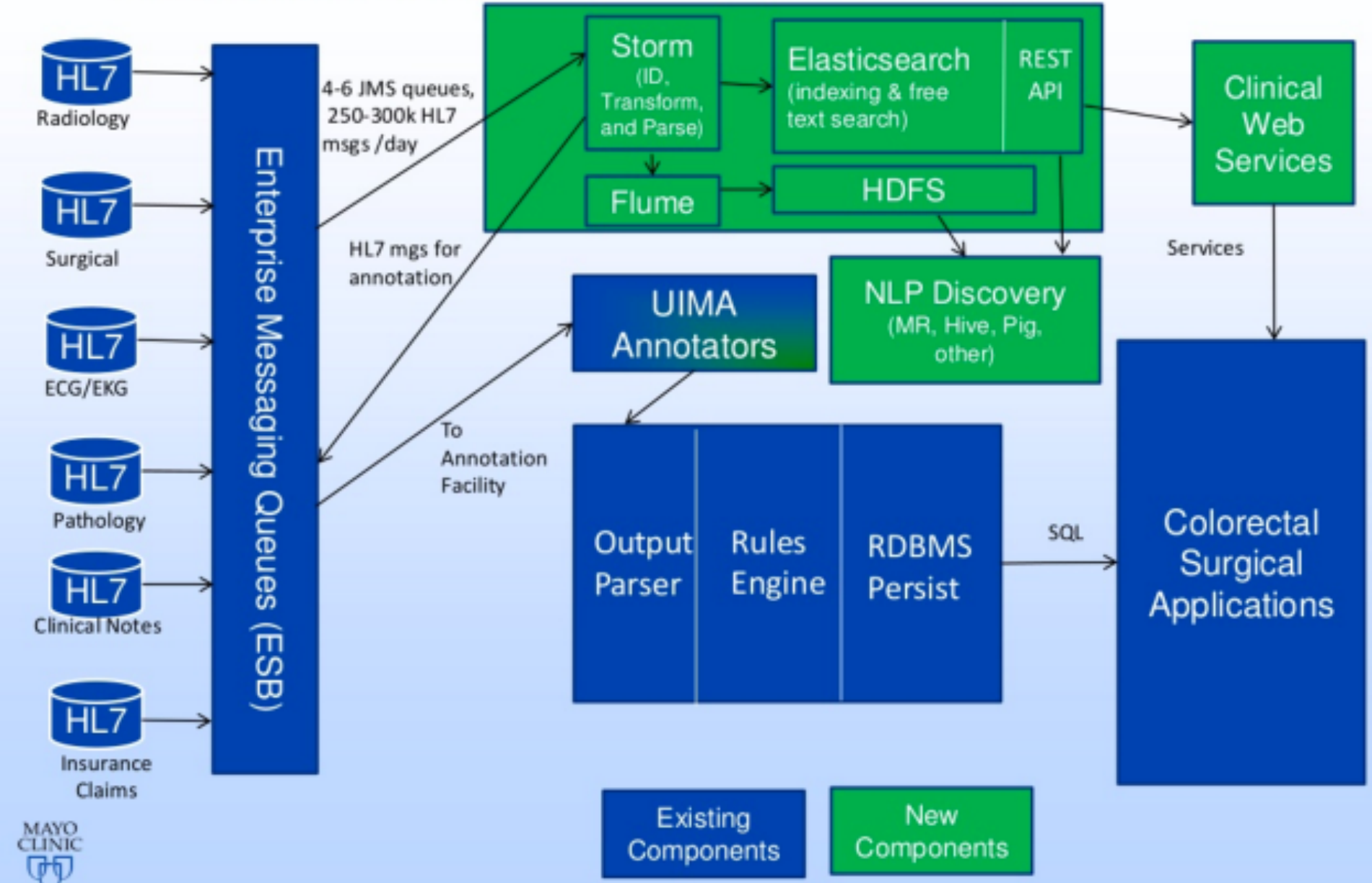
What are sample Hadoopable problems (in Healthcare) ?

6. Real-time super fast text search
   - Mayo clinic moved from processing thousands of HL7 messages to tens of thousands of HL7 messages daily
   - Free text query times moved from minutes to milliseconds
   - Simplified architecture to increase volume and reduce costs
   - http://www.slideshare.net/Hadoop_Summit/big-data-platform-processes-daily-healthcare-data-for-clinic-use-at-mayo-clinic-50571610



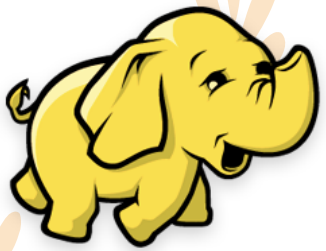
Solution Architecture

In Support of Colorectal Surgery Applications



What are sample Hadoopable problems? (some resources)

- https://www.quora.com/Any-real-life-use-case-of-Apache-Hadoop
- http://www.cloudera.com/content/dam/cloudera/Resources/PDF/whitepaper/10_Common_Hadoopable_Problems.pdf
- http://shop.oreilly.com/product/0636920038450.do
- http://hortonworks.com/industry/healthcare/
- https://wiki.apache.org/hadoop/PoweredBy



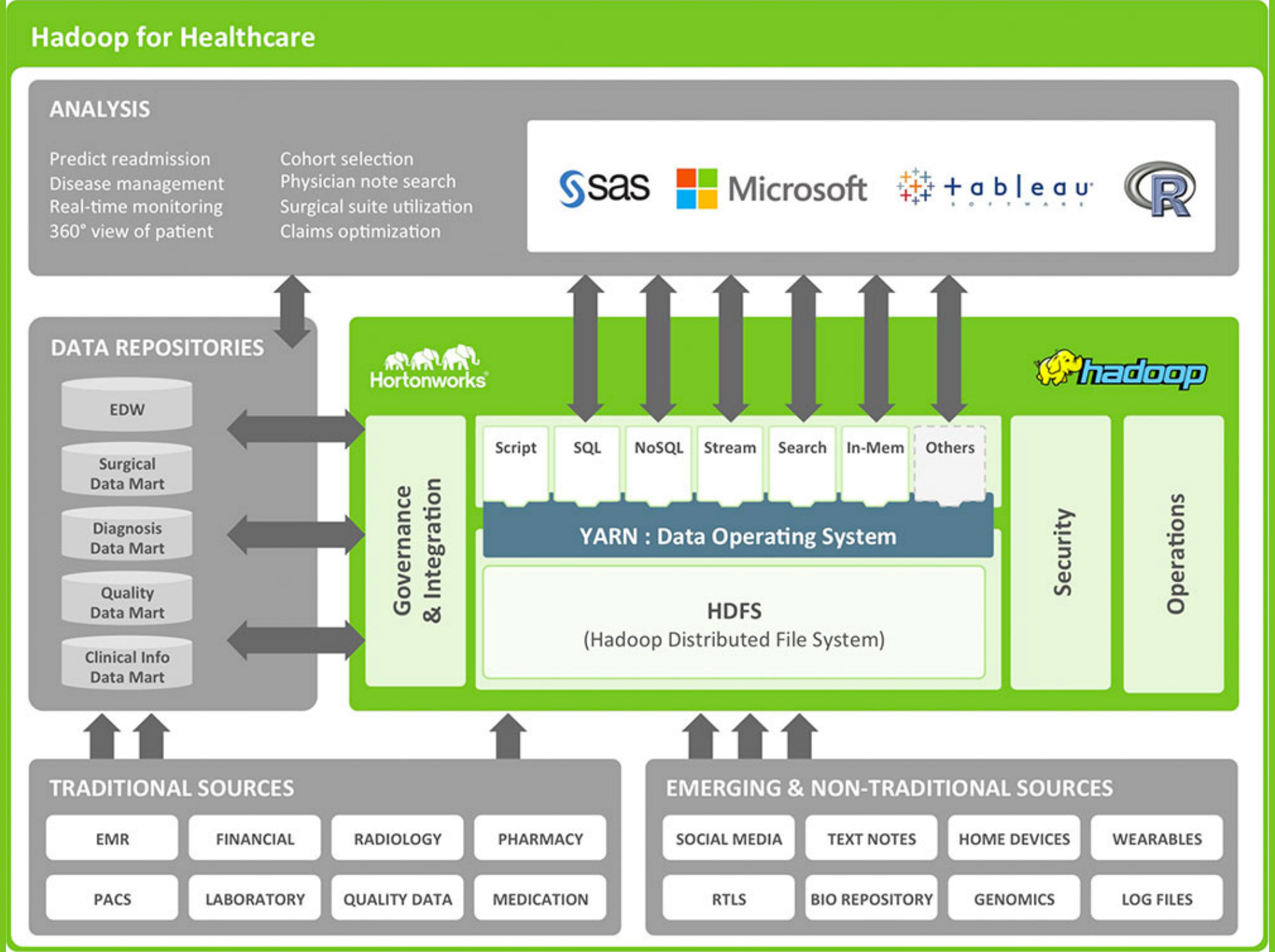
So, we've got big, complex data and big, complex problems

What does Hadoop really look like?

So, what does Hadoop look like?

(more specifically)

Sample Configuration



Apache Hadoop Ecosystem

Ambari
Provisioning, Managing and Monitoring Hadoop Clusters

Sqoop Data Exchange

Flume Log Collector

Zookeeper Coordination

Oozie Workflow

Pig Scripting

Mahout Machine Learning

R Connectors Statistics

Hive SQL Query

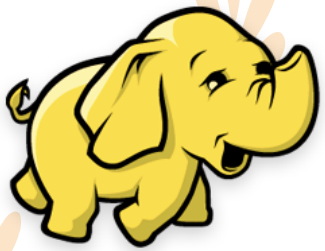
Hbase Columnar Store

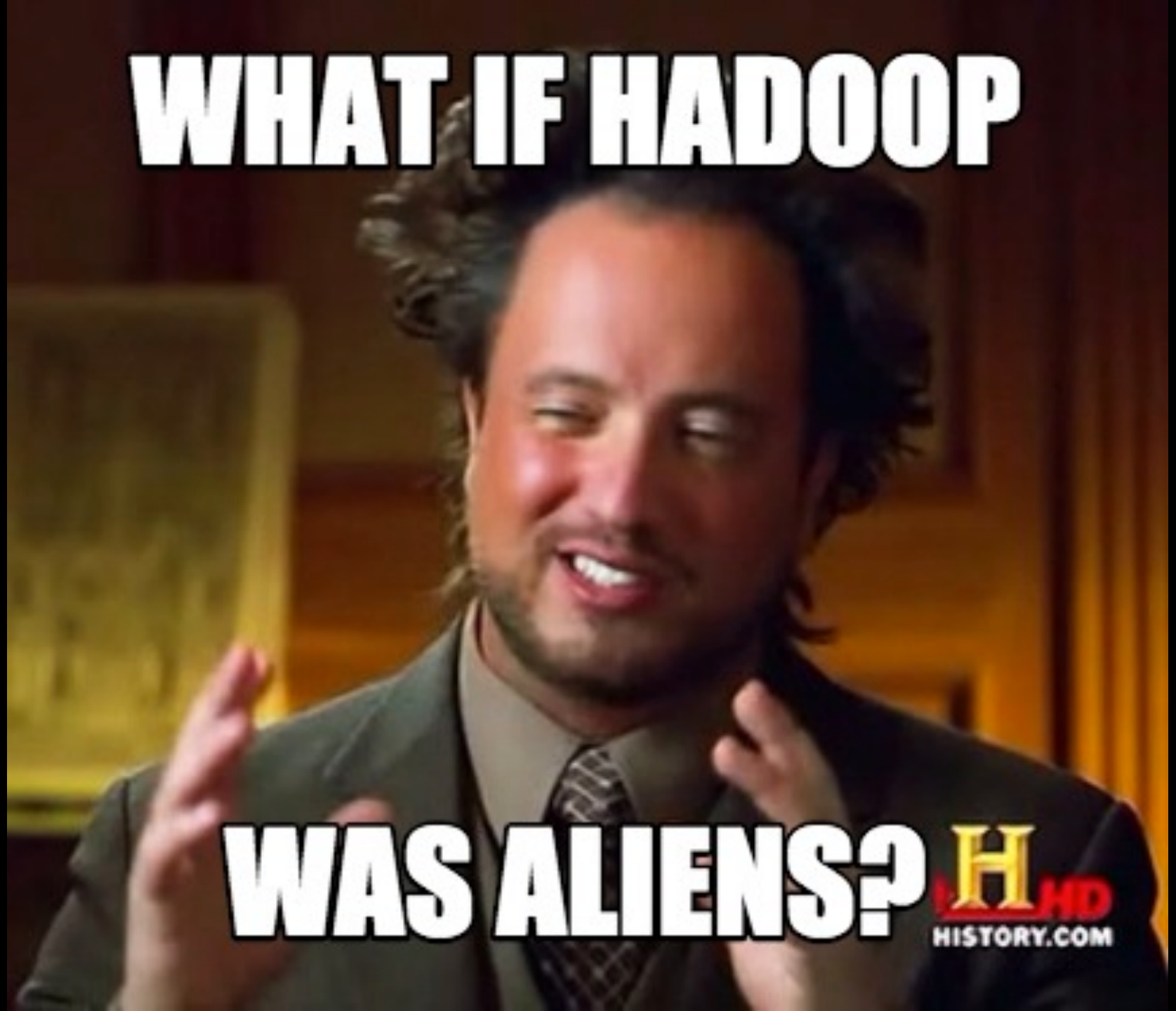
YARN Map Reduce v2
Distributed Processing Framework

HDFS
Hadoop Distributed File System

Brief History of Hadoop

- Created by Cutting and Cafarella in 2005, after needing a batch framework with the Apache Nutch project at Yahoo
- Hadoop named after Cutting's son's yellow plush toy
- Hadoop is now on release 2.6
- https://en.wikipedia.org/wiki/Apache_Hadoop#Timeline
- Google wrote 3 papers that were pivotal to Hadoop evolution
  1. Google File System –> HDFS
  2. MapReduce
  3. BigTable -> HBase



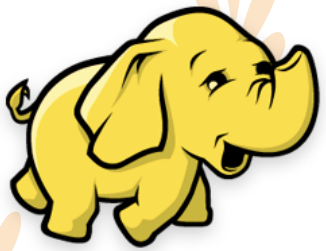

WHAT IF HADOOP
WAS ALIENS?
HISTORY.COM

What are the major components to Hadoop?

(Just 4!)

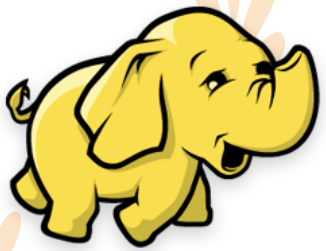
Hadoop Basics
(4 major components)

1. Hadoop Common
2. Hadoop Distributed File System (HDFS)
3. Hadoop Yarn
4. Hadoop MapReduce



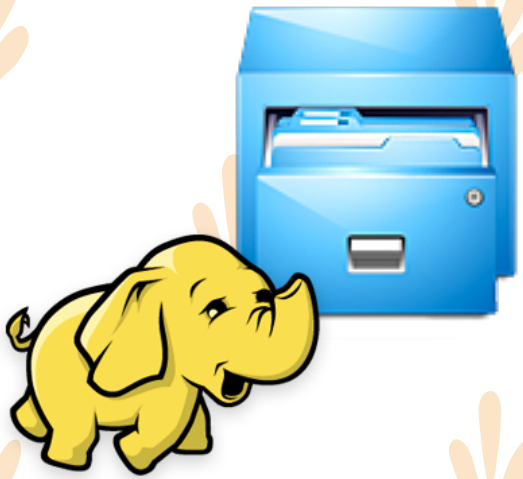
1. Hadoop Common

- Contains libraries and utilities needed by Hadoop modules.
- Also known as Hadoop Core



2. Hadoop Distributed File System (HDFS)

- Distributed file system storing data on commodity machines, providing good bandwidth across the cluster
- Provides high data throughput using MapReduce
- High fault tolerance
- Native support of large data sets

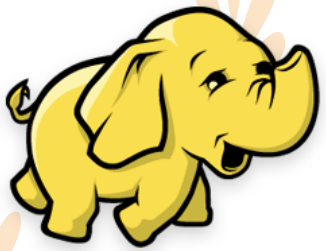


3. Hadoop YARN

- Resource management platform responsible for managing compute resources and schedules users and applications across those resources
- Also known as MapReduce 2.0

4. Hadoop MapReduce

- Programming model that scales data across several, distributed processes
- Popularized at Google to aid in web page indexing
- 2 types of processes
  - Map – takes in different data and distributes across the cluster
  - Reduce – takes the output from the cluster and reduces it down to one data set

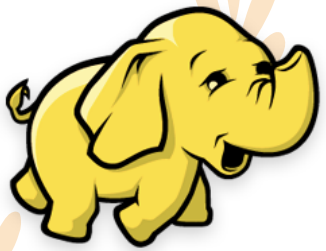


Now we know all there is to know about Hadoop, right?

Let’s explore the ecosystem

Hadoop Architecture

- Subscribes to a master/slave architecture based on Google infrastructure
- Every process is a master or a slave
- There are processes that worry about data, and processes that worry about jobs



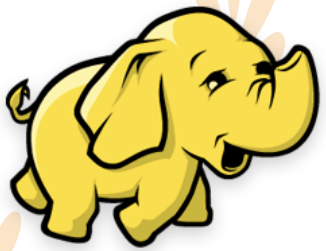
Hadoop Architecture

Master

- **NameNode**
- Centerpiece of the Hadoop File System (HDFS)
- Keeps a directory of all the files and where they are located
- Doesn't store any data
- This is a single point of failure unless a secondary NameNode is configured
- Give good care and feeding

Slave

- **DataNode**
- Where data lives
- There should be many in a production cluster, where data is replicated
- Communicates with NameNode and other DataNodes for processing and replication



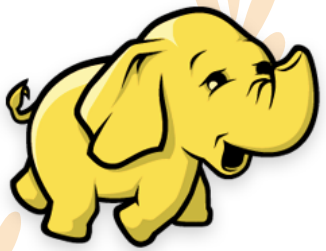
Hadoop Architecture

Master

- **JobTracker**
- Service that farms out processes to specific nodes in the cluster
- Communicates with NameNodes to get data location
- Finds TaskTrackers that run close to where data is located
- Monitors TaskTrackers and listens for heartbeat signals

Slave

- **TaskTracker**
- Node that accepts tasks: map, reduce, shuffle…
- Configured with a number of slots that be occupied to do work
- Spawns processes and monitors them
- Sends heartbeat messages to the JobTracker every few minutes to let the JobTracker know it's still alive (or not)



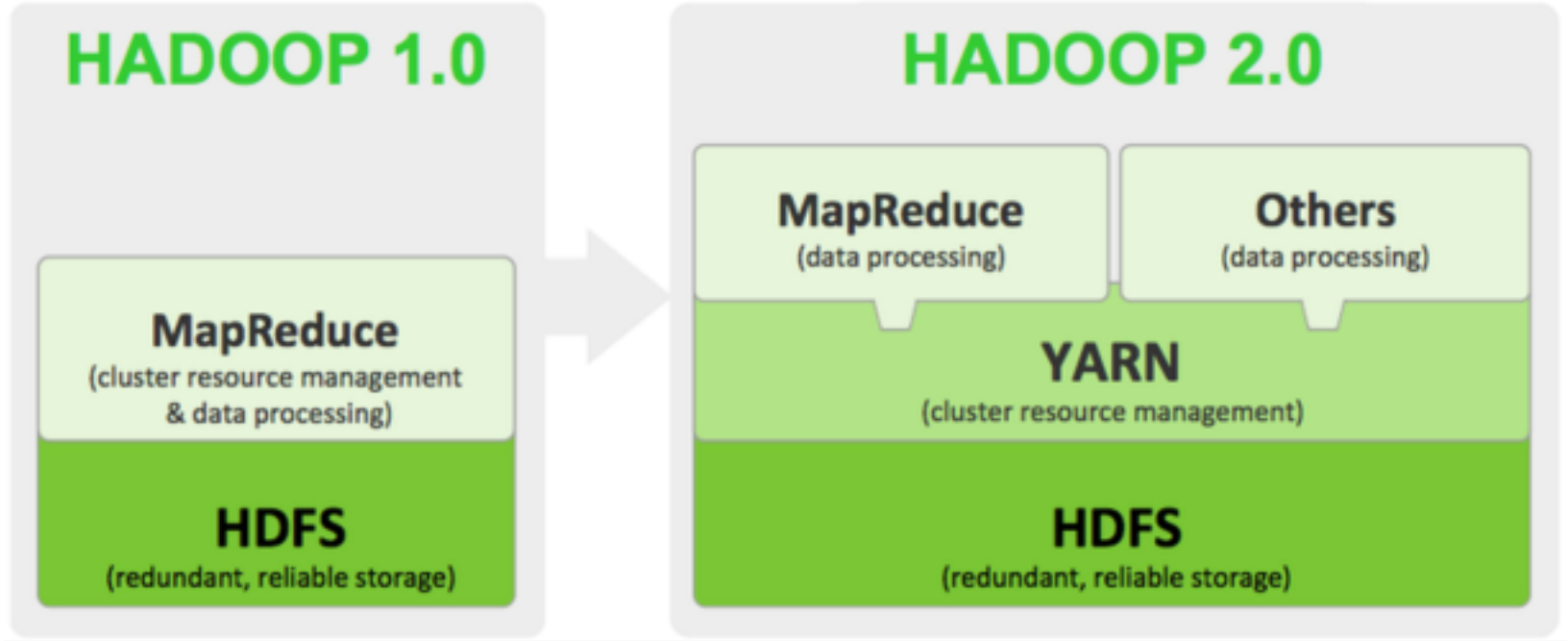
What about MapReduce and YARN?

Let’s talk about that…

MapReduce and Yarn

- Both aimed to do data processing and resource management
- MapReduce was Hadoop 1.x
- Yarn introduced in Hadoop 2.x. MapReduce still exists but only as a data processing framework
- MapReduce as a resource manager had limited resource utilization, and availability as its job tracker was a single point of failure.
- MapReduce had fixed map slots and fixed reduce slots, limiting usage
- MapReduce was challenging for jobs that didn't meet the MapReduce programming paradigm

MapReduce and Yarn

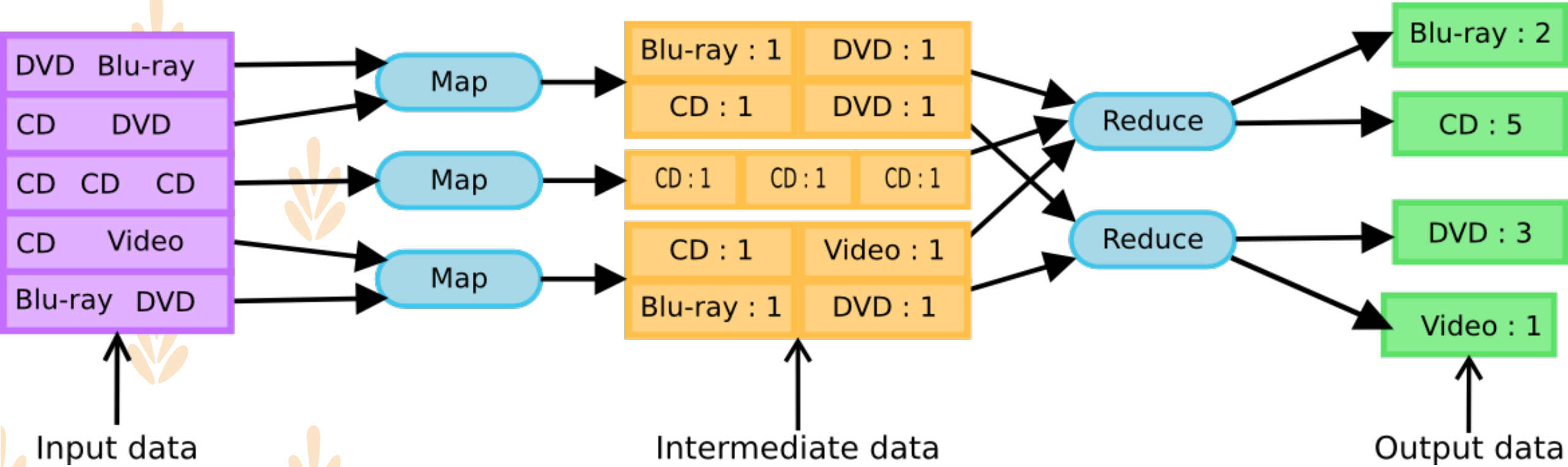


Yarn

- Yarn manages cluster resources and job scheduling
- Delegates MapReduce to a data processor only
- No fixed slots, so more efficient management of resources
- Can run applications that aren’t necessarily MapReduce
- Multiple processes can run across the cluster with shared resources, and Yarn can manage them successfully
- Yarn is backward compatible with MapReduce

MapReduce

- Counting example



How do we store data in Hadoop?

HDFS!

HDFS

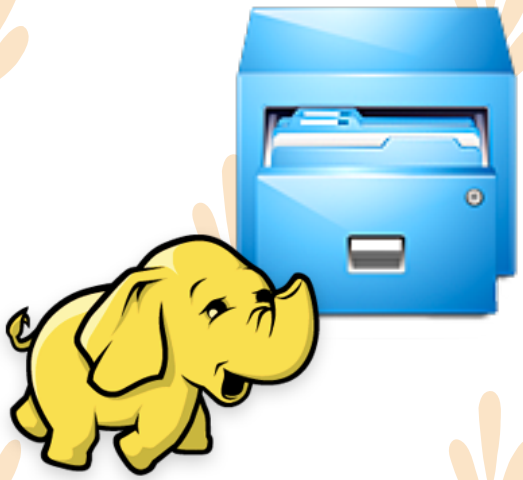
Challenge: Read 1 TB of data

1 machine	10 machines
- 4 I/O channels - Each channel: 100 MB/s	- 4 I/O channels - Each channel: 100 MB/s
=	=
45 Minutes ¼ TB on each hard drive	4.5 Minutes 1/10 TB on each machine 25 GB on each hard drive

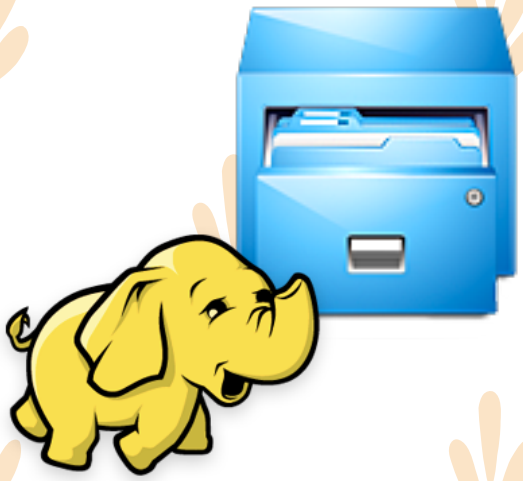
https://www.youtube.com/watch?v=ziqx2hJY8Hg

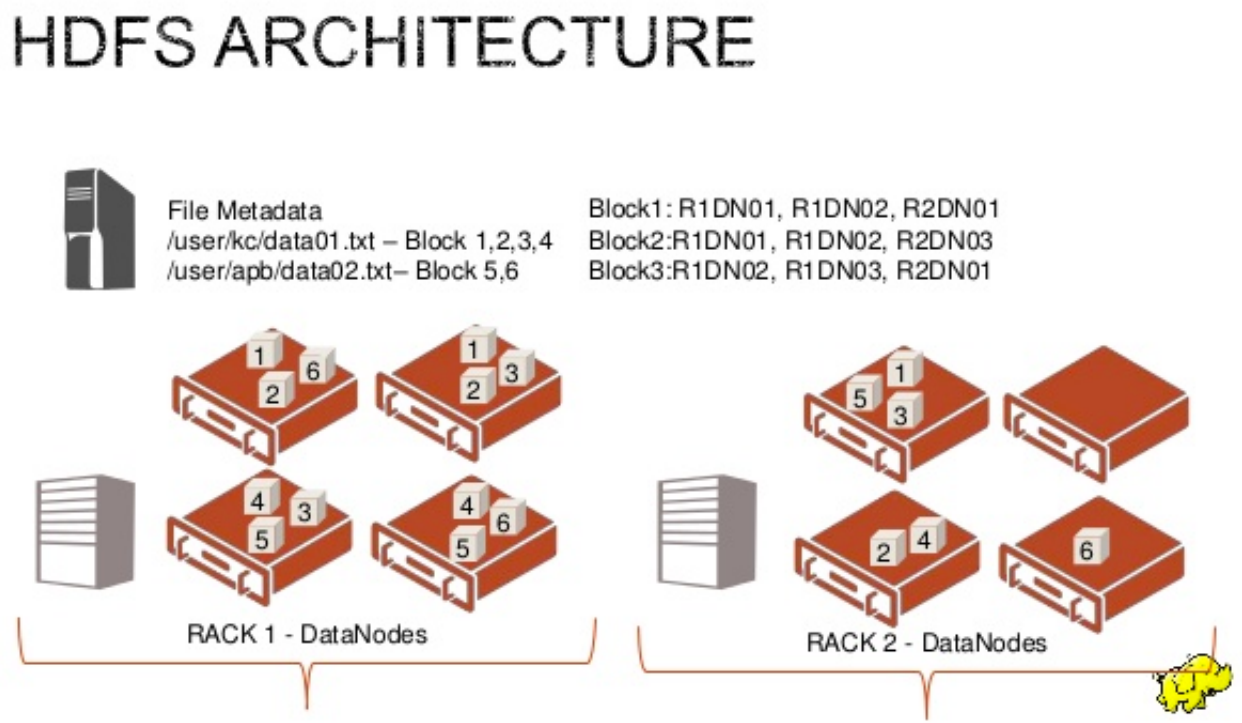
HDFS – The Hadoop File System

- NameNode stores file location
- DataNode stores data blocks (replicated)
- Replication factor can be set in *hdfs-site.xml*
- Highly available and self-healing, if a node dies, it no longer sends a heartbeat signal, that leads to a new replica of the data being spun up by the NameNode
- Underlying file system can be ext3, ext4 or XFS
- NameNode has web interface to show status



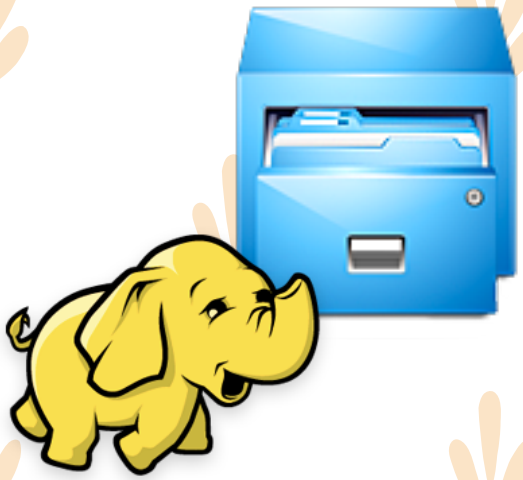
HDFS – Data replication example





HDFS – Commands

- `hdfs dfs -ls` (list files)
- `hdfs dfs -mkdir` (make directory)
- `hdfs dfs -copyFromLocal` (copy from local file system to HDFS)
- `hdfs dfs -copyToLocal` (copy from HDFS to local file system)
- `hdfs dfs -moveToLocal`
- `hdfs dfs -moveFromLocal`
- `hdfs dfs -rm` (remove)
- `hdfs dfs -tail` (view tail of file)
- `hdfs dfs -chmod` (change permissions)
- `hdfs dfs -setrep -w 4 -R /dir/s-dir` (change replication factor to 4 in this directory recursively)
- (can also use '`hadoop fs`' instead of 'hdfs dfs')



http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html

HBase

- Highly scalable, replicated data store, automatic sharding and failover support
- Data store rather than a relation database, e.g. key-value pairs
- Map (stored like map), persistent (data is persisted outside an application), distributed (data blocks are distributed on cluster), sorted (sorted by row name), multidimensional (column families can contain any number of columns), and sparse (good for sparse data sets).
- Interactive shell, and also Java APIs
- Modelled after Google's BigTable
- Can run in 2 modes: Distributed or Standalone
- Typically doesn't use MapReduce

HBase

- Not really ‘SQL like’ compared to Hive and Pig
- Must always query by unique row name
- Common attributes are stored together in ’column families’
- Data in ‘column families’ is stored together
- Schema is dynamic and new columns can be added on the fly
- All data is lexicographically sorted
- All data returns as a byte array in an application
- Resources
  - https://hbase.apache.org/book.html
  - http://jimbojw.com/wiki/index.php?title=Understanding_Hbase_and_BigTable
  - https://www.youtube.com/watch?v=KZps2dzr_u4

How HBase Represents Your Data

Row key	info: height	info:state	roles:hadoop	roles:hbase
cutting	'9ft'	'CA'	'Founder'	
tlipcon	'5ft7'	'CA'	'PMC' @ts=2011 'Committer' @ts=2010	'Committer'

Key: cutting/info:height/<timestamp> **Value**: '9ft'
Key: tlipcon/roles:hbase/<timestamp> **Value**: 'Committer'

HBase - Commands

- `create 'test', 'cf'` (create a table and column family)
- `list 'test'` (list info about the test table)
- `put 'test', 'row1', 'cf:a', 'value1'` (insert data (value1) at row1 (unique), column a, on the cf column family)
- `scan 'test'` (get all the rows in the test table)
- `get 'test', 'row1'` (get row1 of the test table)
- `disable 'test'` (disables the table)
- `enable 'test'` (enables the table)
- `drop 'test'` (drops table; must be disabled)

```
public static final byte[] CF = "cf".getBytes();
public static final byte[] ATTR = "attr".getBytes();
...

Table table = ...      // instantiate a Table instance

Scan scan = new Scan();
scan.addColumn(CF, ATTR);
scan.setRowPrefixFilter(Bytes.toBytes("row"));
ResultScanner rs = table.getScanner(scan);
try {
  for (Result r = rs.next(); r != null; r = rs.next()) {
    // process result...
  }
} finally {
  rs.close();  // always close the ResultScanner!
}
```

Can we query Hadoop data like SQL?

Well, sort of, mostly!

Hive

- Hive is ‘SQL on Hadoop’ - opens up Big Data to the masses
- This doesn’t mean Hadoop necessarily always meets the ACID (atomicity, consistency, isolation, durability) model
- Most GUI tools for Hadoop (e.g. Hue, Ambari) have Hive query windows
- Can support many data formats (HBase, flat files, etc.)
- Provides a layer of abstraction on top of MapReduce
- Initially developed at Facebook
- https://cwiki.apache.org/confluence/display/Hive/Home
- https://www.pluralsight.com/courses/sql-hadoop-analyzing-big-data-hive

Hive

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name
[(col_name data type [COMMENT col_comment], ...)]
[PARTITIONED BY (col_name data type [COMMENT col_comment], ...)]
[ROW FORMAT row_format] [STORED AS file_format]
[LOCATION hdfs_path]
[TBLPROPERTIES (property_name=property_value, ...)];
```

- External means => raw data is stored outside of Hive just has metadata
- Partitioning will split data into subdirectories (can be by day, location, etc.)
- Stored as => file format (textfile, avro, parquet, etc.)
- Row Format => delimited by, terminated by
- Location => Where in HDFS

Hive

```
CREATE EXTERNAL TABLE page_view(viewTime INT, userid
BIGINT,
     page_url STRING, referrer_url STRING,
     ip STRING COMMENT 'IP Address of the User',
     country STRING COMMENT 'country of origination')
 COMMENT 'This is the staging page view table'
 ROW FORMAT DELIMITED FIELDS TERMINATED BY '\054'
 STORED AS TEXTFILE
 LOCATION ’/user/system/page_views';
```

Query

Function	MySQL	HiveQL
Retrieving information	`SELECT from_columns FROM table WHERE conditions;`	`SELECT from_columns FROM table WHERE conditions;`
All values	`SELECT * FROM table;`	`SELECT * FROM table;`
Some values	`SELECT * FROM table WHERE rec_name = "value";`	`SELECT * FROM table WHERE rec_name = "value";`
Multiple criteria	`SELECT * FROM table WHERE rec1="value1" AND rec2="value2";`	`SELECT * FROM TABLE WHERE rec1 = "value1" AND rec2 = "value2";`
Selecting specific columns	`SELECT column_name FROM table;`	`SELECT column_name FROM table;`
Retrieving unique output records	`SELECT DISTINCT column_name FROM table;`	`SELECT DISTINCT column_name FROM table;`
Sorting	`SELECT col1, col2 FROM table ORDER BY col2;`	`SELECT col1, col2 FROM table ORDER BY col2;`
Sorting backward	`SELECT col1, col2 FROM table ORDER BY col2 DESC;`	`SELECT col1, col2 FROM table ORDER BY col2 DESC;`
Counting rows	`SELECT COUNT(*) FROM table;`	`SELECT COUNT(*) FROM table;`
Grouping with counting	`SELECT owner, COUNT(*) FROM table GROUP BY owner;`	`SELECT owner, COUNT(*) FROM table GROUP BY owner;`
Maximum value	`SELECT MAX(col_name) AS label FROM table;`	`SELECT MAX(col_name) AS label FROM table;`
Selecting from multiple tables (Join same table using alias w/"AS")	`SELECT pet.name, comment FROM pet, event WHERE pet.name = event.name;`	`SELECT pet.name, comment FROM pet JOIN event ON (pet.name = event.name);`

http://hortonworks.com/wp-content/uploads/downloads/2013/08/Hortonworks.CheatSheet.SQLtoHive.pdf

Metadata

Function	MySQL	HiveQL
Selecting a database	`USE database;`	`USE database;`
Listing databases	`SHOW DATABASES;`	`SHOW DATABASES;`
Listing tables in a database	`SHOW TABLES;`	`SHOW TABLES;`
Describing the format of a table	`DESCRIBE table;`	`DESCRIBE (FORMATTED|EXTENDED) table;`
Creating a database	`CREATE DATABASE db_name;`	`CREATE DATABASE db_name;`
Dropping a database	`DROP DATABASE db_name;`	`DROP DATABASE db_name (CASCADE);`

http://hortonworks.com/wp-content/uploads/downloads/2013/08/Hortonworks.CheatSheet.SQLtoHive.pdf

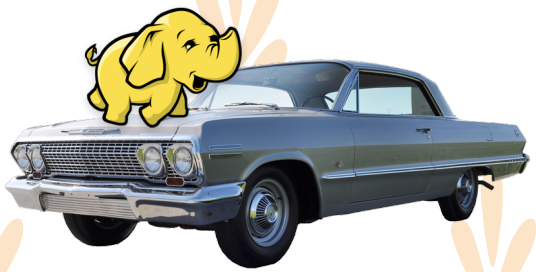
```
Connection con = DriverManager.getConnection("jdbc:hive://localhost:10000/default", "", "");
Statement stmt = con.createStatement();
String tableName = "testHiveDriverTable";
stmt.executeQuery("drop table " + tableName);
ResultSet res = stmt.executeQuery("create table " + tableName + " (key int, value string)");
// show tables
String sql = "show tables '" + tableName + "'";
System.out.println("Running: " + sql);
res = stmt.executeQuery(sql);
if (res.next()) {
  System.out.println(res.getString(1));
}
// describe table
sql = "describe " + tableName;
System.out.println("Running: " + sql);
res = stmt.executeQuery(sql);
while (res.next()) {
  System.out.println(res.getString(1) + "\t" + res.getString(2));
}

// load data into table
// NOTE: filepath has to be local to the hive server
// NOTE: /tmp/a.txt is a ctrl-A separated file with two fields per line
String filepath = "/tmp/a.txt";
sql = "load data local inpath '" + filepath + "' into table " + tableName;
System.out.println("Running: " + sql);
res = stmt.executeQuery(sql);

// select * query
sql = "select * from " + tableName;
System.out.println("Running: " + sql);
res = stmt.executeQuery(sql);
while (res.next()) {
  System.out.println(String.valueOf(res.getInt(1)) + "\t" + res.getString(2));
}
```

Are there any other SQL-like access tools?

Why, yes….

Impala

- High performance SQL query engine running on Hadoop
- Bypasses MapReduce to avoid latency (runs on data nodes)
- Can perform SELECT, JOIN, other SQL aggregate functions
- Same metadata and drivers as Hive
- Data can be stored in HBase or HDFS
- http://impala.io/index.html



Drill

- Open source query engine for big data
- 'Real' SQL - DATE, INTERVAL, TIMESTAMP, and VARCHAR, subqueries, joins, etc.
- Allows for Schema free model (like MongoDB or Elasticsearch)
- No need to flatten or transform data prior to querying
- Works on multiple data sources (Hive, HBase, HDFS, local files)
- https://drill.apache.org/

Phoenix

- Open Source SQL wrapper for HBase (compiles SQL to HBase scans)
- Uses standards JDBC APIs
- Compiles SQL to native HBase commands
- Secondary indexes to improve performance
- Calculates optimal stop/start keys for scans
- Good or better performance
- https://phoenix.apache.org/index.html

How do we load data into Hadoop?

There’s a tool for that!

Sqoop – SQL on Hadoop

- ETL tool to load data from SQL databases to HDFS (and Hive, and HBase)
- Creates a Java (MapReduce) program under the covers, and distributes across the cluster
- Uses a sqoop connector that knows how to map fields through a JDBC driver (may need to load into Hadoop)

Sqoop – Commands

- `sqoop list-databases --connect jdbc:mysql://localhost/ --username root -P` (lists all databases, prompts for password)
- `sqoop list-tables --connect jdbc:mysql://localhost/serviceorderdb --username root -P` (lists all tables, prompts for password)
- `sqoop import --connect jdbc:mysql://localhost/databasename --username $USER_NAME --password $PASSWORD$ --table tablename --m 1` (import table, passing in password, with one mapper)
- (You can also use a config file to store passwords)

Sqoop – Commands

- `sqoop import --connect jdbc:mysql://localhost/testDb --username root –P --table student --where "id>1" -m 1 --target-dir /user/hdfsuser/mystudents` (imports data from student table, where id > 1 into /user/hdfsuser/mystudents with one mapper)
- (Multiple mappers requires a primary key)
- `sqoop import-all-tables -m 1 --connect jdbc:postgresql://laertes.ohdsi.org:5432/vocabularyv5 --username=username -P --hive-import --hive-database cdmv5 -- --schema=unrestricted` (import all tables in unrestricted schema into Hive)

Sqoop – Commands

- `sqoop import -m 2 --connect jdbc:postgresql://laertes.ohdsi.org:5432/vocabularyv5 --username=username -P --table cohort --split-by cohort_definition_id --hive-import -hive-table cdmv5.cohort` (import cohort table into Hive, passing in unique key so job can be split across 2 mappers)
- `sqoop import --connect jdbc:mysql://localhost/serviceorderdb --username root -P --table customercontactinfo --columns "customernum,customername" --hbase-table customercontactinfo --column-family CustomerName --hbase-row-key customernum -m 1` (importing into HBase)

What about this Pig thing I heard about once?

Oink!

Pig

- High level data analysis application framework
- Can run local or distributed on a cluster
- Language called Pig Latin, compiled to MapReduce jobs
- Similar functionality to SQL, but can be extended with custom functions in Java, Python and Ruby
- Provides a command line tool called grunt
- Doesn’t require Java coding experience, SQL developers can adapt to using Pig fairly easily
- Can also be used to ETL files from HDFS to HBase
- Originally built at Yahoo, so data analysts could write MapReduce jobs

Pig

- Procedural, Data flow language, so complex sequences are split up, step by step
- Can use regular expressions and other Java/SQL functions
- Schemas can be defined at runtime (can run on many data sources, like flat files, HBase, local files)
- Resources
  - Language Basics - http://pig.apache.org/docs/r0.15.0/basic.html
  - Functions - http://pig.apache.org/docs/r0.15.0/func.html
  - https://www.pluralsight.com/courses/pig-latin-getting-started
  - https://cwiki.apache.org/confluence/display/PIG/Index
  - Pig v. SQL - https://developer.yahoo.com/blogs/hadoop/comparing-pig-latin-sql-constructing-data-processing-pipelines-444.html

Pig vs. SQL

SQL

```
select * from patients
where age > 18;
```

(declarative)

Pig Latin

```
patients = FOREACH
patient_list GENERATE id,
name, age;

patients_filtered = FILTER
patients BY age > 18;

DUMP patients_filtered;
```

(procedural)

For this simple example, SQL is simpler, but with bigger data sets, and more complex problems, Pig can be more descriptive and much more powerful, for some recent examples I did, see https://tools.regenstrief.org/wiki/display/bigdata/NLP+Pig+Scripts

```
input_lines = LOAD '/tmp/word.txt' AS (line:chararray);
words = FOREACH input_lines GENERATE FLATTEN(TOKENIZE(line)) AS word;
filtered_words = FILTER words BY word MATCHES '\\w+';
word_groups = GROUP filtered_words BY word;
word_count = FOREACH word_groups GENERATE COUNT(filtered_words) AS count, group AS word;
ordered_word_count = ORDER word_count BY count DESC;
STORE ordered_word_count INTO '/tmp/results.txt';
```

Pig Latin Word Count

7 Lines vs. 45 Lines

SQL like syntax

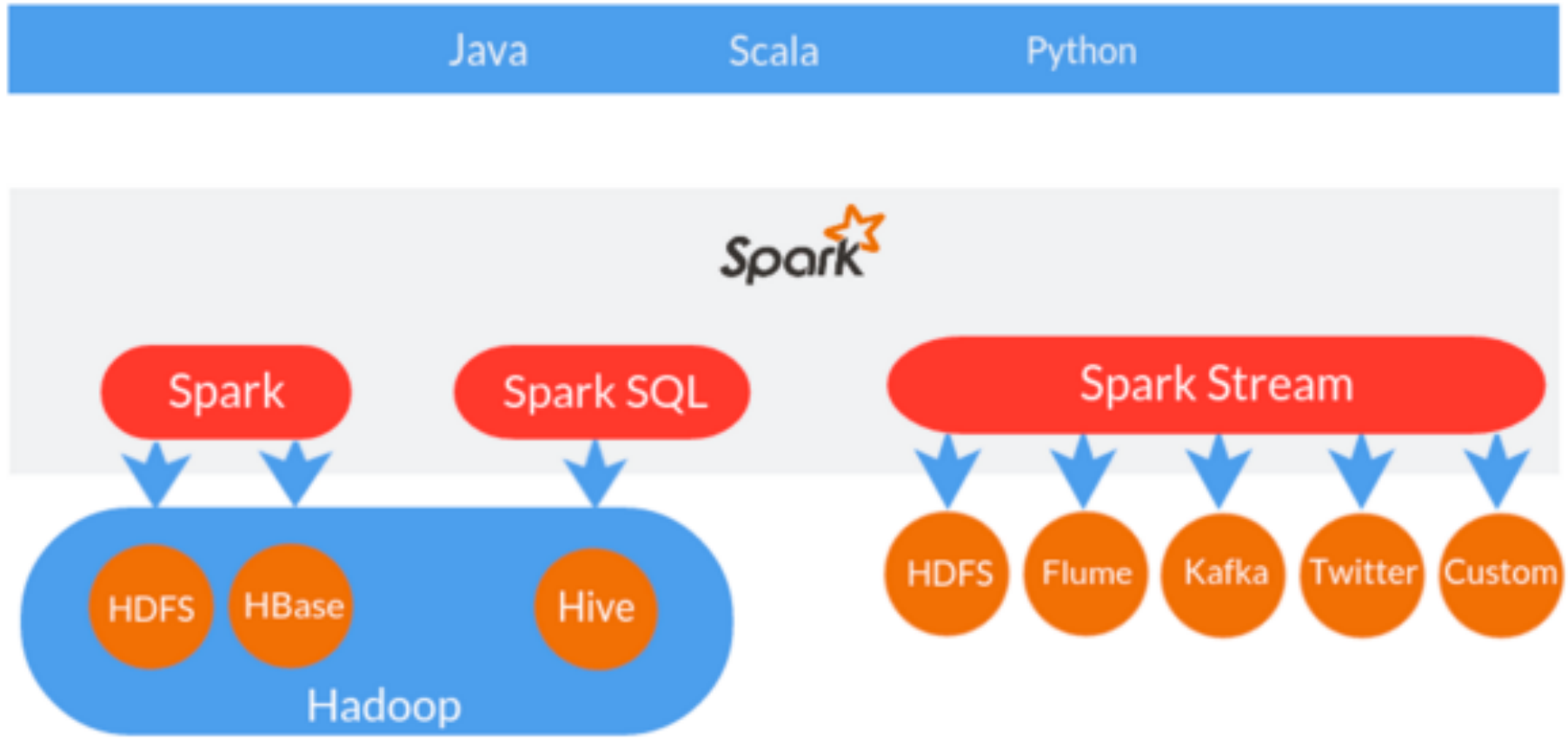
```
import java.io.IOException;
import java.util.*;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class WordCount {

 public static class Map extends Mapper {
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedExcep
        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            word.set(tokenizer.nextToken());
            context.write(word, one);
        }
    }
 }

 public static class Reduce extends Reducer {

    public void reduce(Text key, Iterable values, Context context)
      throws IOException, InterruptedException {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
 }
```

What about machine learning?

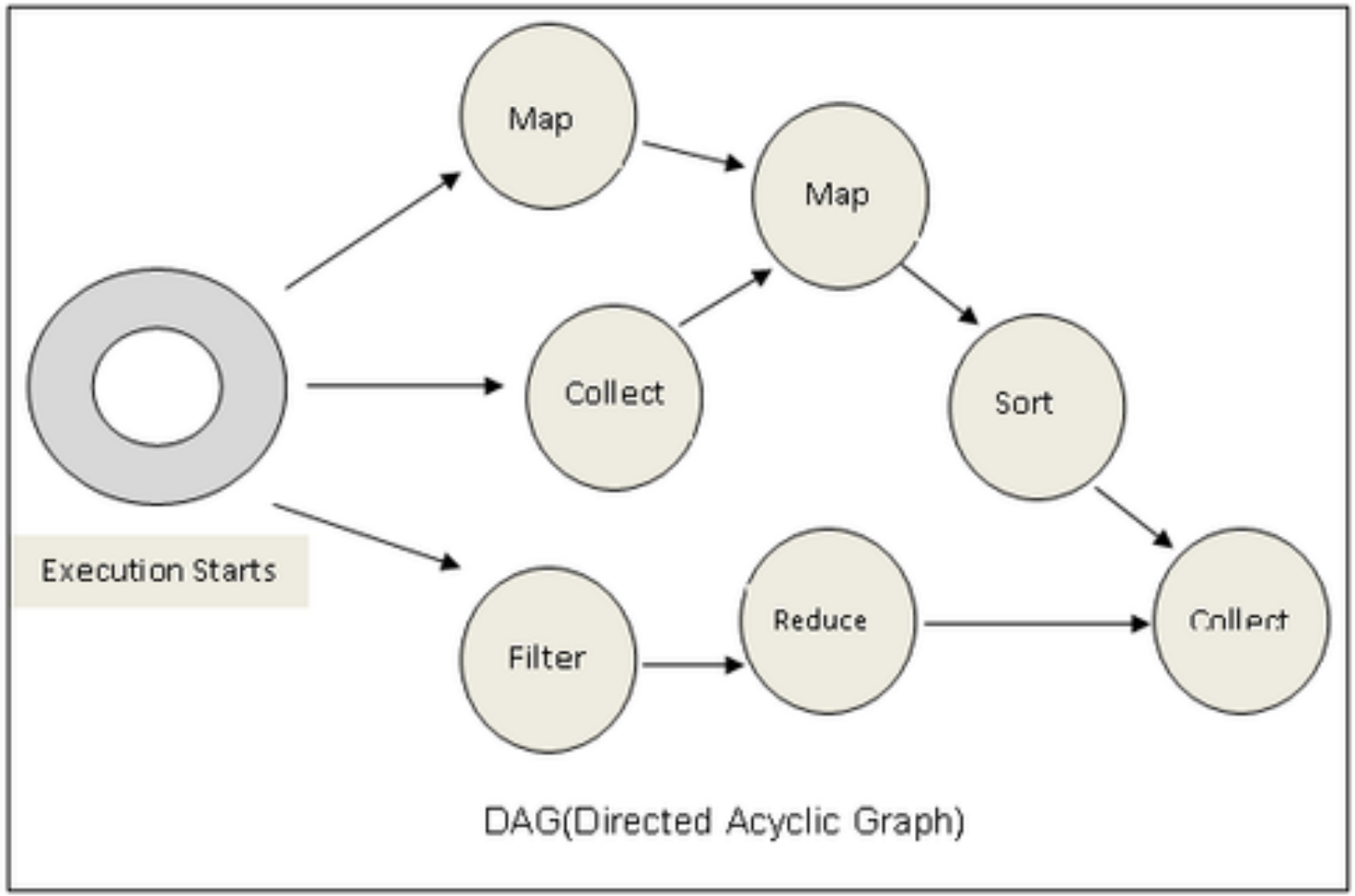
I heard Hadoop can do that.

Spark

- Fast, in-memory data processing system
- Up to 100x faster than MapReduce
- Runs on YARN
- More than just Machine Learning!
- Has built-in APIs for Streaming, SQL, Graph Processing and Machine Learning
- Originally developed at UC Berkeley's AMPLab
- Built mostly in Scala, but also supports Java, Python, R
- Can also run standalone, outside Hadoop
- DevU in April
- https://spark.apache.org/



http://aptuz.com/blog/is-apache-spark-going-to-replace-hadoop/


Map
Map
Collect
Sort
Execution Starts
Filter
Reduce
Collect
DAG(Directed Acyclic Graph)

Mahout

- Produces implementations of scalable machine learning algorithms
- Areas include collaborative filtering (e.g. Matrix Factorization with ALS), classification (e.g. Random Forest), clustering (e.g. k-Means Clustering) and generic math and collections modules
- Built to work with Spark and H20 (http://www.h2o.ai/)
- Written in Scala and Java
- http://mahout.apache.org/

Collaborative Filtering

User-Based Collaborative Filtering

Item-Based Collaborative Filtering

Matrix Factorization with ALS

Matrix Factorization with ALS on Implicit Feedback

Weighted Matrix Factorization, SVD++

Classification

Logistic Regression - trained via SGD

Naive Bayes / Complementary Naive Bayes

Random Forest

Hidden Markov Models

Multilayer Perceptron

Clustering

Canopy Clustering

k-Means Clustering

Fuzzy k-Means

Streaming k-Means

Spectral Clustering

Dimensionality Reduction *note: most scala-based dimensionality reduction algorithms are available through the* Mahout Math-Scala Core Library for all engines
Singular Value Decomposition

Lanczos Algorithm

Stochastic SVD

PCA (via Stochastic SVD)

QR Decomposition

Topic Models

Latent Dirichlet Allocation

Miscellaneous

RowSimilarityJob

ConcatMatrices

Collocations

Sparse TF-IDF Vectors from Text

XML Parsing

Email Archive Parsing

Lucene Integration

Evolutionary Processes

Mahout Math-Scala Core Library and Scala DSL

Mahout Distributed BLAS. Distributed Row Matrix API with R and Matlab like operators. Distributed ALS, SPCA, SSVD, thin-QR. Similarity Analysis.

Mahout Interactive Shell

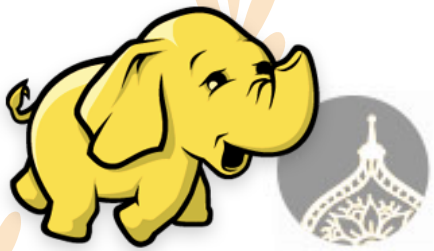
Interactive REPL shell for Spark optimized Mahout DSL

Isn’t there a front end user interface for Hadoop?

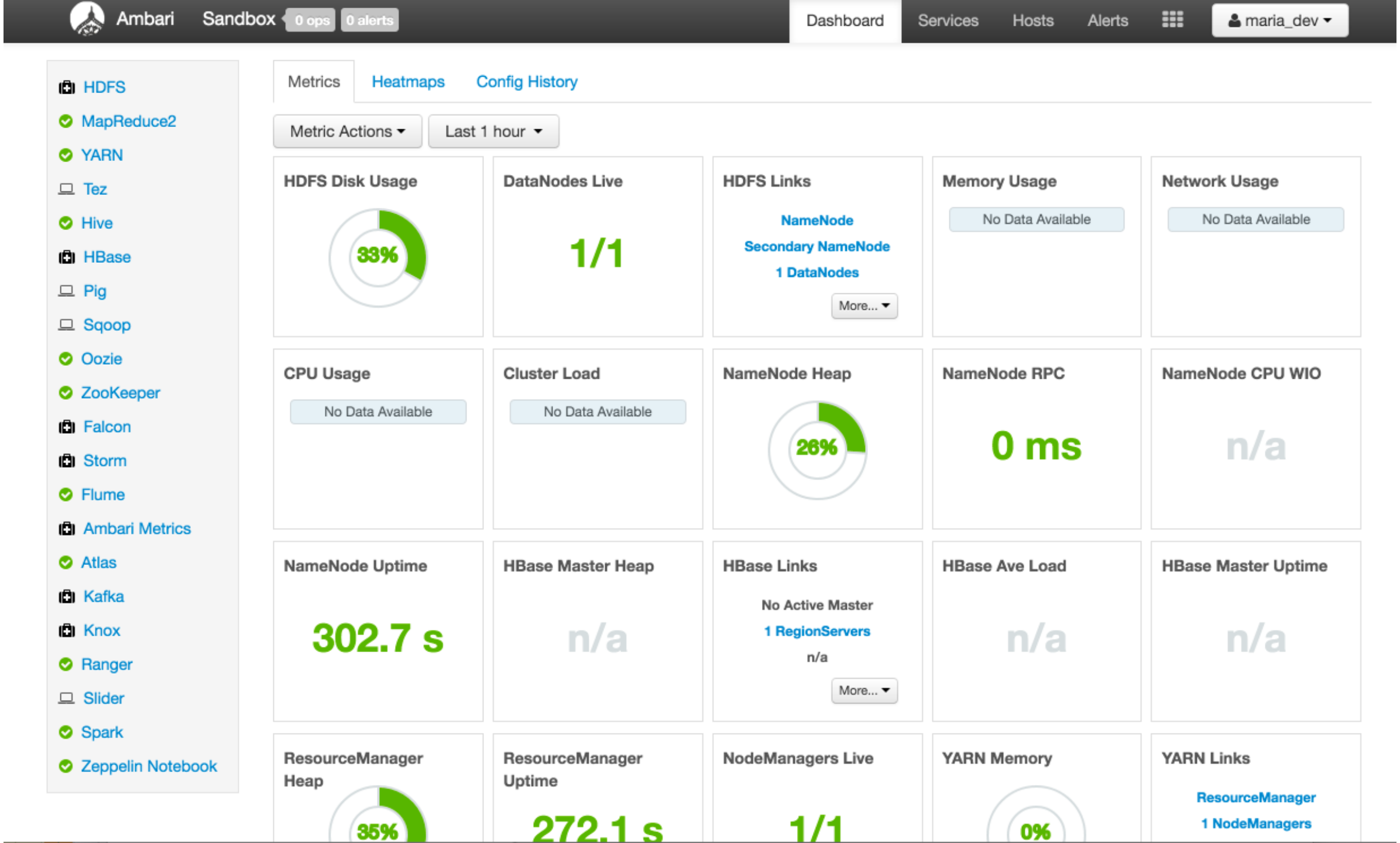
Of course!

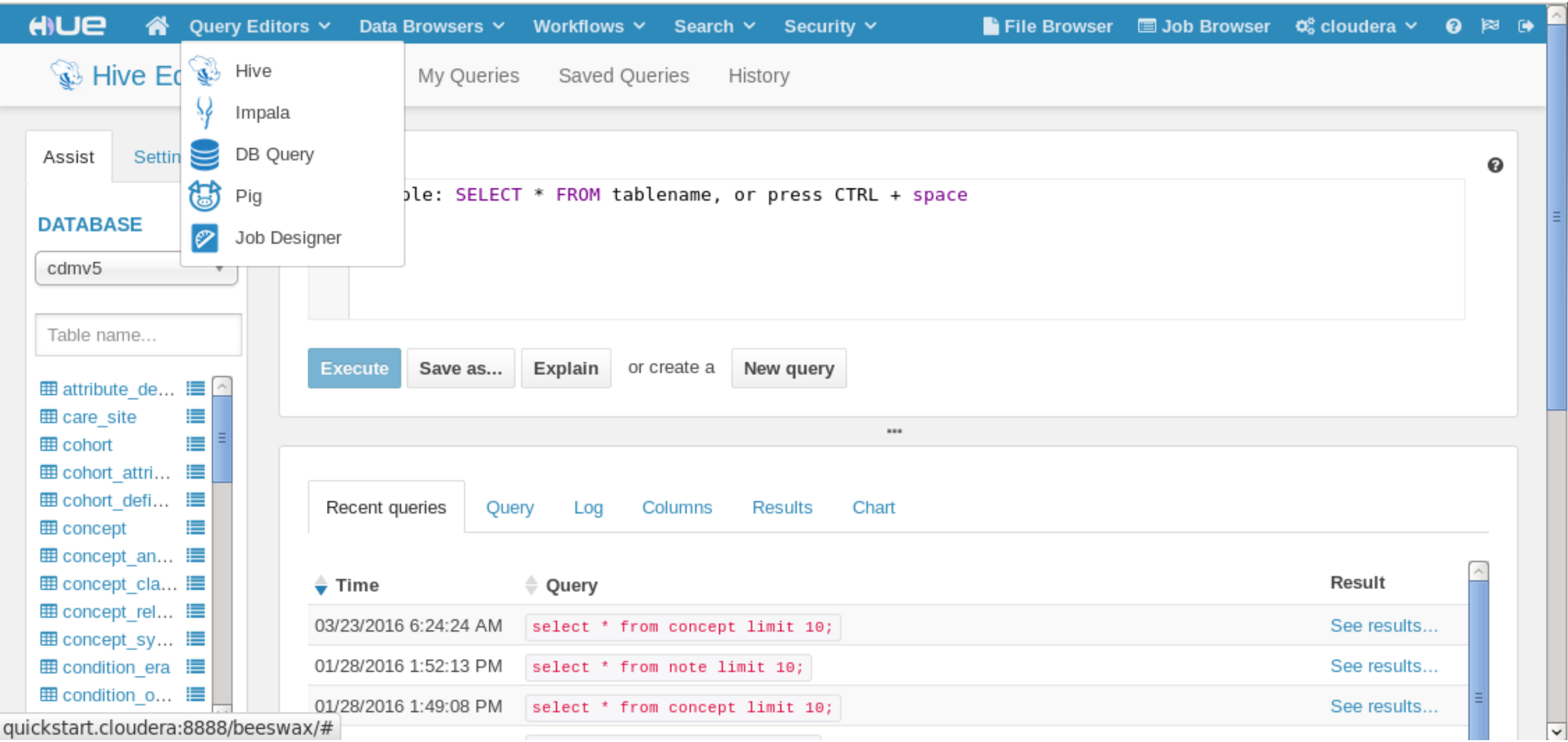
Ambari/HUE

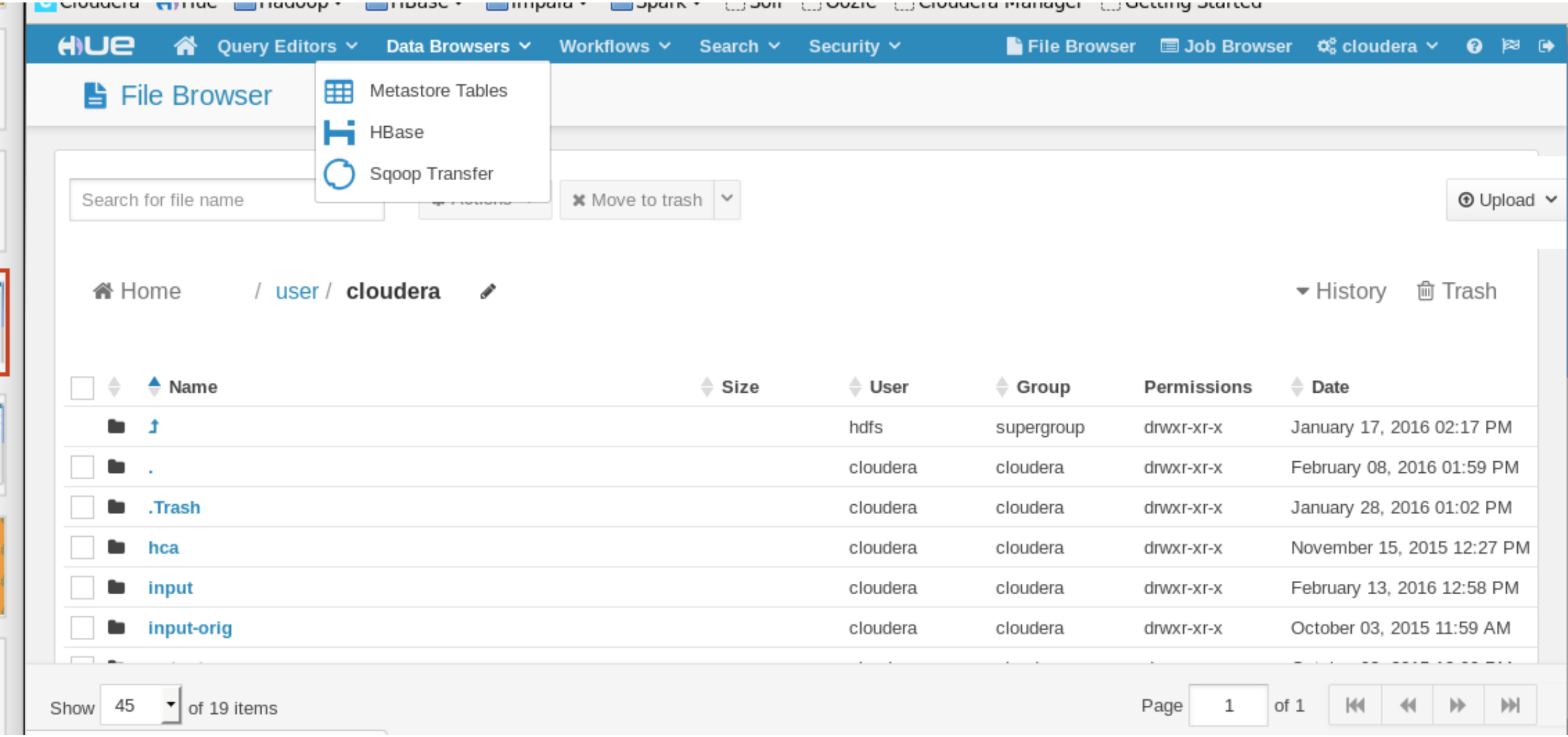
- Tool for managing data settings, view cluster performance, systems up/down status
- Has interactive tools for navigating HDFS, Hive, etc.
- Delivered based on distribution



Apache Ambari HUE


Ambari
Sandbox
0 ops
0 alerts
Dashboard
Services
Hosts
Alerts
maria_dev
HDFS
MapReduce2
YARN
Tez
Hive
HBase
Pig
Sqoop
Oozie
ZooKeeper
Falcon
Storm
Flume
Ambari Metrics
Atlas
Kafka
Knox
Ranger
Slider
Spark
Zeppelin Notebook
Metrics
Heatmaps
Config History
Metric Actions
Last 1 hour
HDFS Disk Usage
33%
DataNodes Live
1/1
HDFS Links
NameNode
Secondary NameNode
1 DataNodes
More...
Memory Usage
No Data Available
Network Usage
No Data Available
CPU Usage
No Data Available
Cluster Load
No Data Available
NameNode Heap
26%
NameNode RPC
0 ms
NameNode CPU WIO
n/a
NameNode Uptime
302.7 s
HBase Master Heap
n/a
HBase Links
No Active Master
1 RegionServers
n/a
More...
HBase Ave Load
n/a
HBase Master Uptime
n/a
ResourceManager Heap
35%
ResourceManager Uptime
272.1 s
NodeManagers Live
1/1
YARN Memory
0%
YARN Links
ResourceManager
1 NodeManagers


HUE
Query Editors
Data Browsers
Workflows
Search
Security
File Browser
Job Browser
cloudera
Hive
Impala
DB Query
Pig
Job Designer
My Queries
Saved Queries
History
Assist
DATABASE
cdmv5
Table name...
attribute_de...
care_site
cohort
cohort_attri...
cohort_defi...
concept
concept_an...
concept_cla...
concept_rel...
concept_sy...
condition_era
condition_o...
ble: SELECT * FROM tablename, or press CTRL + space
Execute
Save as...
Explain
or create a
New query
Recent queries
Query
Log
Columns
Results
Chart
Time
Query
Result
03/23/2016 6:24:24 AM
select * from concept limit 10;
See results...
01/28/2016 1:52:13 PM
select * from note limit 10;
See results...
01/28/2016 1:49:08 PM
select * from concept limit 10;
See results...
quickstart.cloudera:8888/beeswax/#

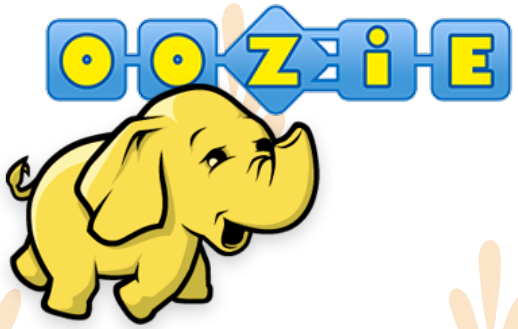

HUE
Query Editors
Data Browsers
Workflows
Search
Security
File Browser
Job Browser
cloudera
Metastore Tables
HBase
Sqoop Transfer
File Browser
Search for file name
Move to trash
Upload
Home / user / cloudera
History
Trash
Name
Size
User
Group
Permissions
Date
hdfs supergroup drwxr-xr-x January 17, 2016 02:17 PM
. cloudera cloudera drwxr-xr-x February 08, 2016 01:59 PM
.Trash cloudera cloudera drwxr-xr-x January 28, 2016 01:02 PM
hca cloudera cloudera drwxr-xr-x November 15, 2015 12:27 PM
input cloudera cloudera drwxr-xr-x February 13, 2016 12:58 PM
input-orig cloudera cloudera drwxr-xr-x October 03, 2015 11:59 AM
Show 45 of 19 items
Page 1 of 1

What other Hadoop tools are out there?

Plenty, and more I don’t cover here!

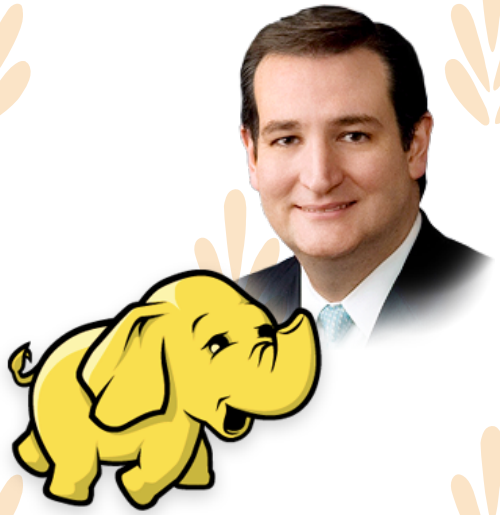
Oozie

- Workflow coordinator for Hadoop
- Directed Acyclical Graphs (DAGs) of actions
- Oozie can support different kinds of operations: MapReduce, Pig, e-mail, SSH, HDFS
- Written in Java
- https://oozie.apache.org/index.html



Tez

- Developer Framework to write native YARN applications
- Allows for a complex directed-acyclic-graph (DAG) of tasks
- Better performing than MapReduce (Hindi for “speed”)
- Interoperates with the Hadoop ecosystem, Pig, Hive, etc.
- Java API to define DAG (overall job), vertex (user logic), edge (connection between vertices)
- https://tez.apache.org/



Zookeeper

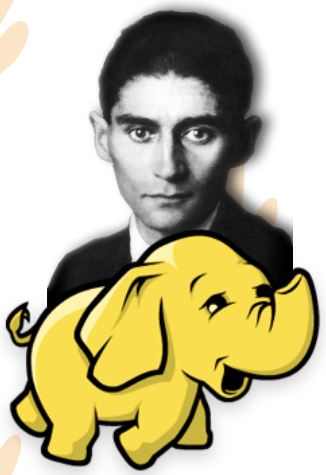
- Centralized service for maintaining configuration and synchronization among distributed services
- Supports high availability services through redundancy
- Stores data in a hierarchical tree structure
- Written in Java
- Used by Solr, ElasticSearch, HBase, Flume, Kafka, many others
- https://zookeeper.apache.org/

Storm

- Distributed Computation Framework
- Written in Clojure and Java
- Designed in a DAG (directed acyclical graph) where spouts and bolts are vertices, in a ‘topology’
- Spouts bring in data streams, bolts process and distribute data
- Data is processed in real-time
- http://storm.apache.org/

Kafka

- Fast, scalable, durable distributed messaging system
- A single Kafka broker can handle hundreds of megabytes of reads and writes per second from thousands of clients.
- Designed to allow a single cluster to serve as the central data backbone for a large organization. Elastically and transparently expanded without downtime.
- Messages are persisted on disk and replicated within the cluster to prevent data loss.
- Modern cluster-centric design that offers strong durability and fault-tolerance guarantees
- Developed at LinkedIn (written in Scala)
- http://kafka.apache.org/



Flume

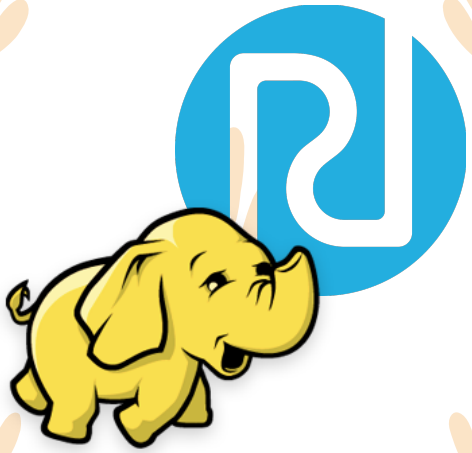
- Distributed tool for collecting, aggregating and moving large amounts of log data
- Built on streaming data model
- Uses YARN to coordinate data flows
- Guarantees data delivery
- Integrates with Kafka
- https://flume.apache.org/

Well, we're almost done.

But this is just the beginning for your Hadoop journey (I hope).

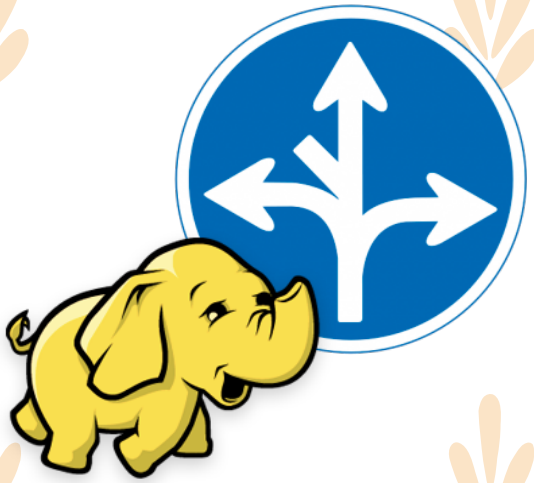
Hadoop at Regenstrief

- Hadoop Cluster providing Distributed Data Storage and Distributed processing via MapReduce.
  9 Data nodes - @ 64GB - 8 1TB Disks
- Currently NLP is using HBase (and has plans for Storm (sooner) and Spark (soon))
- Starting the process of updating Hadoop to add new features (Spark) and more fine grain user access
- Additional hardware is being configured – See David or Clint for questions
- https://tools.regenstrief.org/wiki/display/bigdata/RI+Data+Cluster



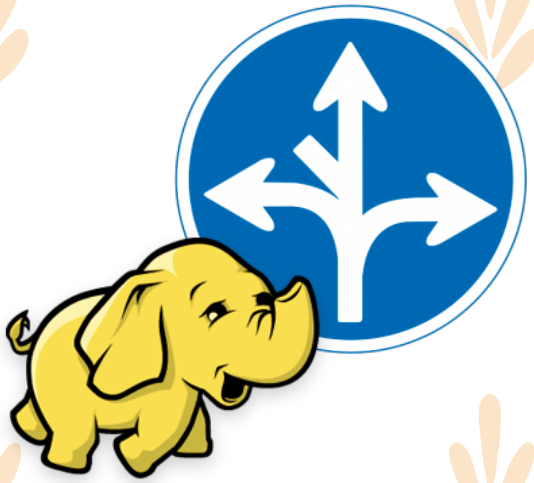
What's next?

- Download the Hortonworks (or Cloudera or MapR) sandboxes
  - Follow tutorials (you'll need VirtualBox or VmWare)
  - Get hands on!
- Take a MOOC or checkout Hadoop on Safari
- Check out the Hadoop Summit – in person or on YouTube - https://www.youtube.com/user/HadoopSummit
- Read the Hadoop wiki - https://wiki.apache.org/hadoop/
- Join the Healthcare Hadoop Users Group – ask me, David or Chris B.
- Come to the TLW Learning session – Friday 3/25, 1PM, RF 209/210
- Support Hadoop at Regenstrief! Talk to me, David, Clint or Chris B.



What's next? - Links

- VirtualBox - https://www.virtualbox.org/wiki/Downloads
- Hadoop sandboxes
  - Hortonworks sandbox - http://hortonworks.com/products/hortonworks-sandbox/#install
  - Cloudera sandbox - http://www.cloudera.com/downloads/quickstart_vms/5-5.html
  - MapR sandbox - https://www.mapr.com/products/mapr-sandbox-hadoop/download
- Tutorials and MOOCs
  - http://hortonworks.com/tutorials/
  - https://www.coursera.org/specializations/big-data
  - https://www.udacity.com/course/intro-to-hadoop-and-mapreduce--ud617
  - https://www.mapr.com/products/mapr-sandbox-hadoop/tutorials
  - …and lots more



Thanks!