UFPR - Universidade Federal do Paraná
SEPT - Setor de Educação Profissional e Tecnológica

IAA - Especialização em Inteligência Artificial Aplicada

IAA013 - Big Data (Parte 2)

Prof. **João Eugenio** Marynowski – jeugenio@ufpr.br

Programa

- Fundamentos de Big Data
  - Big Data, Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - **Personalização de aplicações Map Reduce**
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
- NoSQL e NewSQL
- Dados em movimento – Processamento de Streaming

Big Data é uma Buzzword

3Vs – Volume, Variedade e Velocidade

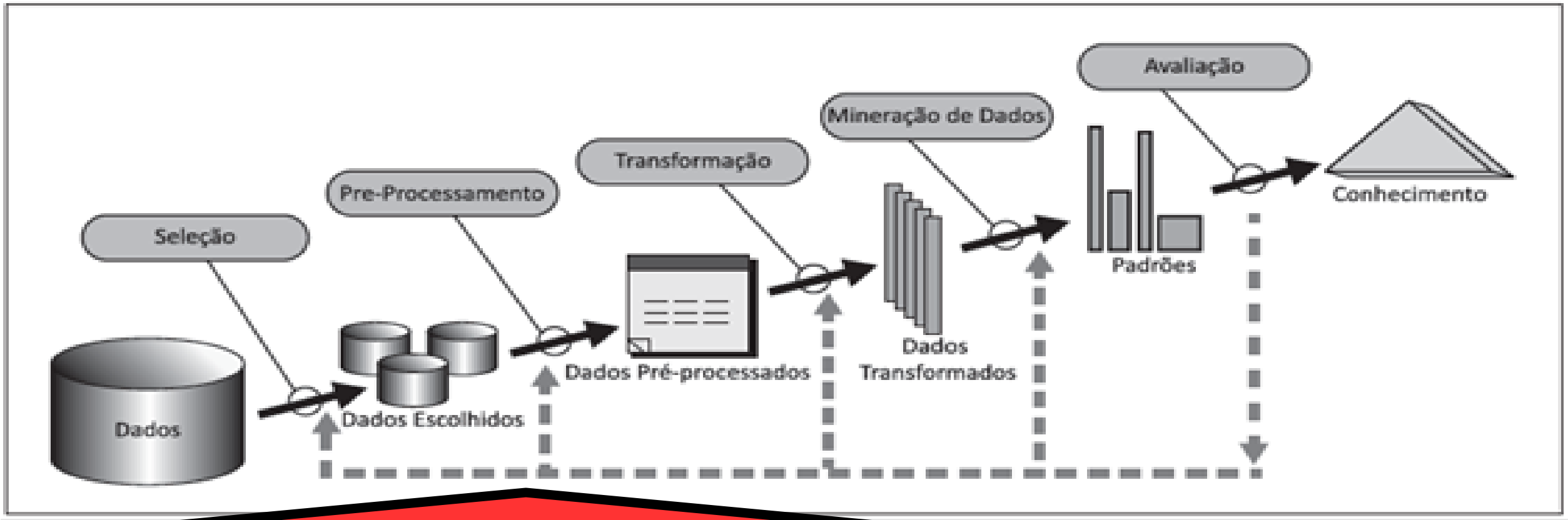
Processamento de grande volumes de dados não-estruturados para tomada de decisão em "tempo real"

UFPR

Especialização em Inteligência Artificial

SEPT

Processo de Descoberta de Conhecimento
KDD - Knowledge Discovery Databases

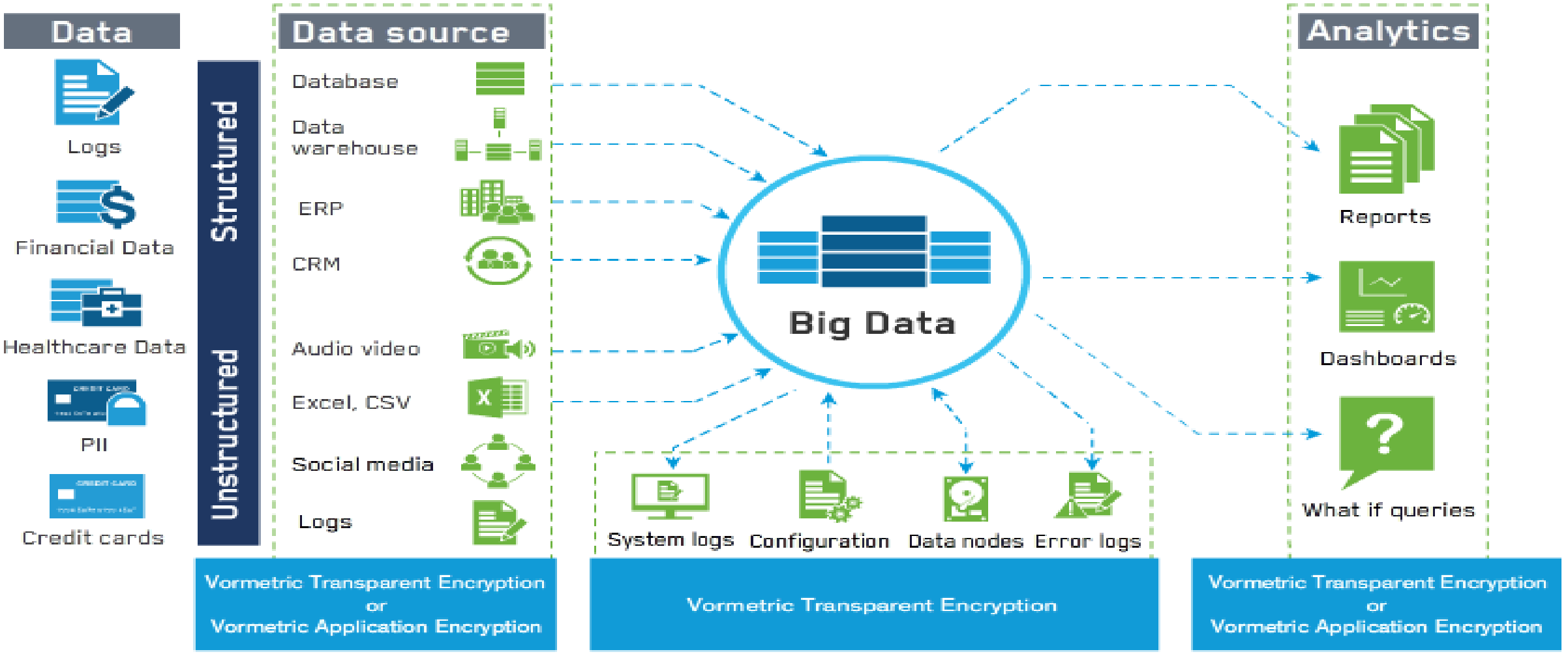


Big Data

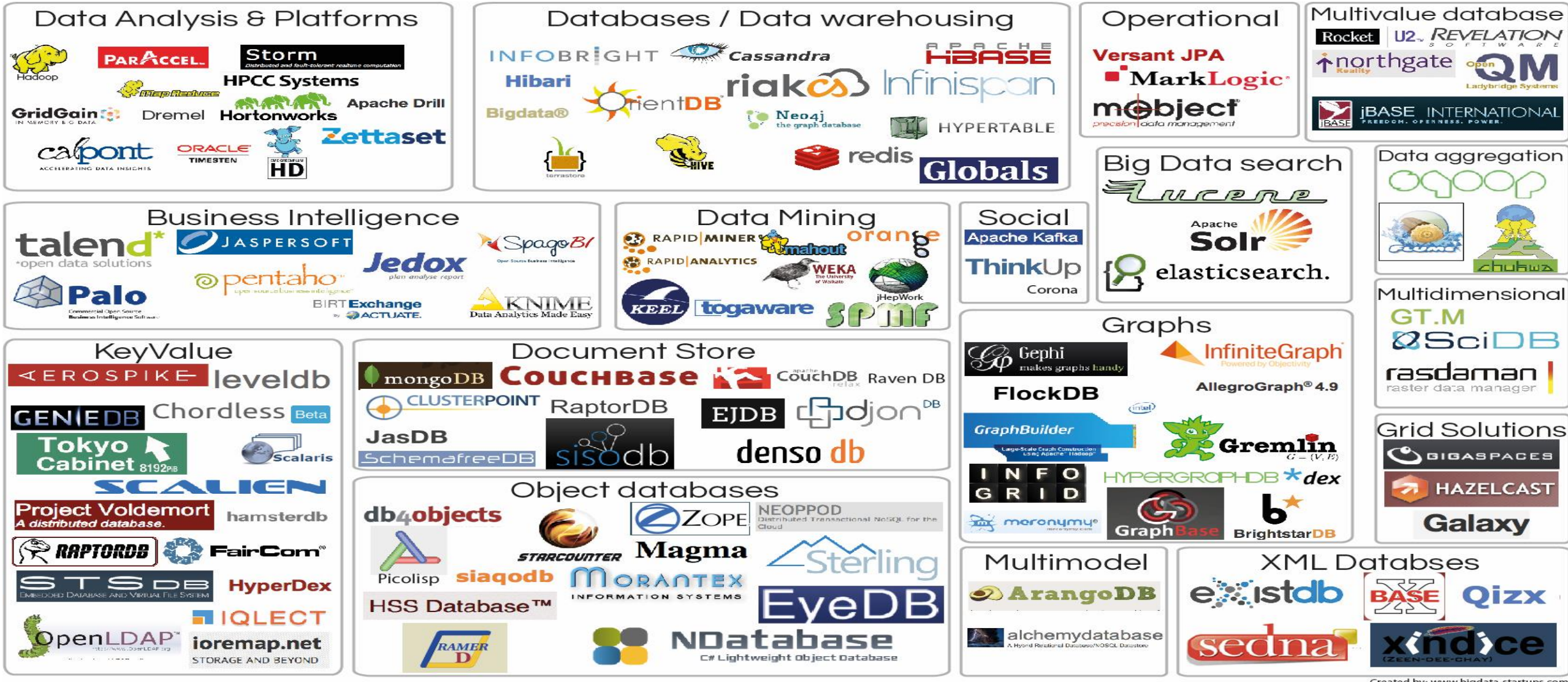
Sistemas Big Data

Sistemas

Data Lake e Data Science



Componentes de Sistemas Big Data



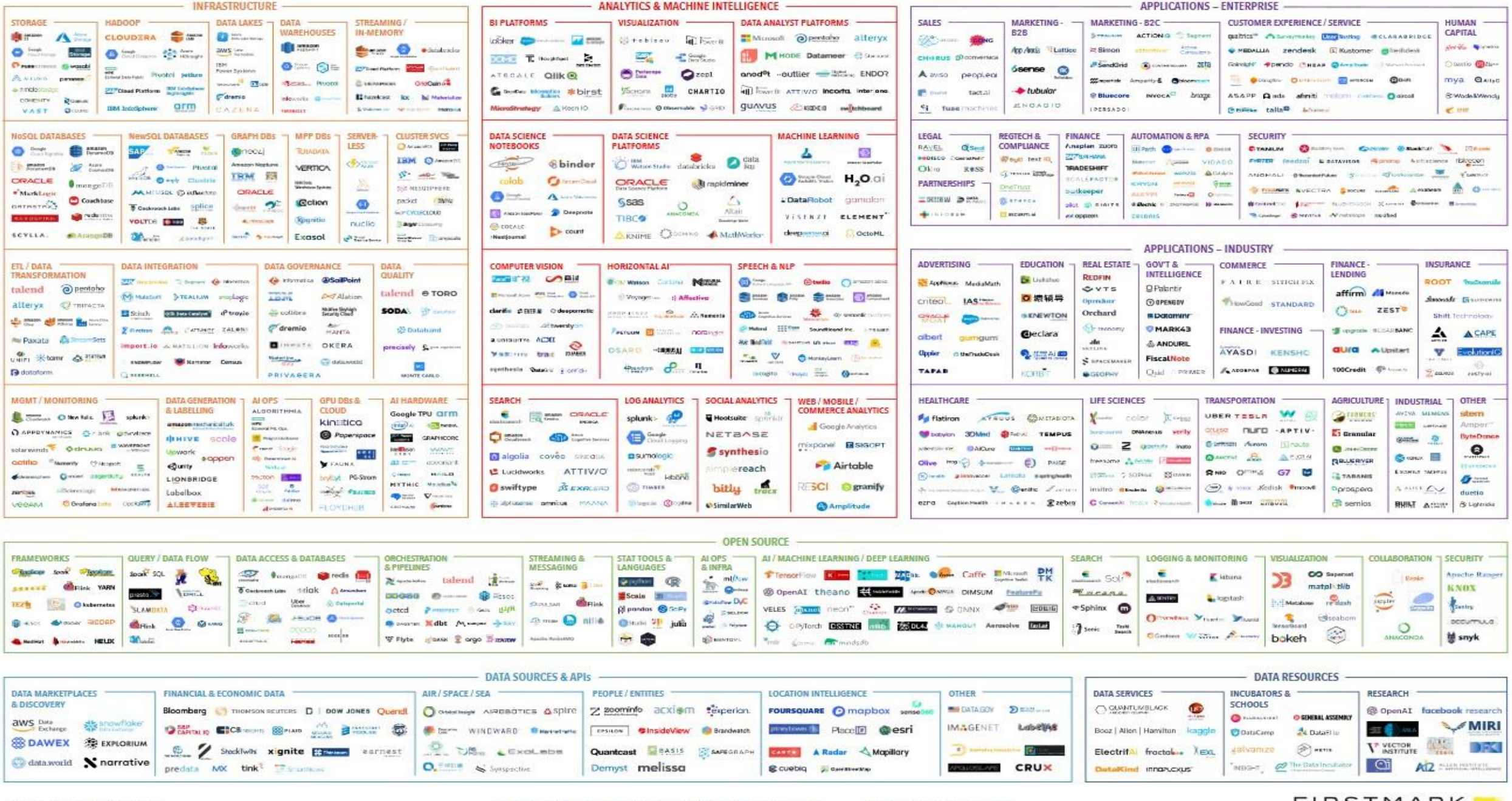
UFPR

Especialização em Inteligência Artificial

Created by: www.bigdata-startups.com

SEPT

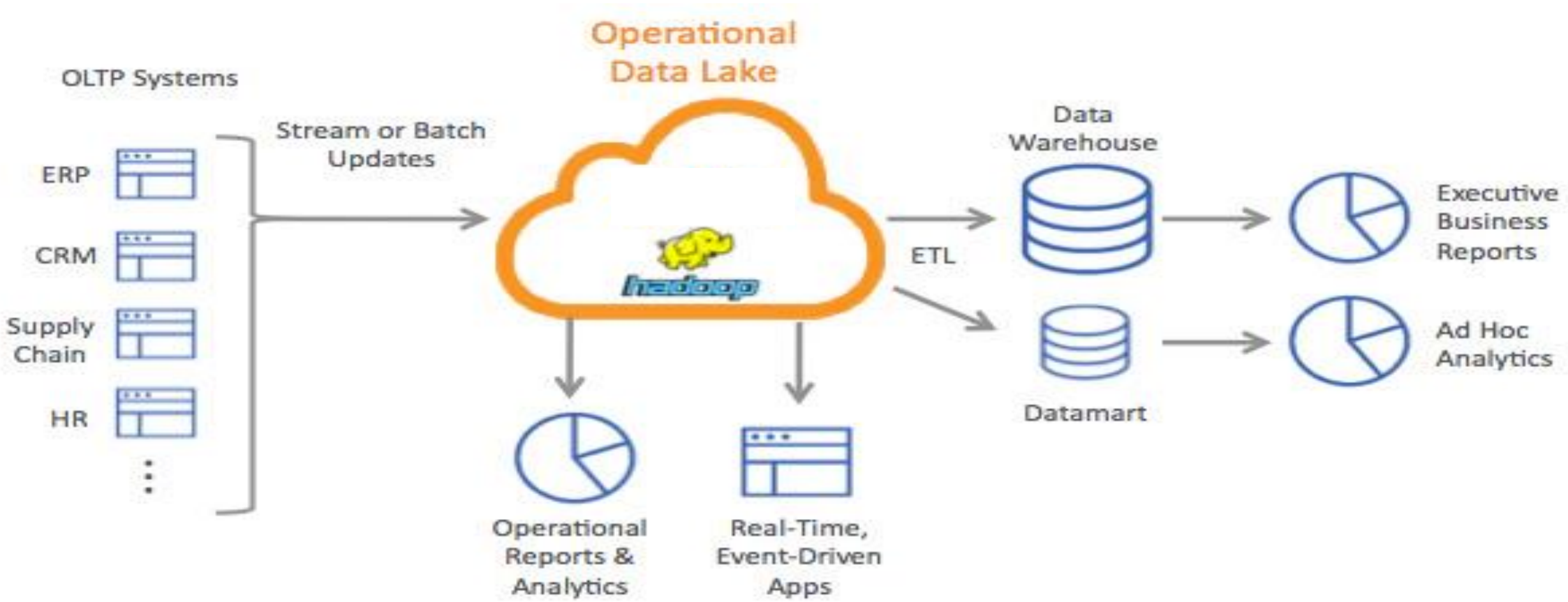
DATA & AI LANDSCAPE 2020



mattturck.com/data2020

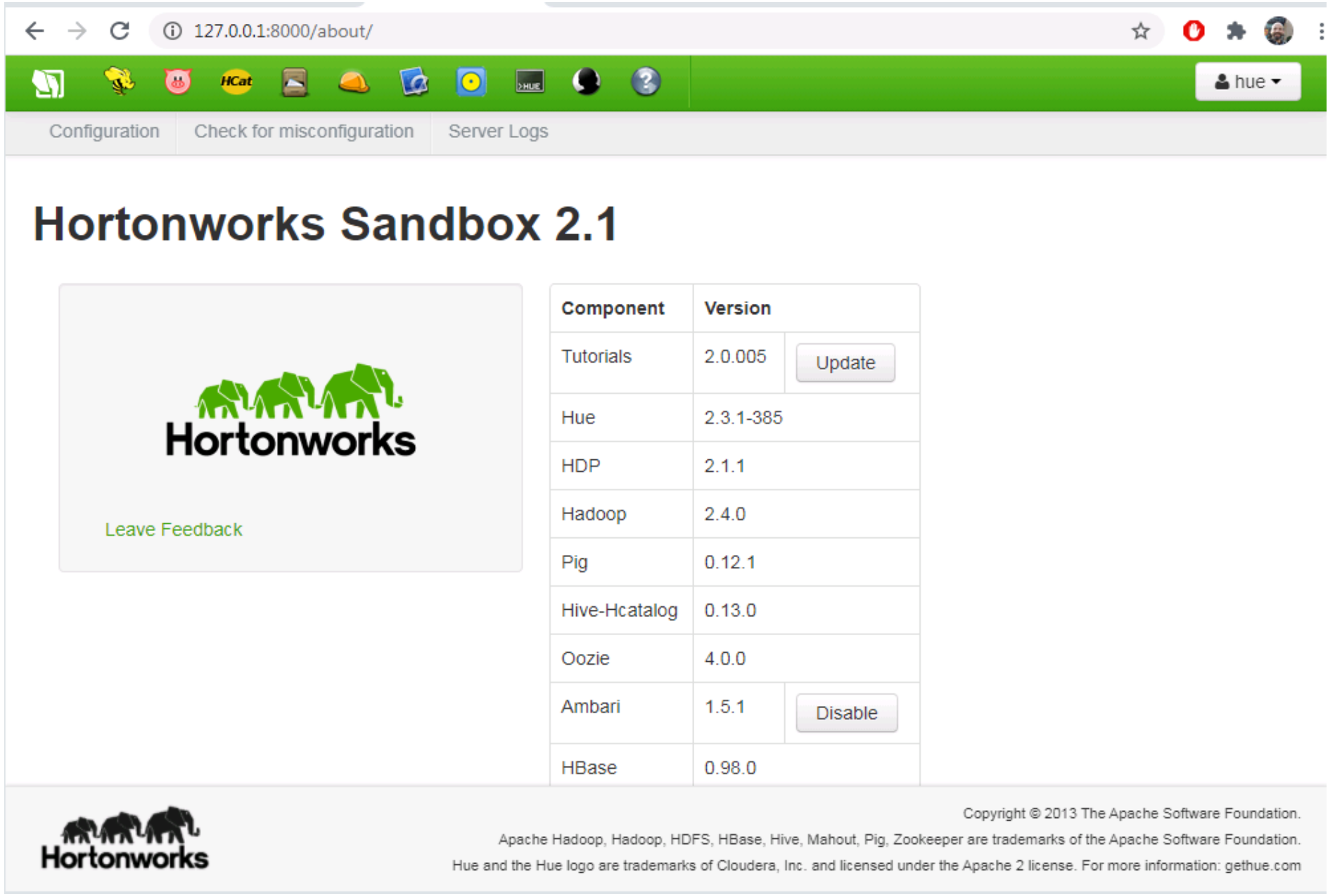
FIRSTMARK
EARLY STAGE VENTURE CAPITAL

Data Lake


OLTP Systems
ERP
CRM
Supply Chain
HR
Stream or Batch Updates
Operational Data Lake
hadoop
ETL
Data Warehouse
Executive Business Reports
Datamart
Ad Hoc Analytics
Operational Reports & Analytics
Real-Time, Event-Driven Apps

Ecosistema Hadoop HDP 2.1

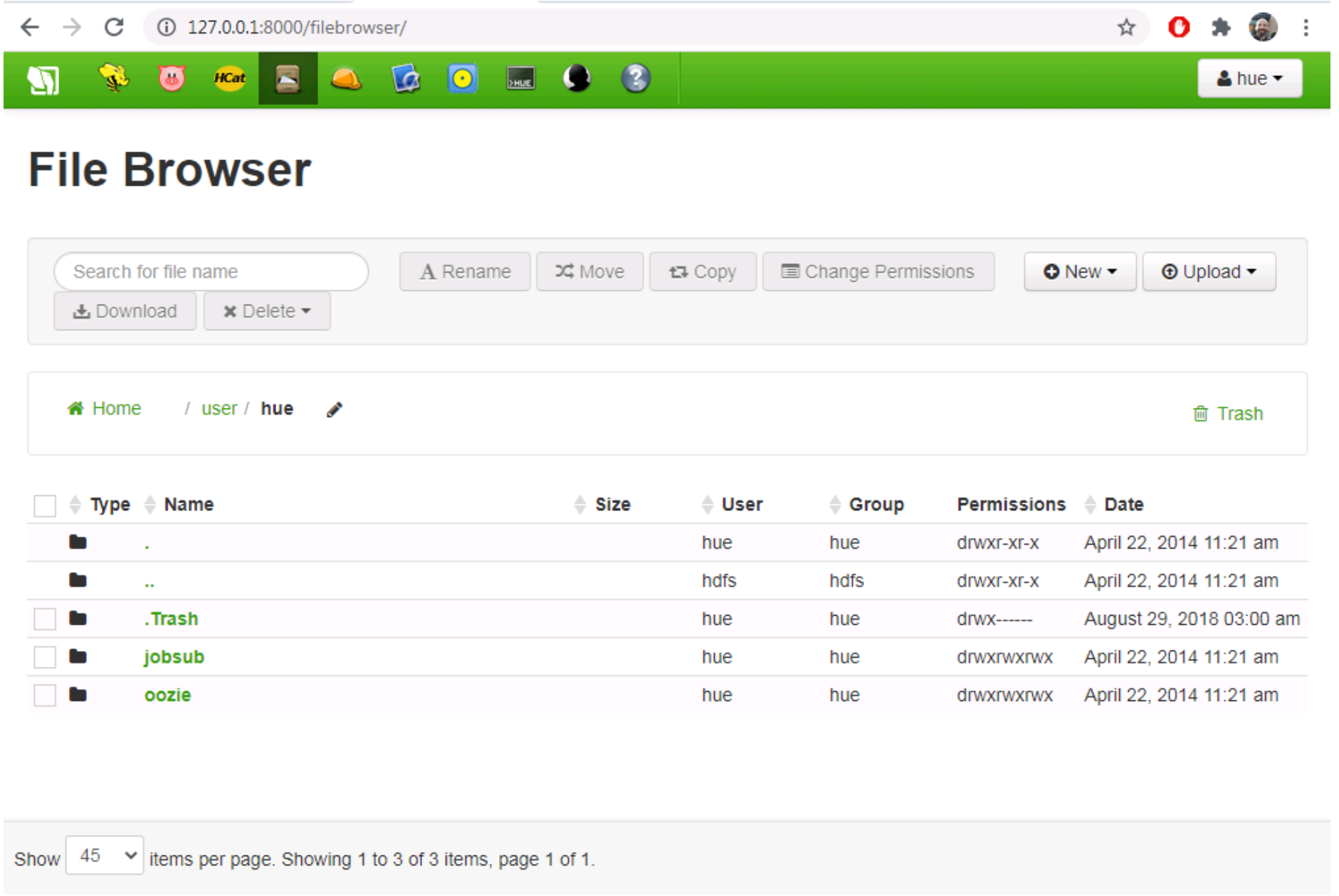
- **Hue – ecosistema Hadoop UI (http://127.0.0.1:8000/)**
- Beeswax – Hive UI (interface BDR → SQL)
- Pig (PigLatin – Bash/SQL)
- Hcatalog – Catálogo de bases de dados
- **Filebrowser – HDFS UI (http://127.0.0.1:8000/filebrowser)**
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- Ambari – Gerênciamento de cluster e aplicações http://<ip>:8080 (admin:admin)
- Hbase – BD orientado a columa
- Knox – Segurança
- Storm – Stream ...



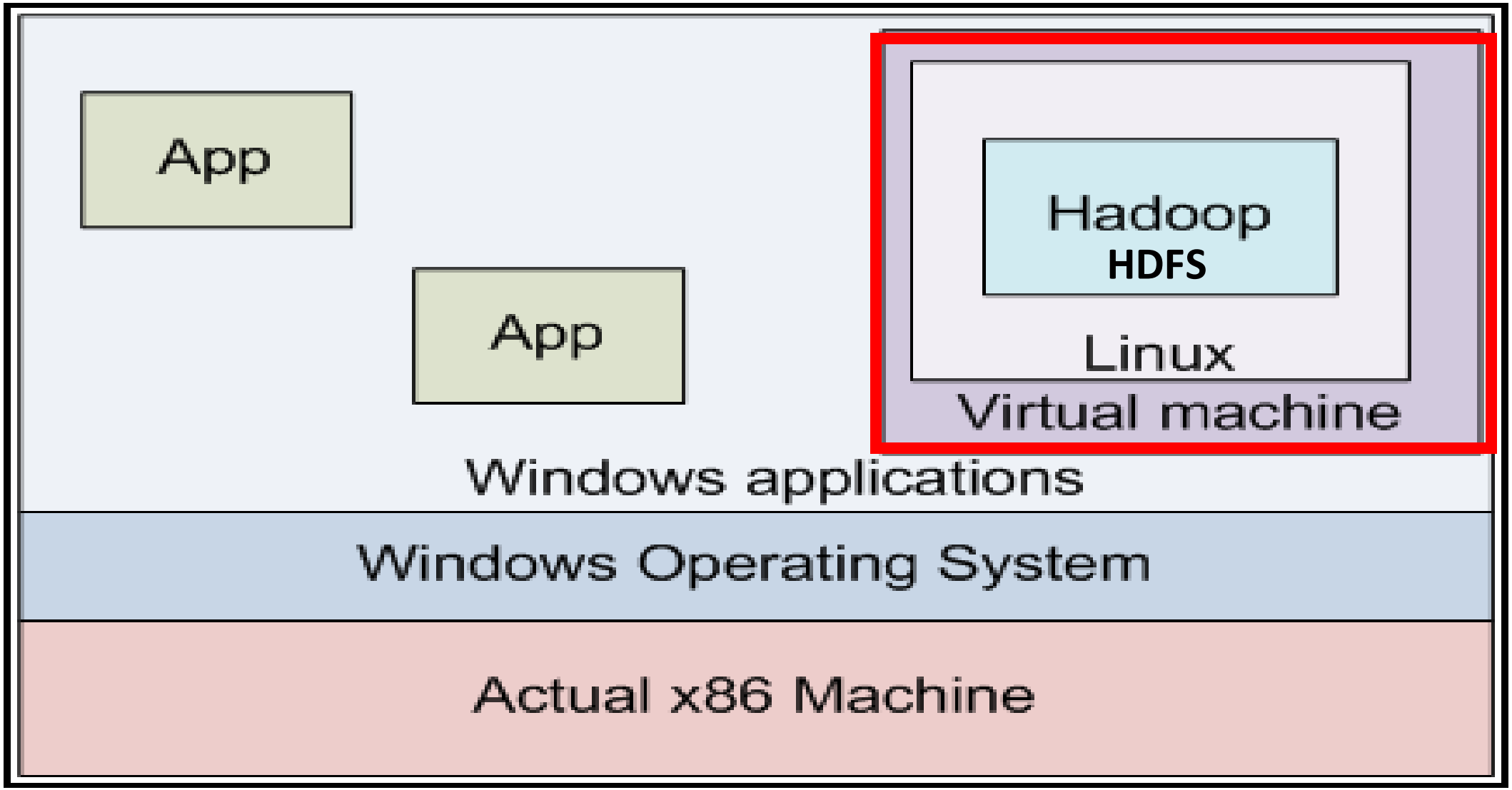
UFPR

Especialização em Inteligência Artificial

SEPT


127.0.0.1:8000/filebrowser/
hue
File Browser
Search for file name
Rename
Move
Copy
Change Permissions
New
Upload
Download
Delete
Home / user / hue
Trash
Type Name Size User Group Permissions Date
. hue hue drwxr-xr-x April 22, 2014 11:21 am
.. hdfs hdfs drwxr-xr-x April 22, 2014 11:21 am
.Trash hue hue drwx------ August 29, 2018 03:00 am
jobsub hue hue drwxrwxrwx April 22, 2014 11:21 am
oozie hue hue drwxrwxrwx April 22, 2014 11:21 am
Show 45 items per page. Showing 1 to 3 of 3 items, page 1 of 1.

Sandbox Hadoop e HDFS



E o MapReduce?

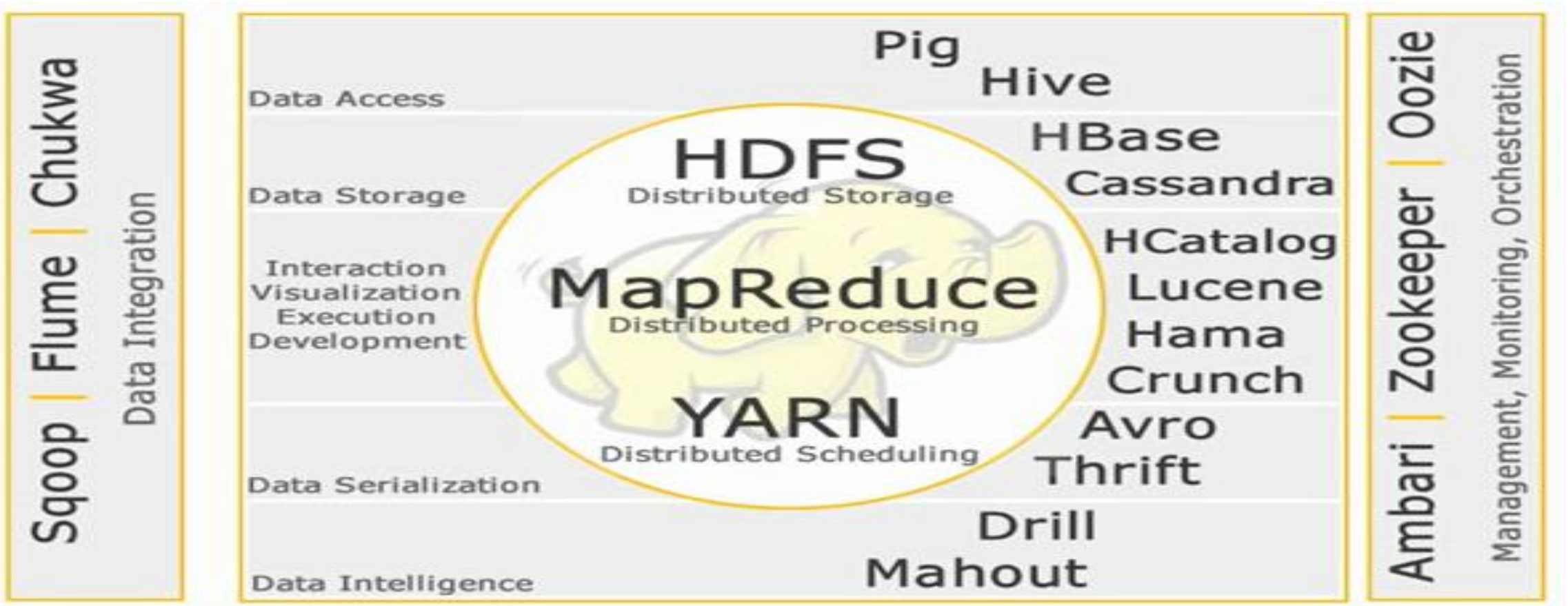
MapReduce (MR) [by Google'04]

- Dean, J., & Ghemawat, S. (2004). **MapReduce: Simplified Data Processing on Large Clusters**. In Proc. of the OSDI - Symp. on Operating Systems Design and Implementation (pp. 137–149). USENIX.
- Modelo de programação e **uma implementação/framework**
  - Programação personalizada
  - Distribuição e Paralelização automática
  - Milhares de máquinas "comuns"
  - Balanceamento de carga
  - Otimização de rede e transferência de arquivos
  - Tolerância à falhas
  - Ambiente de desenvolvimento comum
    - Melhorias beneficiam todos usuários

Hadoop e o Ecosistema Hadoop

[Yahoo et al., 2006]

http://hadoop.apache.org



Modelo de Programação

- Computação baseada em <chave, valor>

- **Map** (<chave, valor>)
  - Saída: um conjunto de pares <chave2, valor2>

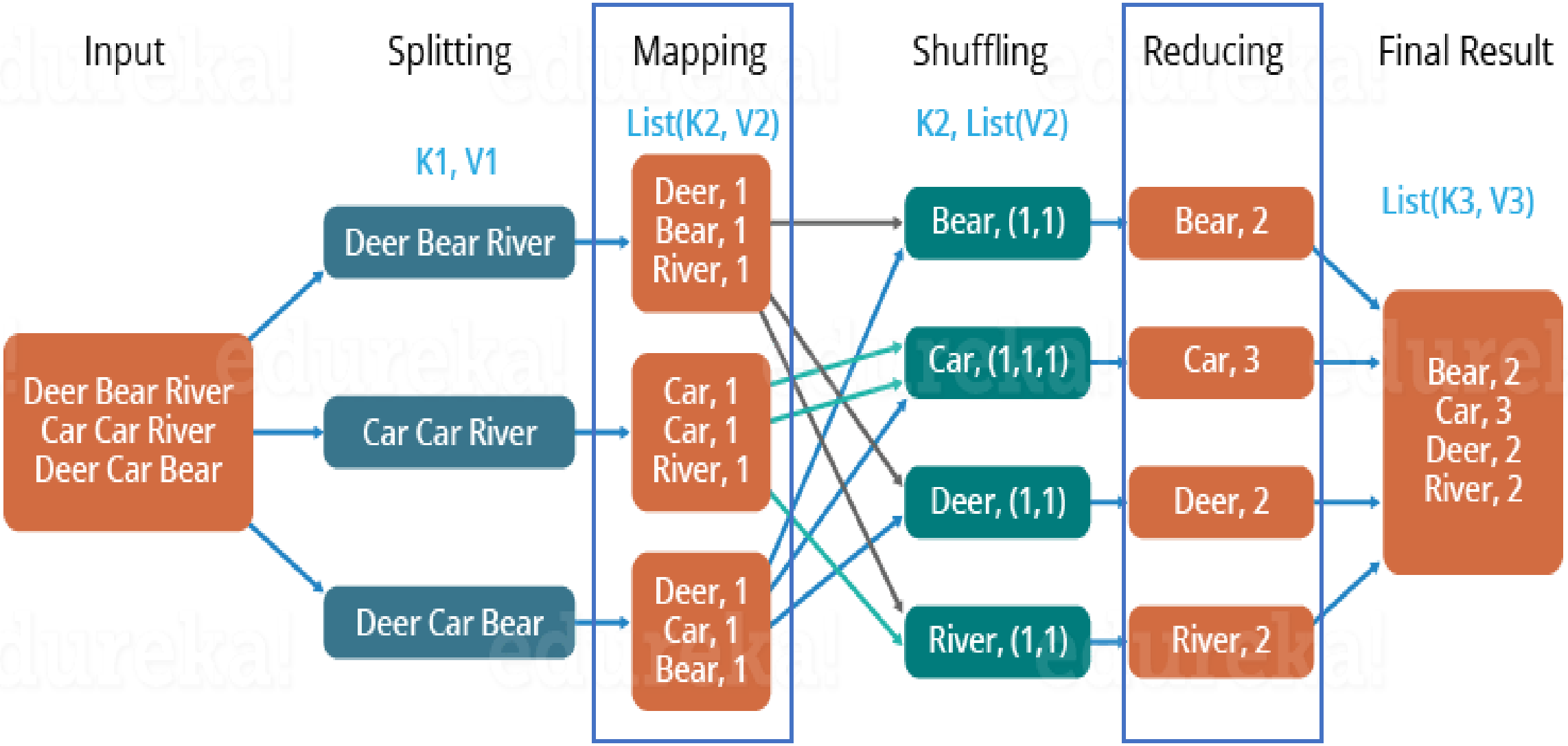
- **Reduce** (<chave2, lista de valor2>)
  - Saída: um par <chave2, valor3>

Problema MR Típico

- Leia a entrada
- **Map**
  - identifique os registros (o dado relevante)
- Combine e Agrupe
- **Reduce**
  - agregue, some, resuma, filtre, transforme,...
- Escreva o resultado

- O esquema permanece o mesmo, altera-se o Map e o Reduce para atender cada problema.

Aplicando Map e Reduce – Contador de Palavras



Algoritmo Map Reduce para Contar Palavras

```
map(String key, String value)
    // key: nome arquivo
    // value: conteúdo do arquivo
    for each word w in value:
        EmitInternediate(w, "1");

reduce(String key, Iterator values)
    // key: uma palavra
    // values: uma lista de contadores
    result=0;
    for each v in values:
        result+= ParseInt(v);
    Emit(key, AsString(result));
```

Executar o Contador de Palavras (wordcount)

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount /mysqld.log /saida

hdfs dfs -cat /saida/part*

hdfs dfs -get /saida /root/

ls -l /root/saida
```

Como implementar e implantar a sua aplicação MR? (hadoop-mapreduce-examples.jar)

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar pi 10 10
```

```
hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount /mysqld.log /saida
```

Implementação e Implantação de uma Aplicação MR

- Gerar o jar com a aplicação MR no desktop*
- Enviar o jar para a sandbox ou cluster Hadoop
  - Winscp, scp ou [filebrowser]! :)
- Executar na sandbox ou cluster

```
hadoop jar seu.jar /mysqld.log /saída
```

* Seu ambiente de desenvolvimento (Sandbox) não tem JDK e fonte do Hadoop

Gerar Jar

- Versão hadoop na sandbox? hadoop version
- https://archive.apache.org/dist/hadoop/common/
- hadoop[version].tar.gz (src não tem os jar)
- Versão Jdk na sandbox? java –version (para penúltimo passo)
- Netbeans, new project, java application
- Java: https://tinyurl.com/AulaWC-java (baseado em http://tinyurl.com/htjlklc)
  - Adequar o pacote (package)
  - Renomear arquivo para ficar igual a classe
- Adicionar bibliotecas (Project, Properties, Library, Add JAR)
  share/hadoop/common/hadoop-common
  share/hadoop/mapreduce/hadoop-mapreduce-client-core
- Project properties, sources, source/binary format (jdk?)
- Netbeans, run, build project

Observações

- Mesmo nome do projeto e pacote
  - erro de invocação do main ou classe não encontrada
- Lembrar dos três ambientes
  - Desktop (tua máquina)
  - VM (ou servidor Hadoop externo)
  - HDFS
- Execução a partir do arquivo jar da VM (Linux), e não do HDFS
- Ambiente de execução local/stand-alone (junto com IDE)

Programa

- Fundamentos de Big Data
  - Big Data, Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - **Personalização de aplicações Map Reduce**
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
- NoSQL e NewSQL
- Dados em movimento – Processamento de Streaming

Observações Avançadas

- Execução de aplicações em outra linguagens
  - $ hadoop jar /usr/lib/hadoop-mapreduce/**hadoop-streaming.jar -mapper** /mapper.py **-reducer** /reducer.py **-input** /mysqld.log **-output** /saida-py

  https://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/

- Entrada da aplicação MR personalizada (INPUT e blocos HDFS)
  - Divisão de palavra/registro/objeto/tag/marcação/...

WordCount main()

```
public static void main(String[] args) throws Exception {
      JobConf conf = new JobConf(AulaWC.class);
      conf.setJobName("wordcount");

      conf.setInputFormat(TextInputFormat.class);
      conf.setOutputFormat(TextOutputFormat.class);

      conf.setOutputKeyClass(Text.class);
      conf.setOutputValueClass(IntWritable.class);

      conf.setMapperClass(Map.class);
      conf.setCombinerClass(Reduce.class);
      conf.setReducerClass(Reduce.class);

      FileInputFormat.setInputPaths(conf, new Path(args[0]));
      FileOutputFormat.setOutputPath(conf, new Path(args[1]));

      JobClient.runJob(conf);
    }
```

Programa

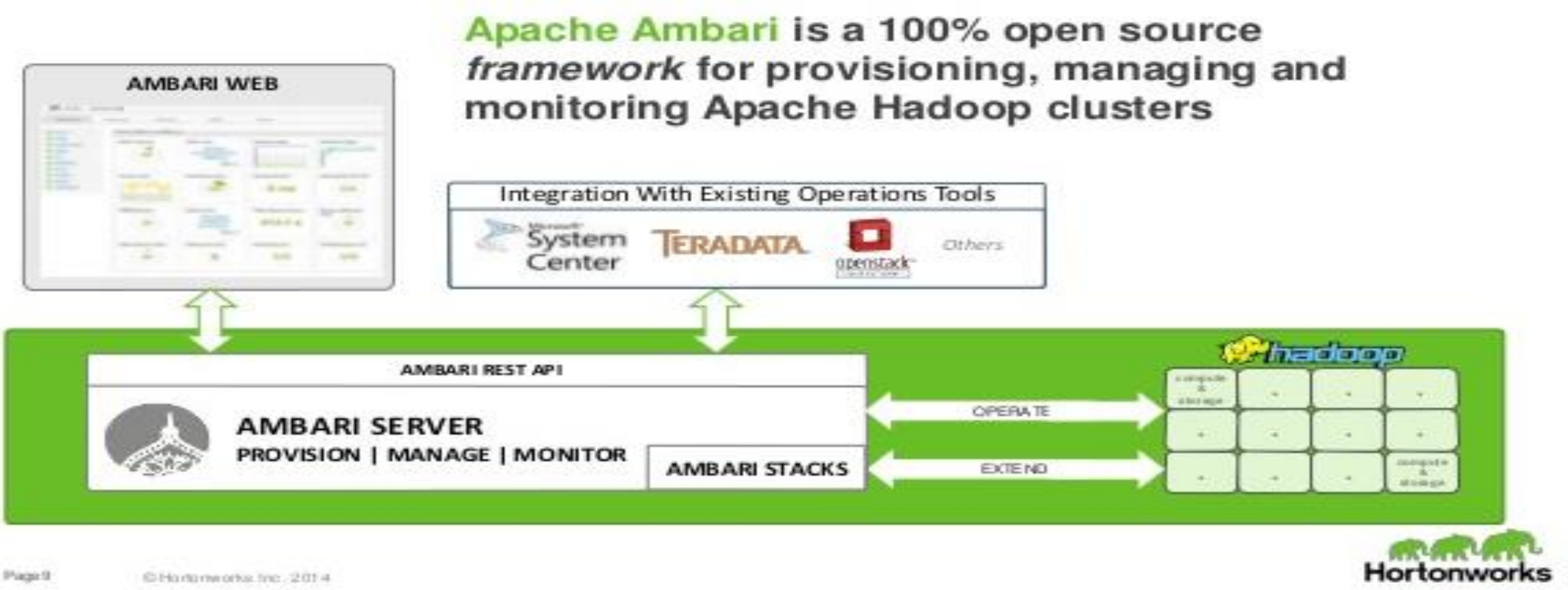
- Fundamentos de Big Data
  - Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - Personalização de aplicações Map Reduce
- **Data Engineering (Gerenciamento/Ferramentas para Big Data)**
  - NoSQL e NewSQL
  - Dados em movimento – Processamento de Streaming

Ecosistema Hadoop HDP 2.1

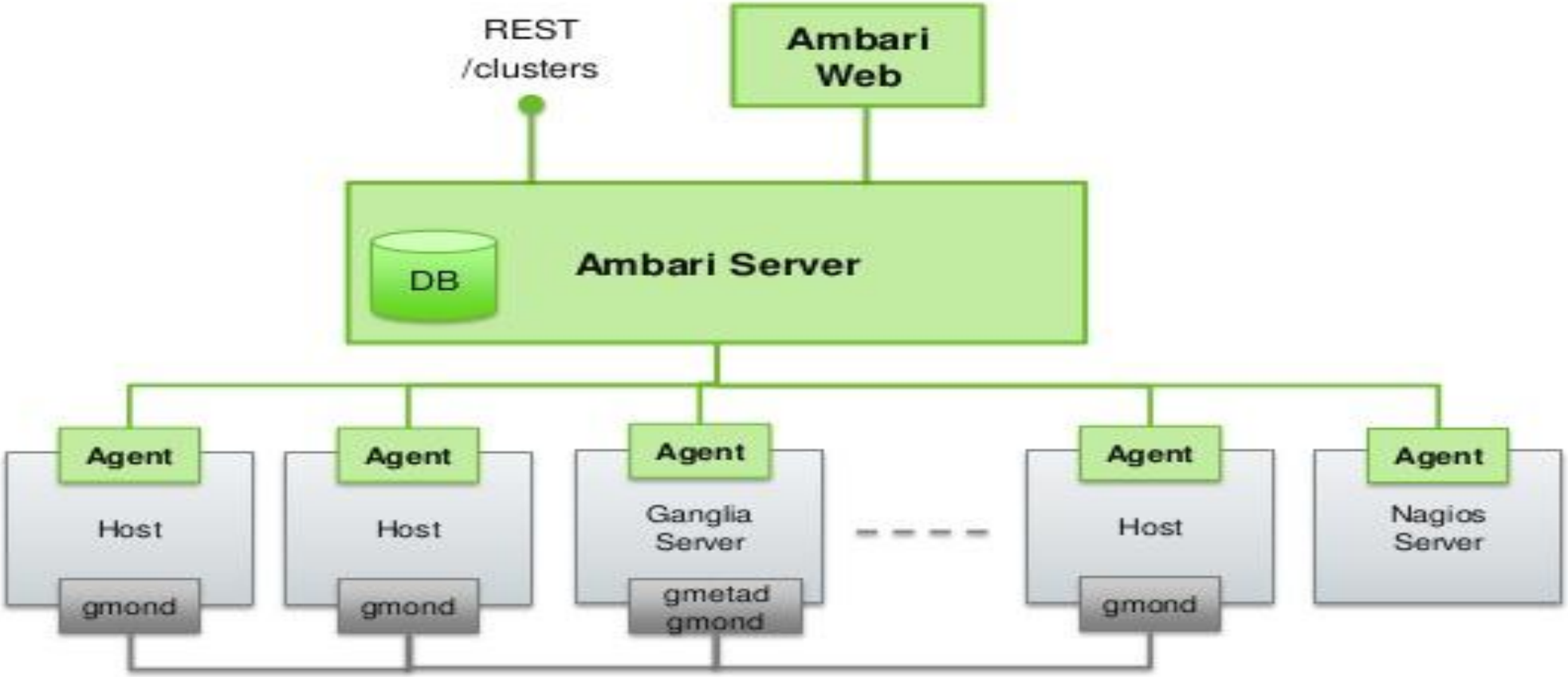
- Hue – ecosistema Hadoop UI
- Beeswax – Hive UI (interface BDR → SQL)
- Pig (PigLatin – Bash/SQL)
- Hcatalog – Catálogo de bases de dados
- Filebrowser – HDFS UI
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- **Ambari – Gerenciamento de cluster e aplicações**
- Hbase – BD orientado a columa
- Knox – Segurança
- Storm – Stream ...

Apache Ambari

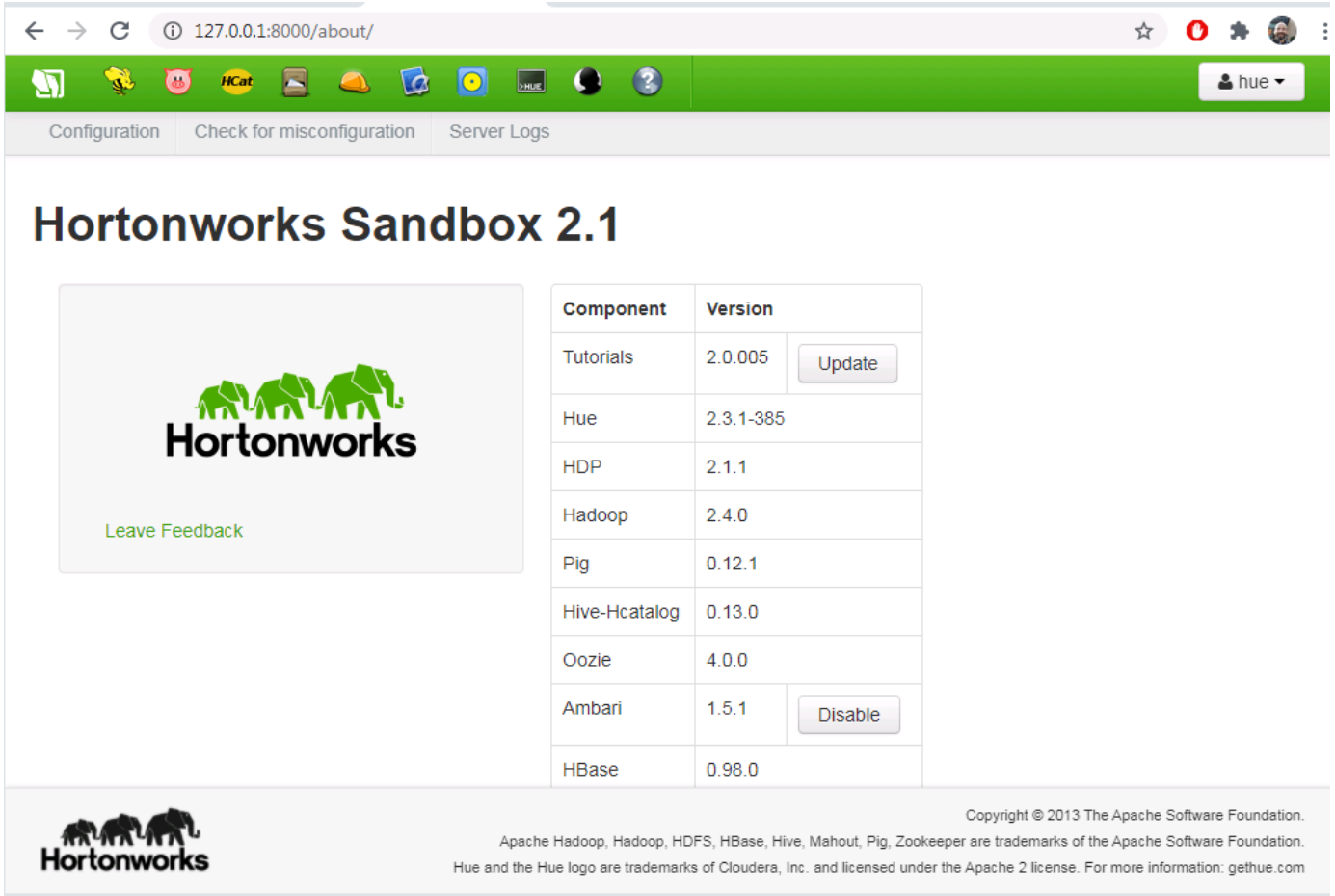
- https://ambari.apache.org



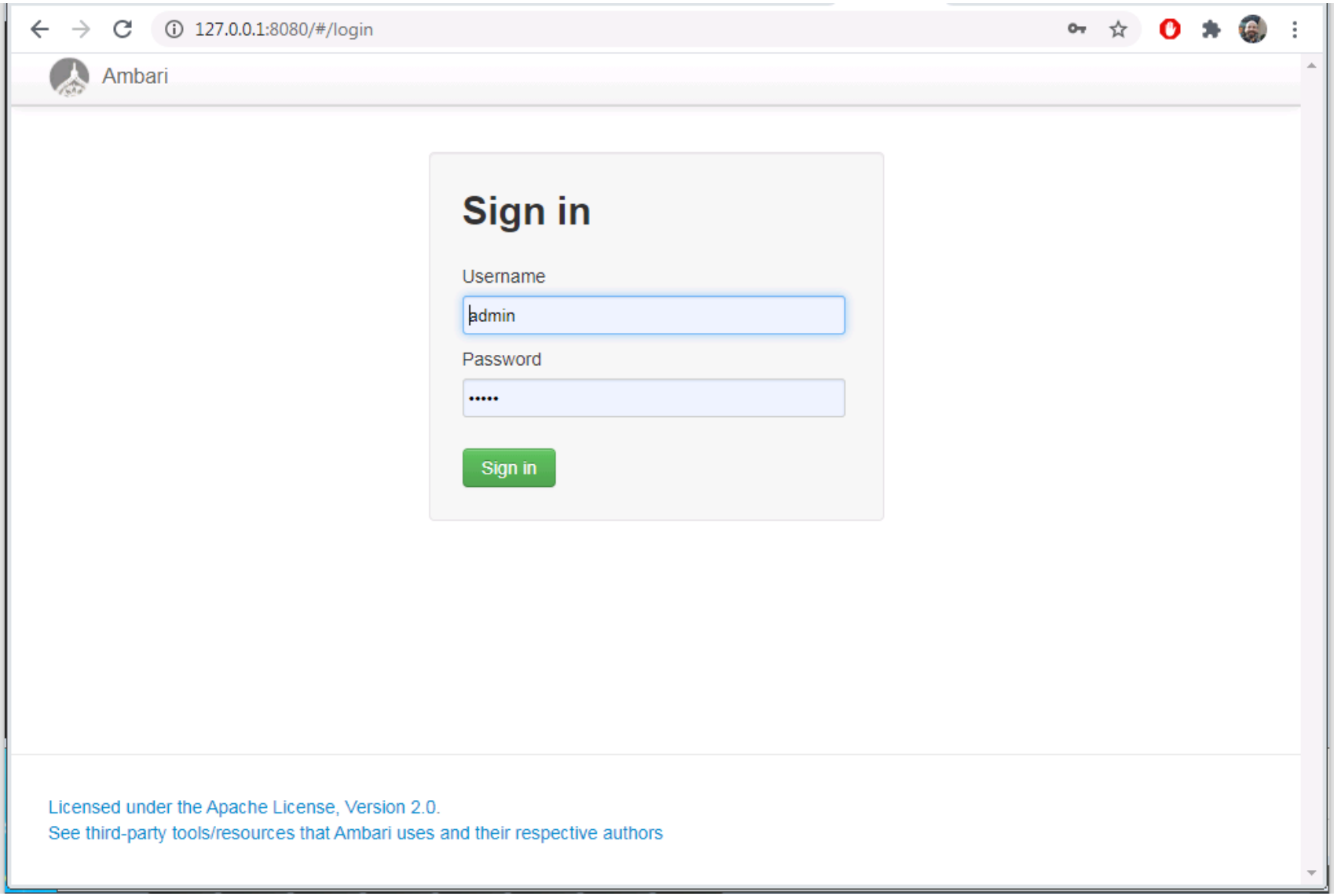
Ambari System Architecture

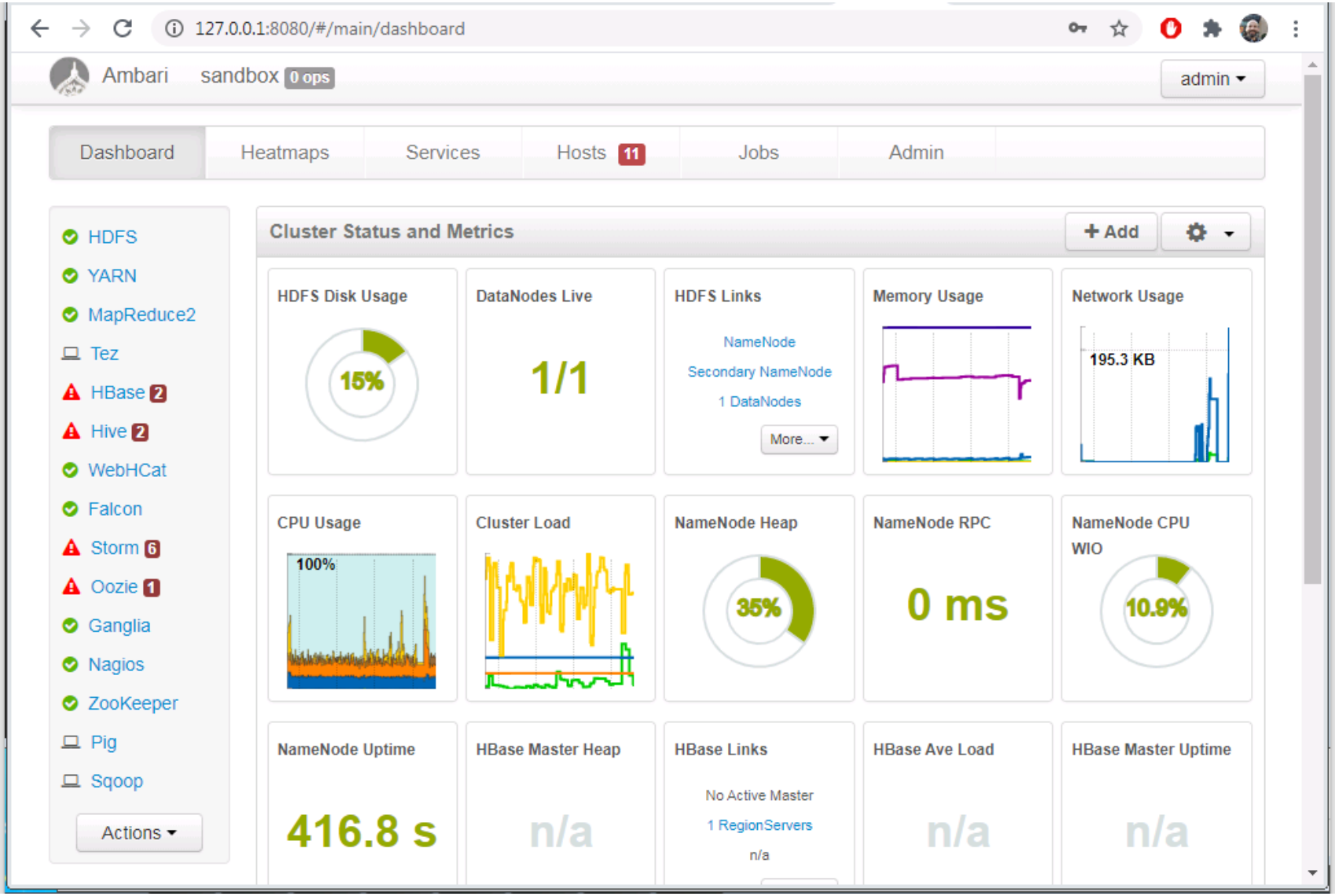


- Start Ambari via HUE
  - http://127.0.0.1:8000 (Ambari, Enable)



http://<ip>:8080 (admin:admin)



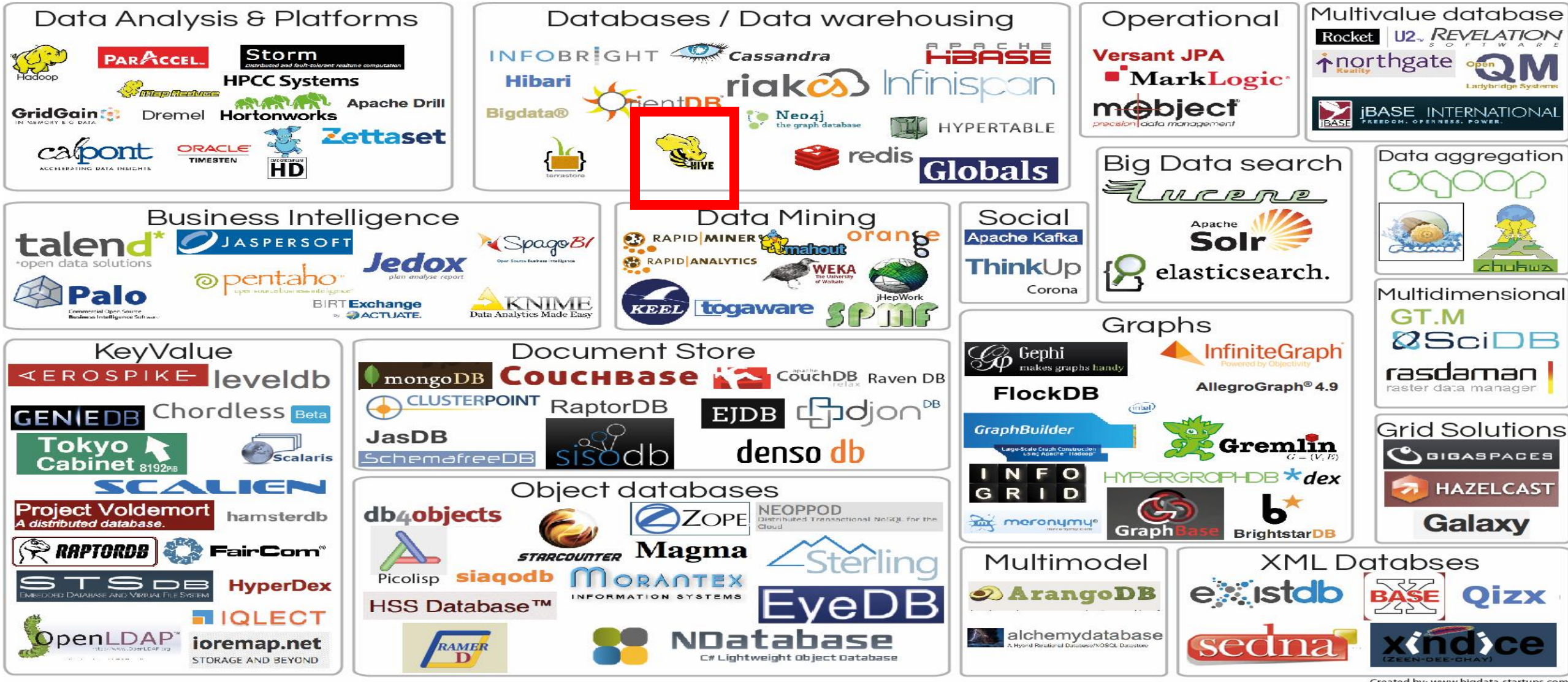

127.0.0.1:8080/#/main/dashboard
Ambari
sandbox
0 ops
admin
Dashboard
Heatmaps
Services
Hosts 11
Jobs
Admin
HDFS
YARN
MapReduce2
Tez
HBase 2
Hive 2
WebHCat
Falcon
Storm 6
Oozie 1
Ganglia
Nagios
ZooKeeper
Pig
Sqoop
Actions
Cluster Status and Metrics
+ Add
HDFS Disk Usage
15%
DataNodes Live
1/1
HDFS Links
NameNode
Secondary NameNode
1 DataNodes
More...
Memory Usage
Network Usage
195.3 KB
CPU Usage
100%
Cluster Load
NameNode Heap
35%
NameNode RPC
0 ms
NameNode CPU WIO
10.9%
NameNode Uptime
416.8 s
HBase Master Heap
n/a
HBase Links
No Active Master
1 RegionServers
n/a
HBase Ave Load
n/a
HBase Master Uptime
n/a

- Apresentar menu geral
- Services: Iniciar, Parar, Reiniciar Serviços
- Hosts: Adicionar e gerenciar componentes (ecossistema Hadoop) em máquinas,... Clicando em uma detalhar os componentes.
- Usaremos mais ao longo do curso

Ecosistema Hadoop HDP 2.1

- Hue – ecosistema Hadoop UI
- **Beeswax – Hive UI (interface BDR → SQL)**
- Pig (PigLatin – Bash/SQL)
- Hcatalog – Catálogo de bases de dados
- Filebrowser – HDFS UI
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- Ambari – Gerência do cluster e aplicações http://<ip>:8080 (admin:admin)
- Hbase – BD orientado a coluna
- Knox – Segurança
- Storm – Stream ...

Componentes de Sistemas Big Data



Created by: www.bigdata-startups.com

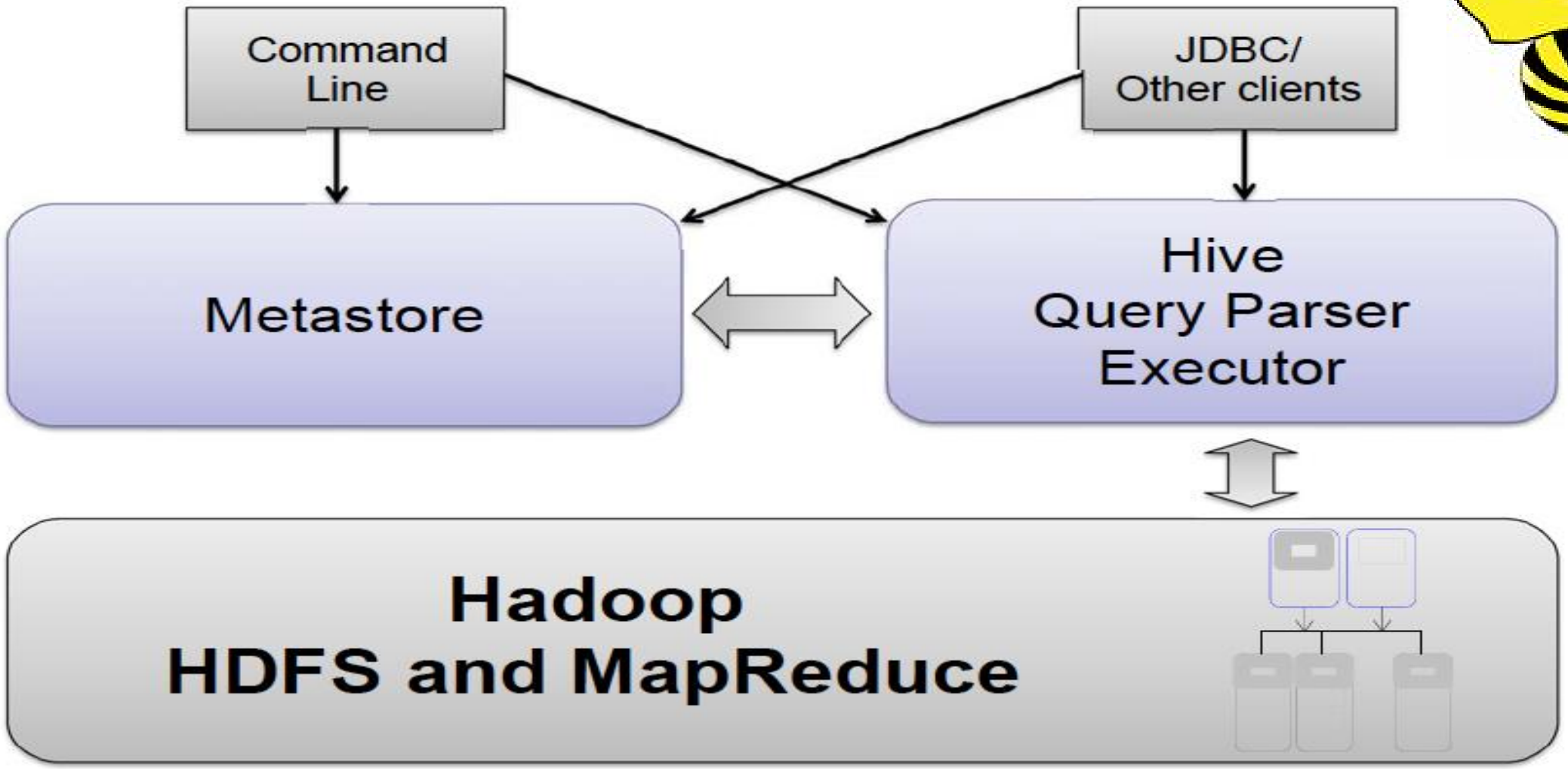
UFPR

Especialização em Inteligência Artificial

SEPT

http://hive.apache.org

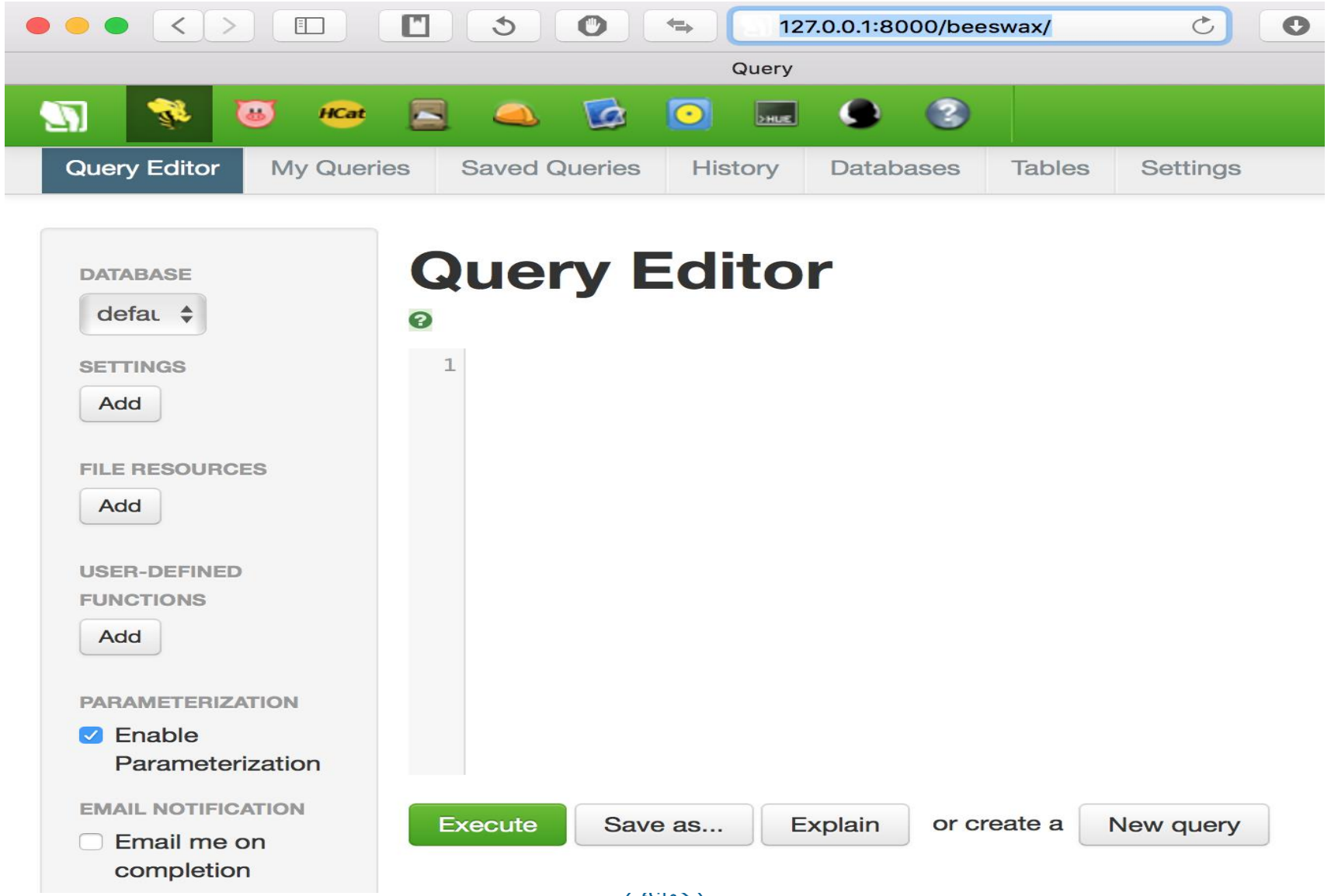
Hive: A Petabyte Scale Data Warehouse Using Hadoop



Hive

- Thusoo, A., Sarma, J. Sen, Jain, N., Shao, Z., Chakka, P., Zhang, N., … Murthy, R. (2010). Hive - A Petabyte Scale Data Warehouse Using Hadoop. Proc. of the ICDE – Intl. Conf. on Data Engineering
- Facebook
- **Hive** (Framework) and **HiveQL**(SQL-like)
  - HiveQL to MapReduce jobs
- Custom MapReduce scripts
- **Data Model: Database, Table, Column, Row and Partition**
- Text files and SequenceFiles (binary key/value)
- Data summarization, query and analysis
- Not OLTP, no real-time queries, no row-level updates

Beeswax (HIVE Interface)



- Apresentar menu
- Tabelas sample_07 e sample_08 (code, description, salary, total_emp)
- Fazer consultas básicas: COUNT, SUM, MAX, MIN, AVG, …
  - LOG sem e com submissão de tarefas mapreduce
- My queries (SORT BY [DESC], JOIN, GROUP BY)…
- Tables, Create a new manually… from a file... (teste from /saida/part-r-00000) **Warning... move file...**

SELECT col_1, col_2 FROM teste SORT BY col_2 DESC LIMIT 1

- String x Int para col_2

- Onde ficam as tabelas/arquivos?
- CREATE TABLE aula AS SELECT ...;

Atividade 5 – Executar exemplos HIVE

- Enviar um arquivo PDF com a execução de duas consultas conforme a seguir, ou outras do seu interesse

  - listar os 10 maiores salários de 2008 (sample_08);

```
SELECT salary FROM sample_08 SORT BY salary DESC LIMIT 10;
```

  - listar as palavras que apareceram mais que 5 vezes do arquivo importado a partir da saída do wordcount (/saida/part-r-0000)

```
SELECT col_1, col_2 FROM teste WHERE col_2 > 5;
```

Hive via comando

- ***Cria arquivo exemplo

```
cat > /root/user-posts.txt
user1,Funny Story,1343182026191
user2,Cool Deal,1343182133839
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
<crt+c>
```

- Execução via comando

hive

```
Hive history
file=/tmp/hadoop/hive_job_log_hadoop_201207312052_
1402761030.txt
hive>
```

```
hive> CREATE TABLE posts (user STRING, post STRING, time BIGINT)
      ROW FORMAT DELIMITED
      FIELDS TERMINATED BY ','
      STORED AS TEXTFILE;
OK
Time taken: 10.606 seconds

hive> show tables;
OK
posts
Time taken: 0.221 seconds

hive> describe posts;
OK
user string
post string
time bigint
Time taken: 0.212 seconds

hive> describe formatted posts;
Hive> show CREATE TABLE posts;
```

```
hive> !cat /root/user-posts.txt;
user1,Funny Story,1343182026191
user2,Cool Deal,1343182133839
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
hive> LOAD DATA LOCAL INPATH '/root/user-posts.txt'
      OVERWRITE INTO TABLE posts;
Copying data from file:/root/user-posts.txt
Copying file: file:/root/user-posts.txt
Loading data to table default.posts
Moved to trash: hdfs://sandbox:8020/apps/hive/warehouse/posts
Table default.posts stats: [num_partitions: 0, num_files: 1,
num_rows: 0, total_size: 135, raw_data_size: 0]
OK
Time taken: 1.795 seconds
Hive> quit

$ hdfs dfs -cat /apps/hive/warehouse/posts/user-posts.txt
user1,Funny Story,1343182026191
user2,Cool Deal,1343182133839
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
```

Loading Data

- HDFS COPY (Linux)
  - `hive> LOAD DATA LOCAL INPATH '/saida/part-r-00000'`
    `OVERWRITE INTO TABLE teste;`
- HDFS MOVE (HDFS)
  - `hive> LOAD DATA INPATH '/saida/part-r-00000'`
    `OVERWRITE INTO TABLE teste;`
  - O arquivo é movido para /apps/hive/warehouse/
- HDFS Link
  - `hive> CREATE EXTERNAL TABLE teste2(col_1 STRING, col_2 INT)`
    `ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'`
    `STORED AS TEXTFILE LOCATION '/saida/';`
  - Apenas aponta para o arquivo no HDFS

Schema Violation (Insert Incompatible Data)

```
hive> !cat /root/user-posts-inconsistentFormat.txt;
user1,Funny Story,1343182026191
user2,Cool Deal,2012-01-05
user4,Interesting Post,1343182154633
user5,Yet Another Blog,13431839394
hive> describe posts;
OK
user string
post string
time bigint
Time taken: 0.289 seconds
hive> LOAD DATA LOCAL INPATH
    > '/root/user-posts-inconsistentFormat.txt'
    > OVERWRITE INTO TABLE posts;
OK
Time taken: 0.612 seconds
hive> select * from posts;
OK
user1 Funny Story 1343182026191
user2 Cool Deal NULL
user4 Interesting Post 1343182154633
user5 Yet Another Blog 13431839394
Time taken: 0.136 seconds
```

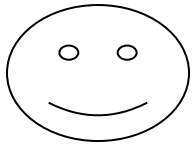
Insert Data

- Insert data into Hive tables from queries

```
hive> INSERT INTO TABLE teste2
    > SELECT * FROM teste;

hive> INSERT INTO TABLE teste2
    > SELECT ‘joao’, 134
    > FROM teste LIMIT 1;


hive> cat /apps/hive/warehouse/teste2/part-r-00000
```

Drop Table

```
hive> DROP TABLE posts;
OK
Time taken: 2.182 seconds

hive> exit;


$ hdfs dfs -ls /apps/hive/warehouse/
$
```



Create Partitioned Table

```
hive> CREATE TABLE posts (user STRING, post STRING, time BIGINT)
      PARTITIONED BY(country STRING)
      ROW FORMAT DELIMITED
      FIELDS TERMINATED BY ','
      STORED AS TEXTFILE;
hive> LOAD DATA LOCAL INPATH 'data/user-posts-US.txt’
 OVERWRITE INTO TABLE posts PARTITION(country='US');
hive> LOAD DATA LOCAL INPATH 'data/user-posts-AUSTRALIA.txt’
 OVERWRITE INTO TABLE posts PARTITION(country='AUSTRALIA');
$ hdfs dfs -ls -R /user/hive/warehouse/posts
/user/hive/warehouse/posts/country=AUSTRALIA
/user/hive/warehouse/posts/country=AUSTRALIA/user-posts-
AUSTRALIA.txt
/user/hive/warehouse/posts/country=US
/user/hive/warehouse/posts/country=US/user-posts-US.txt
```

Load Data into Partitioned Table

```
hive> LOAD DATA LOCAL INPATH 'data/user-posts-US.txt'
> OVERWRITE INTO TABLE posts;
FAILED: Error in semantic analysis: Need to specify partition
columns because the destination table is partitioned

hive> LOAD DATA LOCAL INPATH 'data/user-posts-US.txt'
> OVERWRITE INTO TABLE posts PARTITION(country='US');
OK
Time taken: 0.225 seconds

hive> LOAD DATA LOCAL INPATH 'data/user-posts-AUSTRALIA.txt'
> OVERWRITE INTO TABLE posts PARTITION(country='AUSTRALIA');
OK
Time taken: 0.236 seconds
hive>
```

Partitioned Table

```
hive> show partitions posts;
OK
country=AUSTRALIA
country=US
Time taken: 0.095 seconds
hive> exit;
```

```
$ hdfs dfs -ls -R /user/hive/warehouse/posts
/user/hive/warehouse/posts/country=AUSTRALIA
/user/hive/warehouse/posts/country=AUSTRALIA/user-posts-
AUSTRALIA.txt
/user/hive/warehouse/posts/country=US
/user/hive/warehouse/posts/country=US/user-posts-US.txt
```

```
hive> select * from posts limit 4;
user1 Funny Story 1343182026191
user2 Cool Deal 1343182133839
user4 Interesting Post 1343182154633
user5 Yet Another Blog 1343183939434
hive> select * from likes limit 4;
user1 12 1343182026191
user2 7 1343182139394
user3 0 1343182154633
user4 50 1343182147364

hive> SELECT p.user, p.post, l.count
      FROM posts p JOIN likes l ON (p.user = l.user);
user1 Funny Story 12
user2 Cool Deal 7
user4 Interesting Post 50

hive> SELECT p.user, p.post, l.count
      FROM posts p LEFT OUTER JOIN likes l ON (p.user = l.user);
```

Regular Expression Deserializer (Apache Weblog Data)

```
CREATE EXTERNAL TABLE IF NOT EXISTS apachelog( host STRING, identity
STRING, user STRING, time STRING, request STRING, status STRING, size
STRING, referer STRING, agent STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe'
WITH SERDEPROPERTIES(
'input.regex' = '([^ ]*) ([^ ]*) ([^ ]*) (-|\\[[^\\]]*\\])
([^\"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-9]*)(?: ([^ \"]*|\"[^\"]*\") ([^
\"]*|\"[^\"]*\"))?', 'output.format.string' = '%1$s %2$s %3$s %4$s %5$s
%6$s %7$s %8$s %9$s')
LOCATION "../apache.access.log";

SELECT * FROM apachelog WHERE host = '200.192.112.33'
SELECT * FROM apachelog ORDER BY time;
```

```
hive> select count (1) from teste;
Total MapReduce jobs = 1
Launching Job 1 out of 1
...
Starting Job = job_1343957512459_0004, Tracking URL =
http://localhost:8088/proxy/application_1343957512459_0004/
Kill Command = hadoop job -Dmapred.job.tracker=localhost:10040 -kill
job_1343957512459_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers:
2012-08-02 22:37:24,962 Stage-1 map = 0%, reduce = 0%
2012-08-02 22:37:32,664 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 2.6
MapReduce Total cumulative CPU time: 2 seconds 640 msec
Ended Job = job_1343957512459_0004
MapReduce Jobs Launched:
Job 0: Map: 1 Reduce: 1 Accumulative CPU: 2.64 sec HDFS Read: 0 HDFS Write: 0
SUCESS
Total MapReduce CPU Time Spent: 2 seconds 640 msec
OK
4
Time taken: 14.204 seconds
```

Hive Refs

- Hadoop: The Definitive Guide. Tom White.
- Hive. Edward Capriolo, Dean Wampler, Jason Rutherglen
- http://hive.apache.org/
- https://cwiki.apache.org/confluence/display/Hive/Home
- https://cwiki.apache.org/confluence/display/Hive/GettingStarted
- http://www.dowdandassociates.com/blog/content/howto-use-hive-with-apache-logs/
- ...

Atividade 6 – Hive

- Enviar um arquivo PDF com uma descrição breve sobre a diferença entre os dados manipulados por uma aplicação MapReduce e os manipulados pelo Hive, justificando e exemplificando a utilização do Hive.

Ecosistema Hadoop HDP 2.1

- Hue – ecosistema Hadoop UI
- Beeswax – Hive UI (interface BDR → SQL)
- **Pig (PigLatin – Bash/SQL)**
- Hcatalog – Catálogo de bases de dados
- Filebrowser – HDFS UI
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- Ambari – Gerência do cluster e aplicações http://<ip>:8080 (admin:admin)
- Hbase – BD orientado a columa
- Knox – Segurança
- Storm – Stream ...

Apache Pig

- Projeto Apache – http://pig.apache.org
- Amplamente aceito e usado
  - Yahoo!, Twitter, Netflix, etc...
- Casos de uso
  - Auditoria de Logs de Usuários
  - Extract Transform Load (ETL)
  - “Limpeza” de Logs
  - Junções com outros fontes (BDs)
- http://wiki.apache.org/pig/PigLatin
- http://www.coreservlets.com/hadoop-tutorial/#Pig-1
- Hadoop: The Definitive Guide

Modos de Execução do Pig

- Local ($pig -x local)
  - JVM
  - Prototipação, experimentação e desenvolvimento

- Hadoop Mode ($pig | $pig -x mapreduce)
  - Pig “traduz” Pig Latin em Jobs MapReduce e executa-os no cluster Hadoop
  - Executado em clusters semi ou totalmente distribuído

- HUE
  - http://127.0.0.1:8000/pig/
  - http://127.0.0.1:8000/shell/ Pig Shell (Grunt)

Simple Pig Latin Example

```
$ pig
grunt> cat /training/playArea/pig/a.txt
a       1
d       4
c       9
k       6
grunt> records = LOAD '/training/playArea/pig/a.txt' as
(letter:chararray, count:int);
grunt> dump records;
...
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer
.MapReduceLauncher - 50% complete
2012-07-14 17:36:22,040 [main] INFO
org.apache.pig.backend.hadoop.executionengine.mapReduceLayer
.MapReduceLauncher - 100% complete
...
(a,1)
(d,4)
(c,9)
(k,6)
grunt>
```

Start Grunt with default MapReduce mode

Grunt supports file system commands

Load contents of text files into a Bag named *records*

Display records bag to the screen

Results of the bag named records are printed to the screen

Case Sensitivity and Conventions

- Case Sensitive
  - Alias names
  - Functions
- Case Insensitive
  - Pig Latin Keywords

```
counts = ForEACH charGroup GeNERATE group,
COUNT(c);
```

- General conventions
  - Upper case is a system keyword
  - Lowercase is something that you provide

Pig Latin Concepts

- Data Types
  - Field – piece of data (data atom)
    10.4 5 word 4
  - Tuple – ordered set of fields, denoted by (and)
    (10.4, 5, word, 4)
  - Bag – collection of tuples, denoted by chaves { and }
    { (10.4, 5, word, 4), (7.5, 7, this, 1) }
  - Map – collection of tuples, denoted by colchetes [and]
    [k1#(10.4, 5, word, 4); k2#(7.5, 7, this, 1)]
- Similar to Relational Database
  - Bag is a table in the database
  - Tuple is a row in a table

* Bags do not require that all tuples contain the same fields

Pig waits DUMP and STORE

- No action is taken until DUMP or STORE commands are encountered
  - Pig will parse, validate and analyze statements but not execute them

```
grunt> records = LOAD '/training/playArea/pig/a.txt' as
(letter:chararray, count:int);
grunt> ...
grunt> ...
grunt> DUMP final_bag;
```

DUMP – displays results to the screen

STORE – saves results (typically to a file)

Pig Script 1

```
lines = LOAD '/mysqld.log' AS (line:chararray);
words = FOREACH lines GENERATE flatten(TOKENIZE(line)) AS word:chararray;
wordGroup = GROUP words BY word;
countWord = FOREACH wordGroup GENERATE group, COUNT(words);
orderedCountWord = ORDER countWord BY $1 DESC;
DUMP orderedCountWord;
result = LIMIT orderedCountWord 1;
STORE result INTO '/saidaPig';
```

Atividade 7 – Pig

- Enviar um pdf com as saídas da execução do Script 1, descrevendo o que ocorre em cada linha/comando.

DESCRIBE and GROUP

```
grunt> chars = LOAD '/training/playArea/pig/b.txt' AS
(c:chararray);
grunt> describe chars;
chars: {c: chararray}
grunt> dump chars;
(a)
(k)
...
...
(k)
(c)
(k)
grunt> charGroup = GROUP chars by c;
grunt> describe charGroup;
charGroup: {group: chararray,chars: {(c: chararray)}}
grunt> dump charGroup;
(a,{(a),(a),(a)})
(c,{(c),(c)})
(i,{(i),(i),(i)})
(k,{(k),(k),(k),(k)})
(l,{(l),(l)})
```

Creates a new bag with element named *group* and element named *chars*

The chars bag is grouped by “c”; therefore ‘group’ element will contain unique values

‘*chars*’ element is a bag itself and contains all tuples from ‘*chars*’ bag that match the value form ‘c’

ILUSTRATE

```
grunt> chars = LOAD '/training/playArea/pig/b.txt' AS (c:chararray);

grunt> charGroup = GROUP chars by c;

grunt> ILLUSTRATE charGroup;

-----------------------------
| chars    | c:chararray |
-----------------------------
|          | c           |
|          | c           |
-----------------------------
-------------------------------------------------------------------------
| charGroup | group:chararray       | chars:bag{:tuple(c:chararray)} |
-------------------------------------------------------------------------
|           | c                     | {(c), (c)}                     |
-------------------------------------------------------------------------
```

FOREACH

- FOREACH <bag> GENERATE <data>
  - Iterate over each element in the bag and produce a result
  - Ex: grunt> result = FOREACH bag GENERATE f1;

```
grunt> records = LOAD 'data/a.txt' AS (c:chararray, i:int);
grunt> dump records;
(a,1)
(d,4)
(c,9)
(k,6)
grunt> counts = FOREACH records GENERATE i;
grunt> dump counts;
(1)
(4)
(9)
(6)
```

FOREACH and Functions

- FOREACH b GENERATE group, <FUNCTION>(a);
  - Function: COUNT, CONCAT, SUM, … UDFs

```
grunt> chars = LOAD 'data/b.txt' AS (c:chararray);
grunt> charGroup = GROUP chars by c;
grunt> dump charGroup;
(a,{(a),(a),(a)})
(c,{(c),(c)})
(i,{(i),(i),(i)})
(k,{(k),(k),(k),(k)})
(l,{(l),(l)})
grunt> describe charGroup;
charGroup: {group: chararray,chars: {(c: chararray)}}
grunt> counts = FOREACH charGroup GENERATE group, COUNT(chars);
grunt> dump counts;
(a,3)
(c,2)
(i,3)
(k,4)
(l,2)
```

TOKENIZE

- Splits a string into tokens and outputs as a bag of tokens
  - space, double quote("), coma(,), parenthesis(()) and star(*)

```
grunt> linesOfText = LOAD 'data/c.txt' AS (line:chararray);
grunt> dump linesOfText;
(this is a line of text)
(yet another line of text)
(third line of words)
grunt> tokenBag = FOREACH linesOfText GENERATE TOKENIZE(line);
grunt> dump tokenBag;
({(this),(is),(a),(line),(of),(text)})
({(yet),(another),(line),(of),(text)})
({(third),(line),(of),(words)})
grunt> describe tokenBag;
tokenBag: {bag_of_tokenTuples: {tuple_of_tokens: (token: chararray)}}
```

STRSPLIT Functions

- Splits a string around matches of a given regular expression.
- STRSPLIT(string, regex, limit)
  - String - The string to be split.
  - Regex - The regular expression.
  - Limit - The number of times the pattern (the compiled representation of the regular expression) is applied.
- Ex: (open:source:software)
  - STRSPLIT (string, ':',2) → ((open,source:software))
  - STRSPLIT (string, ':',3) → ((open,source,software)).

String Functions

- INDEXOF
- LAST_INDEX_OF
- LCFIRST
- LOWER
- REGEX_EXTRACT
- REGEX_EXTRACT_ALL
- REPLACE
- STRSPLIT
- SUBSTRING
- TRIM
- UCFIRST
- UPPER

FLATTEN Operator

- Re-arranges output
  - Flattens nested bags and data types

```
grunt> dump tokenBag;
({(this),(is),(a),(line),(of),(text)})
({(yet),(another),(line),(of),(text)})
({(third),(line),(of),(words)})
grunt> flatBag = FOREACH tokenBag GENERATE flatten($0);
grunt> dump flatBag;
(this)
(is)
(a)
...
...
(third)
(line)
(of)
(words)
```

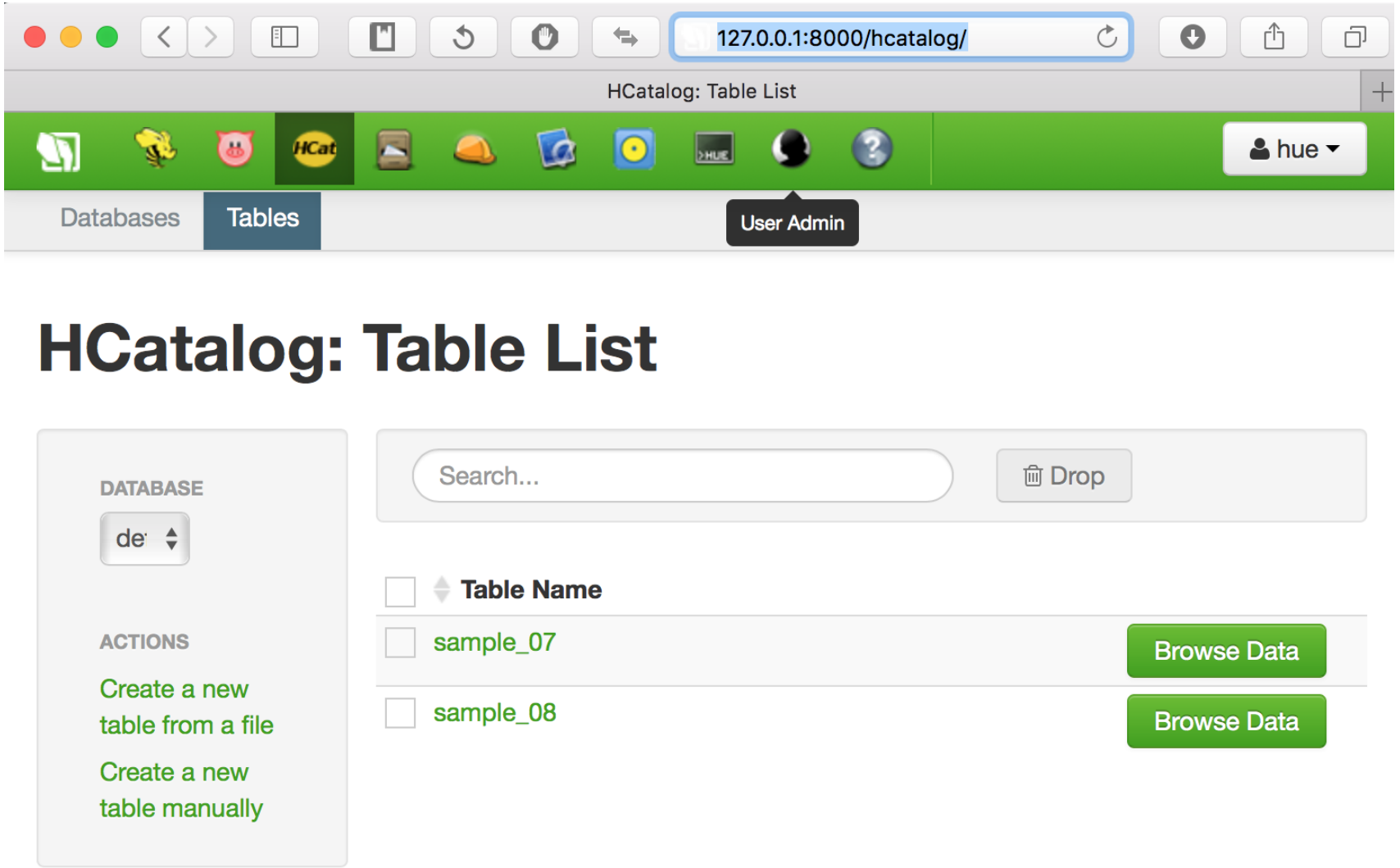
Pig Refs

- White, Tom. Hadoop: The Definitive Guide.
- Programing Pig. Alan Gates
- http://pig.apache.org/docs/r0.13.0/index.html
- http://pig.apache.org/docs/r0.13.0/basic.html
- https://pig.apache.org/docs/r0.7.0/piglatin_ref1.html
- https://pig.apache.org/docs/r0.7.0/piglatin_ref2.html
- https://cwiki.apache.org/confluence/display/PIG/PigTools
- ...

Ecosistema Hadoop HDP 2.1

- Hue – ecosistema Hadoop UI
- Beeswax – Hive UI (interface BDR → SQL)
- Pig (PigLatin – Bash/SQL)
- **Hcatalog – Catálogo de bases de dados**
- Filebrowser – HDFS UI
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- Ambari – Gerência do cluster e aplicações http://<ip>:8080 (admin:admin)
- Hbase – BD orientado a columa
- Knox – Segurança
- Storm – Stream ...

HCatalog


127.0.0.1:8000/hcatalog/
HCatalog: Table List
hue
Databases
Tables
User Admin
HCatalog: Table List
DATABASE
de
ACTIONS
Create a new table from a file
Create a new table manually
Search...
Drop
Table Name
sample_07
Browse Data
sample_08
Browse Data

HCatalog e Pig

Pig UI

```
teste2 = LOAD 'default.teste2’
  USING org.apache.hcatalog.pig.HCatLoader();
limit1 = LIMIT teste2 1;
DUMP limit1;
```

Ecosistema Hadoop HDP 2.1

- Hue – ecosistema Hadoop UI
- Beeswax – Hive UI (interface BDR → SQL)
- Pig (PigLatin – Bash/SQL)
- Hcatalog – Catálogo de bases de dados
- Filebrowser – HDFS UI
- **Job Browser – Hadoop Jobs**
- **Job Designer – Aplicações Hadoop (Sqoop/Attic dblink SGBDR)**
- **Oozie designer – Diversos sistemas/aplicações (Airflow)**
- Ambari – Gerência do cluster e aplicações
- Hbase – BD orientado a columa
- **Knox – Segurança**
- Storm – Stream ...

- :8088 – ResourceManager Web UI
  - nodes slaves ativos, senão reinicar NodeManager nos slaves
- :19888 – jobhistory
  - quais máquinas participaram da execução (Job ID, Map|Reduce, task*, Node)

Ecosistema Hadoop HDP 2.1

- Hue – ecosistema Hadoop UI
- Beeswax – Hive UI (interface BDR → SQL)
- Pig (PigLatin – Bash/SQL)
- Hcatalog – Catálogo de bases de dados
- Filebrowser – HDFS UI
- Job Browser – Hadoop Jobs
- Job Designer – aplicações Hadoop
- Oozie designer – Diversos sistemas/aplicações
- Ambari – Gerência do cluster e aplicações http://<ip>:8080 (admin:admin)
- **Hbase – BD orientado a columa**
- Knox – Segurança
- Storm – Stream ...

Apache HBase

- https://hbase.apache.org
- Orientado a coluna ou família de Colunas (key/value)
- Executa sobre Hadoop e HDFS
- Muitas tabelas grantes – bilhões de linhas X milhões de colunas
- Consistência robusta para leitura e escritas
- Nós Master para coordenar servidores que distribuem e processam partes dos dados das tabelas
- Programação em Java e comandos **HBase Shell**
  - `create, list, describe, put, get, scan, drop,...`
  - `https://hbase.apache.org/book.html`

Columnar/ColumnFamily Store

- Armazena os dados em colunas ao invés de linhas
- Linha (ID,Last,First,Bonus)

```
1,Doe,John,8000
2,Smith,Jane,4000
3,Beck,Sam,1000
```

- Coluna (ID,Last,First,Bonus)

```
1,2,3
Doe,Smith,Beck
John,Jane,Sam
8000,4000,1000
```

- MIN, MAX, SUM, COUNT e AVG (OLAP/DW)
- Auto-Indexação → menor espaço e maior rapidez
- Ex.: Hbase, Cassanda, Accumulo, Druid, C-Store

Modelos de Dados para Big Data

- Relacional, Objeto e O-R
- Dimensional/Multidimensional (DW)
- Dados Geográficos (GIS)
- Colunar
- Chave-valor
- Documento
- Grafo (Hierárquico e Rede)
- Não estruturados
  - Excel/CSV, Áudio, Vídeo, Logs, …

NoSQL

New-SQL

NoSQL, NoRel e NOSQL

- NoSQL'1998 - non-sql RDBMS
- NoRel - Non-Relational
- NOSQL - Not Only SQL
- **NoSQL → NoRel e NOSQL**

NoSQL

- Grandes conjuntos de dados variados e frequentemente atualizados (3Vs)
- *Scale-out* (*horizontally)* e não Scale-up (verticallly)
- Dados semi ou não-estruturados
- No ACID (*Atomicity Consistency Isolation Durability*)
  - **Desnormalização**
- CAP: *Consistency Availability Partition tolerance*
- Sem autenticação e autorização (padrão)
- Key-value, Document, Graph, Column (NewSQL)

Key-Value Store

- Conjunto associativo (map/hash) de pares <chave,valor>
- Modelo de dados < cpf, “joao, rua alguma, projeto3, dep4,...” >
  < row key+collumn key+ time, value >
  ...
- Sem linguagem de manipulação
- Tabelão – *Big Tables*
- Uma linha (RDBMS) é um valor para uma chave
- Recomendado para dados que não requeiram ou tenham esquema pré-definidos
- Ex.: MapReduce, Hadoop, Redis, Dynamo, MemcacheDB
- BigTable[Chang08], Google web indexing, Earth, Finance

Document Store

- Base é chave-valor mas com extensão para olhar o conteúdo
  - Chave: URI, path, id
  - Valor: documento
- Armazenar documentos em algum formato/codificação
  - XML, YAML, JSON, PDF,...
- Organização/Agrupamento
  - Coleções, Tags, Meta-dados, Hierarquia arquivos
- Ex.: Apache CouchDB, MongoDB, BaseX, SimpleDB

Graph Store

- Armazena Nó, Arestas e as suas ligações
- *Triplestore*/RDF(*Resource Description Framework*/W3C)
  - Sujeito, predicado, objeto

    João conhece Pedro
  - [Entidade|Objeto], atributo, valor

    João altura 80

    Bola cor azul
- Dados cujas relações são melhor representadas por grafos
  - Mapas, Rede, Relações sociais, transporte
- Ex.: Apache Hama, Faunus, InfiniteGraph, Neo4j

Columnar/ColumnFamily Store

- Armazena os dados em colunas ao invés de linhas
- Linha (ID,Last,First,Bonus)

```
1,Doe,John,8000
2,Smith,Jane,4000
3,Beck,Sam,1000
```

- Coluna (ID,Last,First,Bonus)

```
1,2,3
Doe,Smith,Beck
John,Jane,Sam
8000,4000,1000
```

- MIN, MAX, SUM, COUNT e AVG (OLAP/DW)
- Auto-Indexação → menor espaço e maior rapidez
- Ex.: Cassanda, Accumulo, Druid, Hbase, C-Store

NewSQL ou New SQL?

- Não é um novo SQL (http://newsql.sourceforge.net)
- DBMS novos, escaláveis e de alta performance
- Escalabilidade e Performance do NoSQL
- Com as propriedade ACID
- OLTP (On-Line Transaction Processing) para Leitura/Escrita

NewSQL Origins

- H-Store'07 → VoltDB'10
  - VLDB'07 - Mike Stonebraker, Sam Madden, Daniel Abadi, et al. **The end of an architectural era: (it's time for a complete rewrite).**
  - VLDB'08 Kallman, R. et al. **H-store: a high-performance, distributed main memory transaction processing system.**
  - (SIGMOD'09 Pavlo, A. et al. **A Comparison of Approaches to Large-Scale Data Analysis**.)

NewSQL

- In-memory, shared-nothing, horizontal/row, vertical/column, cloud, graph ... DB
- Memória/Coluna
  - SQL Server, Oracle, DB2 In-Memory|Column ...
- Escaláveis
  - MySQL Cluster(InnoDB), Microsoft SQL Azure, InfiniDB, ...

Relational x NoSQL

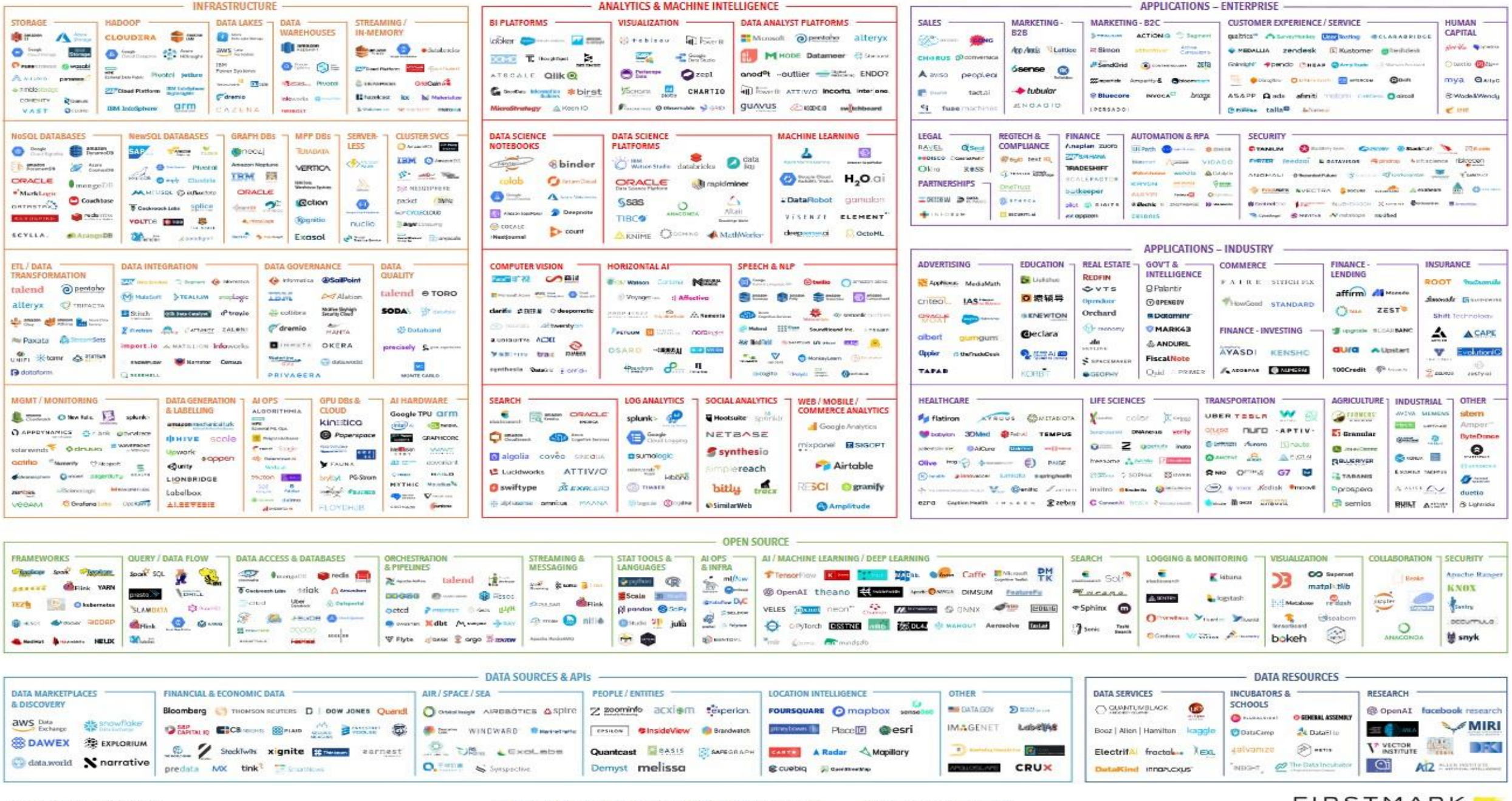
- Acabou o Relacional!
- NoSQL é a solução pra tudo!
- *RelationalDB are far from dead...*
  - http://searchcio.techtarget.com/opinion/Relational-databases-are-far-from-dead-just-ask-Facebook

Relational x NoSQL by Facebook'12

- Exploratory analysis → NoSQL/Hadoop
  Look at the granularity of the data(lowest level)
- Operational analysis → NewSQL/DDBMS
  Look at transformed and aggregated data
- Real-time monitoring, Data streams → NoSQL/Hadoop
- Trending analysis(d m y) → NewSQL/DDBMS

Facebook projects: https://code.facebook.com/

DATA & AI LANDSCAPE 2020



mattturck.com/data2020

FIRSTMARK
EARLY STAGE VENTURE CAPITAL

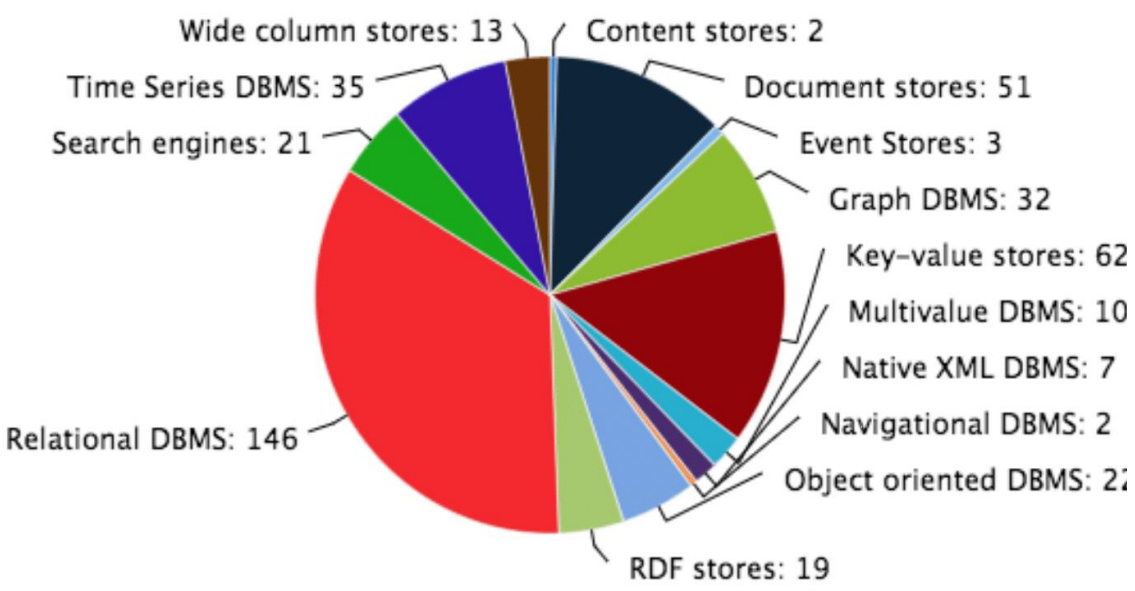
Atividade 8 – Quiz

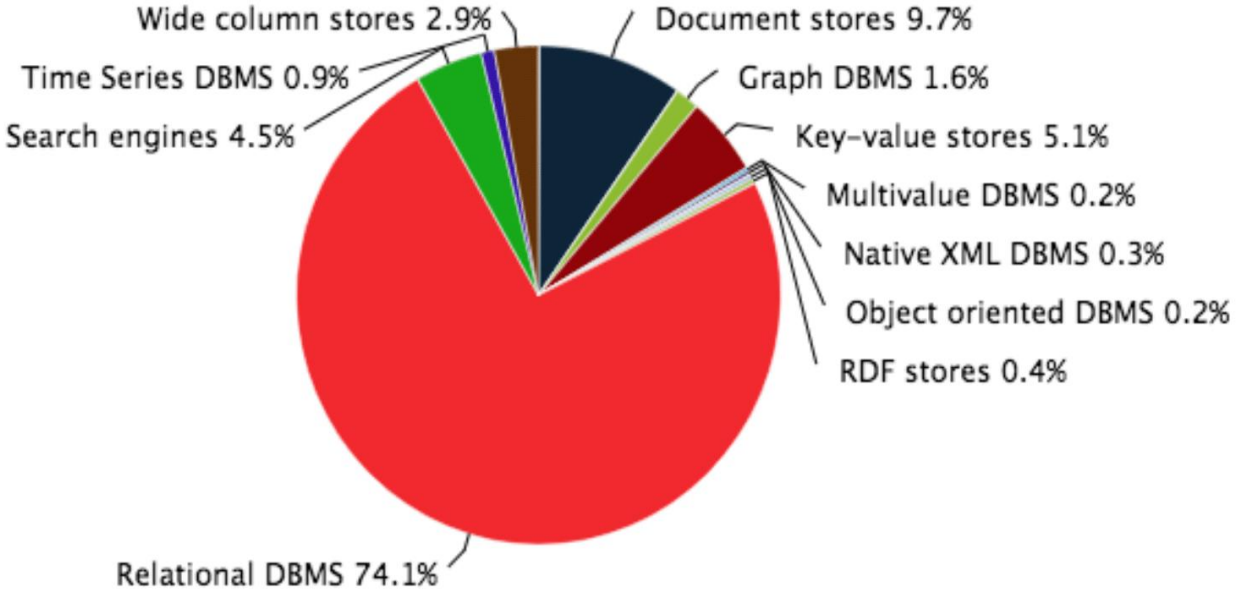
- Enviar um arquivo pdf com as suas respostas para o quiz:

  Big Data Technologies
  - http://tinyurl.com/jqmuptk
  - Ou http://searchdatamanagement.techtarget.com/quiz/Quiz-How-do-relational-databases-and-NoSQL-technologies-compare

* Imprimir salvando como pdf.

Sistemas por Categoria (04/2021)





Hoje??? https://db-engines.com/en/ranking_categories

Evolução das Popularidades

https://db-engines.com/en/ranking_trend

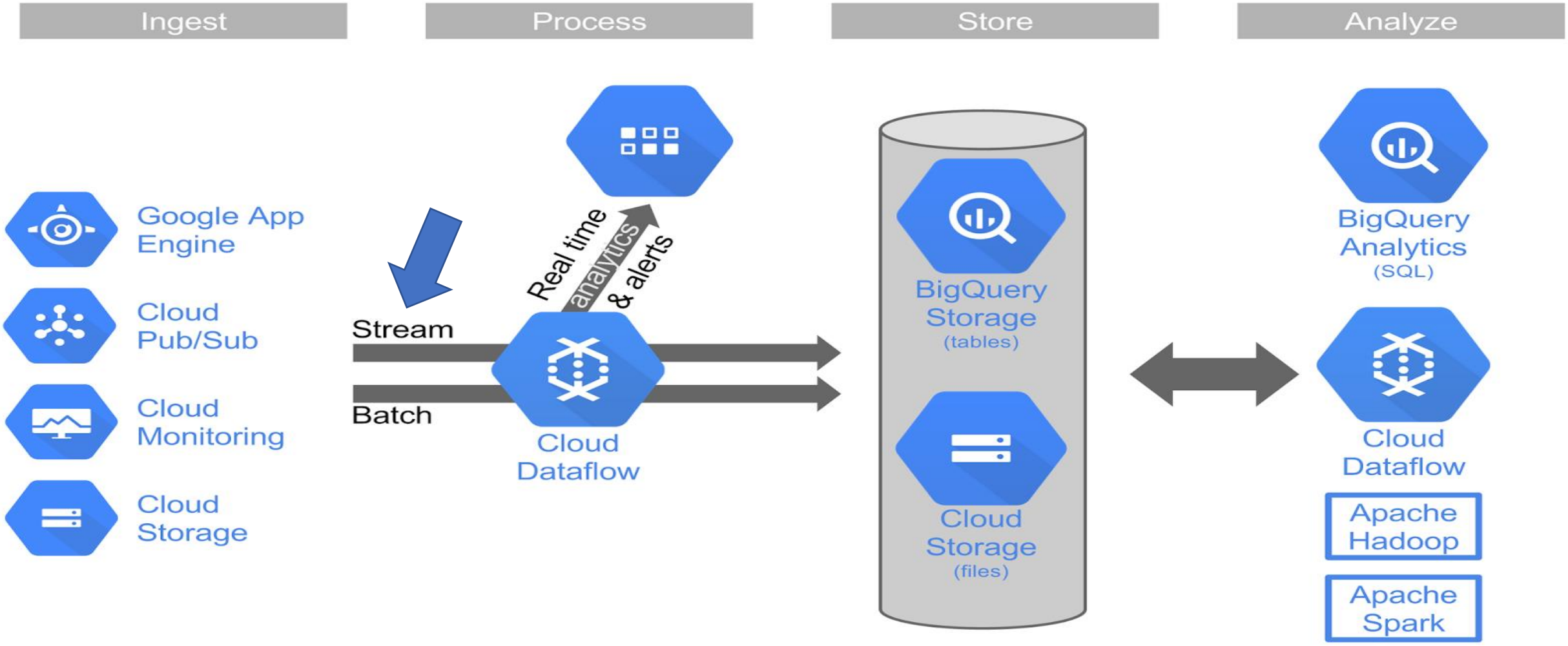
Programa

- Fundamentos de Big Data
  - Big Data, Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - Personalização de aplicações Map Reduce
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
- NoSQL e NewSQL
- **Dados em movimento – Processamento de Streaming**

Novo Problema

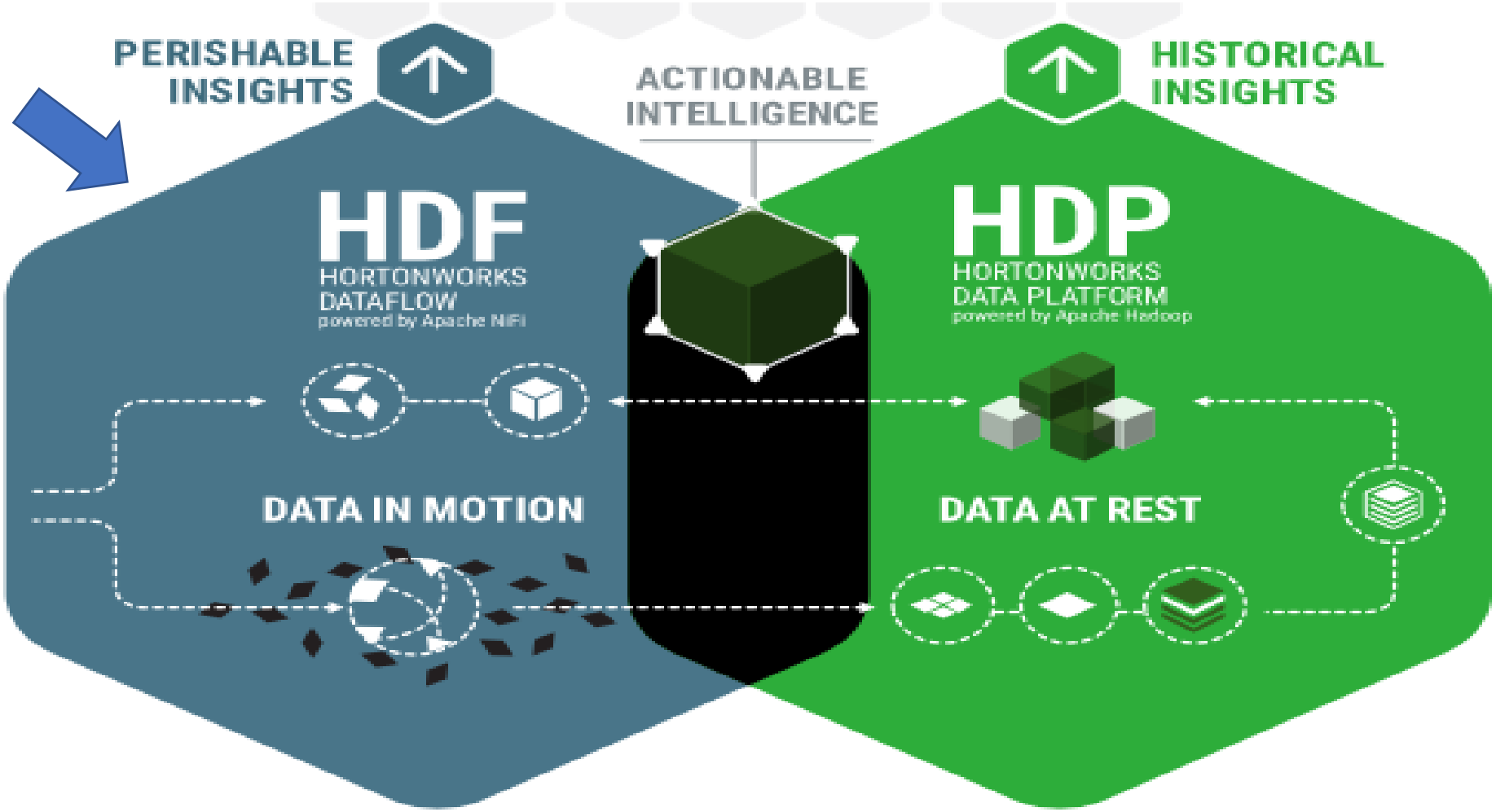
- Como processar eventos complexos e fluxo (streaming) de dados?
  - Banco – Fraude, Cartão, Investimento, ...
  - Comércio – Precificação dinâmica, controle estoque, ...
  - Redes sociais – Tendências, ofertas, ...

Google Public Big Data Cloud



Dados em Movimento e Parados

- http://br.hortonworks.com/solutions/#use-cases



APACHE STORM™

Distributed • Resilient • Real-time

- http://storm.apache.org
- Sistema de Computação em Tempo Real de código aberto
- Processamento de grande volume e velocidade de dados
- Rápido, Escalável, Tolerante a falhas, Confiável e "*Fácil*"
- Análises em tempo real, aprendizado de máquina, computação contínua, ETL distribuída e incremental,...
- Sistema de Processamento de Eventos Complexos

História do Apache Storm

- Storm (2011) by Nathan Marz
- Twitter Storm (2012-2013) - 140 mi tweets/day
- Apache Storm (17/09/2014)
  - Twitter, Yahoo!, Alibaba, Microsoft, Hortonworks, ...
  - Cisco, Spotify, Xerox PARC, Groupon, WebMD, ...

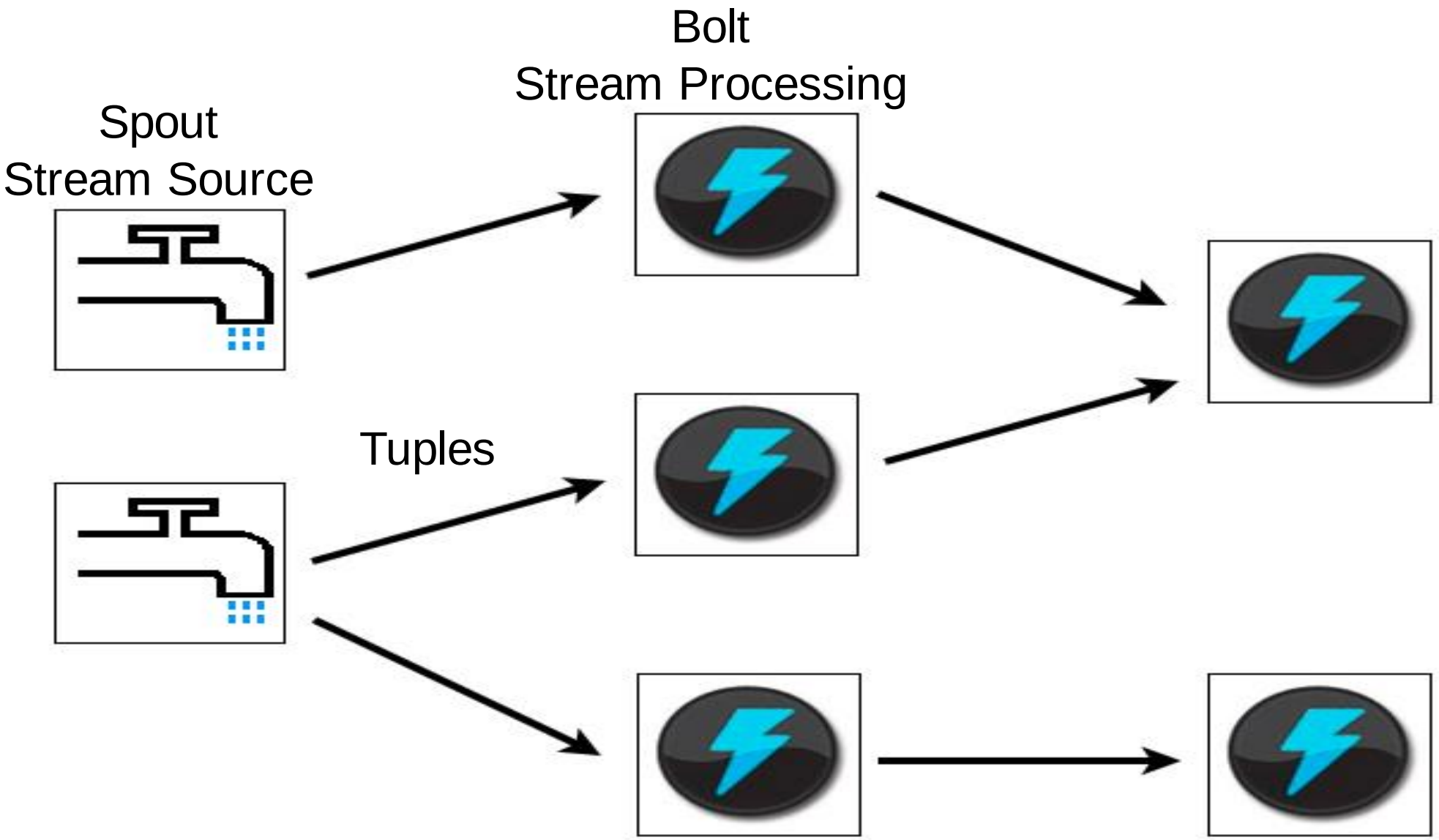
Apache Storm

- Clojure (*closure*), dialeto Lisp (funcional)
  - Simplifica a programação multithreaded
  - Clojure é uma linguagem baseada em VM (estilo JVM)
- APIs para Java, Scala, JRuby, Perl, PHP e SQL
- Hadoop Distributed File System (HDFS), HBase e Apache Kafka
- Serialização compactada
- Suporte para
  - Segurança
  - Sistema de gerenciamento de recursos (Apache YARN)

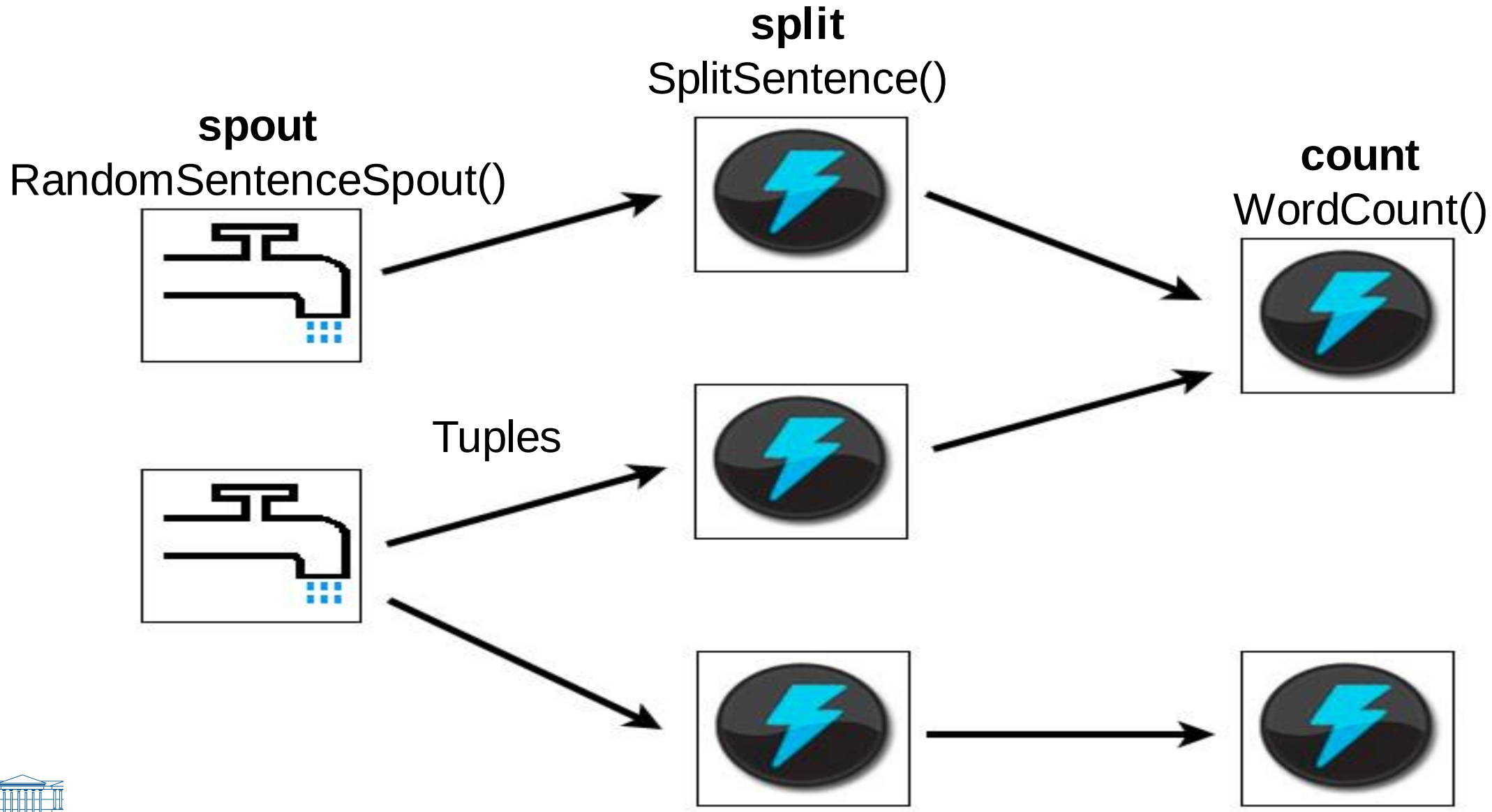
Storm Terminologies and Concepts

- Tuples
  - An ordered list of elements – 4-tuple (7, 1, 3, 7)
- Streams
  - An unbounded sequence of tuples
- Spouts
  - Sources of streams in a computation
- Bolts
  - Process input streams and produce output streams
  - Functions, filter, aggregate, or join data, and databases
- Topologies
  - The overall calculation, represented visually as a network of spouts and bolts

Storm Topology



Topologia WordCount



WordCount Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 2);

builder.setBolt(”split", new SplitSentence(), 3)
           .shuffleGrouping("spout");
builder.setBolt(”count", new WordCount(), 2)
           .fieldsGrouping(”split", new Fields("word"));

Config conf = new Config();
LocalCluster cluster = new LocalCluster();
cluster.submitTopology(args[0], conf,
builder.createTopology());
```

Execução do Wordcount no Storm da HDP 2.1

- Start Ambari
  - http://127.0.0.1:8000 (Ambari, Enable)
- Start Storm
  - Ambari, http://127.0.0.1:8080 (admin:admin)
    (Storm, Service Actions, Start)
- Storm UI
  - http://127.0.0.1:8744
- Submit an wordcount topology to Storm

```
storm jar /usr/lib/storm/contrib/storm-starter/storm-starter-0.9.1.2.1.1.0-385-jar-with-dependencies.jar storm.starter.WordCountTopology WordCount
```

Obs: storm jar storm*.jar <classe para executar> <nome da topologia>

Submetendo uma Topologia

```
[root@sandbox ~]# storm jar /usr/lib/storm/contrib/storm-starter/storm-starter-0.9.1.2.1.1.0-385-jar-with-dependencies.jar
storm.starter.WordCountTopology WordCount
Running: java -client -Dstorm.options= -Dstorm.home=/usr/lib/storm -Dstorm.log.dir=/usr/lib/storm/logs -
Djava.library.path=/usr/local/lib:/opt/local/lib:/usr/lib -Dstorm.conf.file= -cp /usr/lib/storm/lib/kryo-2.17.jar:/usr/lib/storm/lib/logback-classic-
1.0.6.jar:/usr/lib/storm/lib/slf4j-api-1.6.5.jar:/usr/lib/storm/lib/commons-exec-1.1.jar:/usr/lib/storm/lib/clj-stacktrace-
0.2.4.jar:/usr/lib/storm/lib/httpclient-4.1.1.jar:/usr/lib/storm/lib/joda-time-2.0.jar:/usr/lib/storm/lib/commons-codec-
1.4.jar:/usr/lib/storm/lib/jetty-util-6.1.26.jar:/usr/lib/storm/lib/servlet-api-2.5.jar:/usr/lib/storm/lib/jline-
2.11.jar:/usr/lib/storm/lib/carbonite-1.3.2.jar:/usr/lib/storm/lib/reflectasm-1.07-shaded.jar:/usr/lib/storm/lib/curator-client-
1.3.3.jar:/usr/lib/storm/lib/clj-time-0.4.1.jar:/usr/lib/storm/lib/ring-servlet-0.3.11.jar:/usr/lib/storm/lib/jgrapht-core-
0.9.0.jar:/usr/lib/storm/lib/disruptor-2.10.1.jar:/usr/lib/storm/lib/jetty-6.1.26.jar:/usr/lib/storm/lib/ring-core-
1.1.5.jar:/usr/lib/storm/lib/guava-13.0.jar:/usr/lib/storm/lib/ring-devel-0.3.11.jar:/usr/lib/storm/lib/logback-core-
1.0.6.jar:/usr/lib/storm/lib/commons-io-2.4.jar:/usr/lib/storm/lib/meat-locker-0.3.1.jar:/usr/lib/storm/lib/tools.cli-
0.2.2.jar:/usr/lib/storm/lib/clojure-1.4.0.jar:/usr/lib/storm/lib/asm-4.0.jar:/usr/lib/storm/lib/netty-
3.6.3.Final.jar:/usr/lib/storm/lib/tools.macro-0.1.0.jar:/usr/lib/storm/lib/math.numeric-tower-0.0.1.jar:/usr/lib/storm/lib/curator-framework-
1.3.3.jar:/usr/lib/storm/lib/compojure-1.1.3.jar:/usr/lib/storm/lib/ring-jetty-adapter-0.3.11.jar:/usr/lib/storm/lib/storm-core-0.9.1.2.1.1.0-
385.jar:/usr/lib/storm/lib/snakeyaml-1.11.jar:/usr/lib/storm/lib/clout-1.0.1.jar:/usr/lib/storm/lib/httpcore-4.1.jar:/usr/lib/storm/lib/commons-lang-
2.5.jar:/usr/lib/storm/lib/hiccup-0.3.6.jar:/usr/lib/storm/lib/zookeeper.jar:/usr/lib/storm/lib/commons-fileupload-
1.2.1.jar:/usr/lib/storm/lib/minlog-1.2.jar:/usr/lib/storm/lib/tools.logging-0.2.3.jar:/usr/lib/storm/lib/log4j-over-slf4j-
1.6.6.jar:/usr/lib/storm/lib/servlet-api-2.5-20081211.jar:/usr/lib/storm/lib/netty-3.2.2.Final.jar:/usr/lib/storm/lib/json-simple-
1.1.jar:/usr/lib/storm/lib/core.incubator-0.1.0.jar:/usr/lib/storm/lib/commons-logging-1.1.1.jar:/usr/lib/storm/lib/objenesis-
1.2.jar:/usr/lib/storm/contrib/storm-starter/storm-starter-0.9.1.2.1.1.0-385-jar-with-dependencies.jar:/usr/lib/storm/conf:/usr/lib/storm/bin -
Dstorm.jar=/usr/lib/storm/contrib/storm-starter/storm-starter-0.9.1.2.1.1.0-385-jar-with-dependencies.jar storm.starter.WordCountTopology WordCount
1005 [main] INFO  backtype.storm.StormSubmitter - Jar not uploaded to master yet. Submitting jar...
1052 [main] INFO  backtype.storm.StormSubmitter - Uploading topology jar /usr/lib/storm/contrib/storm-starter/storm-starter-0.9.1.2.1.1.0-385-jar-
with-dependencies.jar to assigned location: /hadoop/storm/nimbus/inbox/stormjar-1d482864-4e54-4570-8f41-62e5c940846c.jar
1445 [main] INFO  backtype.storm.StormSubmitter - Successfully uploaded topology jar to assigned location: /hadoop/storm/nimbus/inbox/stormjar-
1d482864-4e54-4570-8f41-62e5c940846c.jar
1446 [main] INFO  backtype.storm.StormSubmitter - Submitting topology WordCount in distributed mode with conf
{"topology.workers":3,"topology.debug":true}
2129 [main] INFO  backtype.storm.StormSubmitter - Finished submitting topology: WordCount
[root@sandbox ~]#
```

Parte do Processamento Stream

```
[root@sandbox ~]# tail -f /var/log/storm/worker-6701.log
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: split:16, stream: default, id: {}, [years]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: count default [years, 10755]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["over"]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: split:21, stream: default, id: {}, ["over"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: count default [over, 10640]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["the"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: spout default [the cow jumped over the moon]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["moon"]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: split:16, stream: default, id: {}, [over]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: count default [over, 10641]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: spout:26, stream: default, id: {}, [four score and seven years ago]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["four"]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: split:21, stream: default, id: {}, ["four"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["score"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: count default [four, 10756]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: split:21, stream: default, id: {}, ["score"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: count default [score, 10756]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["and"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["seven"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["years"]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: split:21, stream: default, id: {}, ["years"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: count default [years, 10756]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["ago"]
2018-09-20 16:05:46 b.s.d.executor [INFO] Processing received message source: spout:24, stream: default, id: {}, [an apple a day keeps the doctor
away]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: spout default [snow white and the seven dwarfs]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["an"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["apple"]
2018-09-20 16:05:46 b.s.d.task [INFO] Emitting: split default ["a"]
```

Parte do Processamento Stream (2)

```
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: spout default [the cow jumped over the moon]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: spout:27, stream: default, id: {}, [the cow jumped over the moon]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["the"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["cow"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["jumped"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["over"]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:19, stream: default, id: {}, ["over"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [over, 11862]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["the"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["moon"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: spout default [an apple a day keeps the doctor away]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:16, stream: default, id: {}, [apple]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [apple, 12097]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:16, stream: default, id: {}, [day]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [day, 12097]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:16, stream: default, id: {}, [keeps]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [keeps, 12097]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:16, stream: default, id: {}, [away]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [away, 12097]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: spout default [an apple a day keeps the doctor away]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:20, stream: default, id: {}, [apple]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [apple, 12098]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:20, stream: default, id: {}, [day]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [day, 12098]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: spout default [four score and seven years ago]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: spout:25, stream: default, id: {}, [four score and seven years ago]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["four"]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:23, stream: default, id: {}, ["four"]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [four, 11973]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: split default ["score"]
```

Finalização do Processamento Stream

```
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [two, 11911]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: split:22, stream: default, id: {}, [with]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: count default [with, 11911]
2018-09-20 16:07:56 b.s.d.task [INFO] Emitting: spout default [four score and seven years ago]
2018-09-20 16:07:56 b.s.d.executor [INFO] Processing received message source: spout:25, stream: default, id: {}, [four score and seven years ago]
2018-09-20 16:07:56 b.s.util [ERROR] Async loop died!
java.lang.RuntimeException: java.lang.RuntimeException: java.lang.RuntimeException: Pipe to subprocess seems to be broken! No output read.
Shell Process Exception:


            at backtype.storm.utils.DisruptorQueue.consumeBatchToCursor(DisruptorQueue.java:107) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-
385]
            at backtype.storm.utils.DisruptorQueue.consumeBatchWhenAvailable(DisruptorQueue.java:78) ~[storm-core-0.9.1.2.1.1.0-
385.jar:0.9.1.2.1.1.0-385]
            at backtype.storm.disruptor$consume_batch_when_available.invoke(disruptor.clj:77) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-385]
            at backtype.storm.daemon.executor$fn__4681$fn__4693$fn__4740.invoke(executor.clj:745) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-
385]
            at backtype.storm.util$async_loop$fn__442.invoke(util.clj:436) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-385]
            at clojure.lang.AFn.run(AFn.java:24) [clojure-1.4.0.jar:na]
            at java.lang.Thread.run(Thread.java:744) [na:1.7.0_45]
Caused by: java.lang.RuntimeException: java.lang.RuntimeException: Pipe to subprocess seems to be broken! No output read.
Shell Process Exception:


            at backtype.storm.task.ShellBolt.execute(ShellBolt.java:164) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-385]
            at backtype.storm.daemon.executor$fn__4681$tuple_action_fn__4683.invoke(executor.clj:630) ~[storm-core-0.9.1.2.1.1.0-
385.jar:0.9.1.2.1.1.0-385]
            at backtype.storm.daemon.executor$mk_task_receiver$fn__4604.invoke(executor.clj:398) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-
385]
            at backtype.storm.disruptor$clojure_handler$reify__3405.onEvent(disruptor.clj:58) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-385]
            at backtype.storm.utils.DisruptorQueue.consumeBatchToCursor(DisruptorQueue.java:104) ~[storm-core-0.9.1.2.1.1.0-385.jar:0.9.1.2.1.1.0-
385]
            ... 6 common frames omitted
```

Administração do Storm

- Listar topologias ativas
  - `storm list`
  - `http://127.0.0.1:8744`
- Sumário da topologia
  - `http://127.0.0.1:8744 (clicar no nome da topologia)`
- Detalhes dos Spout e Bolts
  - `http://127.0.0.1:8744 (clicar nos spouts e bolts)`
- Logs (de acordo com os executores/portas)
  - `less /var/log/storm/worker-6700.log`
  - `tail -f /var/log/storm/worker-6701.log`
    `(Crt+C para sair)`
- Finalizar
  - `storm kill WordCount`
  - `http://127.0.0.1:8744 (Kill)`

Atividade 9 – Storm

- Enviar um arquivo PDF respondendo qual a diferença entre os dados manipulados por aplicações Hadoop e pelo Storm? Comente brevemente justificando e apresentando uma possível utilização do Storm.

Atividade 10 – Estudo de Caso

- Enviar um arquivo PDF contendo uma descrição breve (2 páginas) sobre a implementação de uma aplicação ou estudo de caso envolvendo Big Data e suas ferramentas (NoSQL/Streaming).
  - Caracterizar os dados e seus Vs, e sobre a modelagem

- Também preparar uma apresentação (5 min) para a próxima aula.

Próxima Parte

- Fundamentos de Big Data
  - Data Lake e Data Science
- Map Reduce e Hadoop
  - Utilização da Sandbox/VM
  - Personalização de aplicações Map Reduce
- Data Engineering (Gerenciamento/Ferramentas para Big Data)
- NoSQL e NewSQL
- Dados em movimento – Processamento de Streaming

- Detalhes sórdidos e ruminar tudo isso aí!