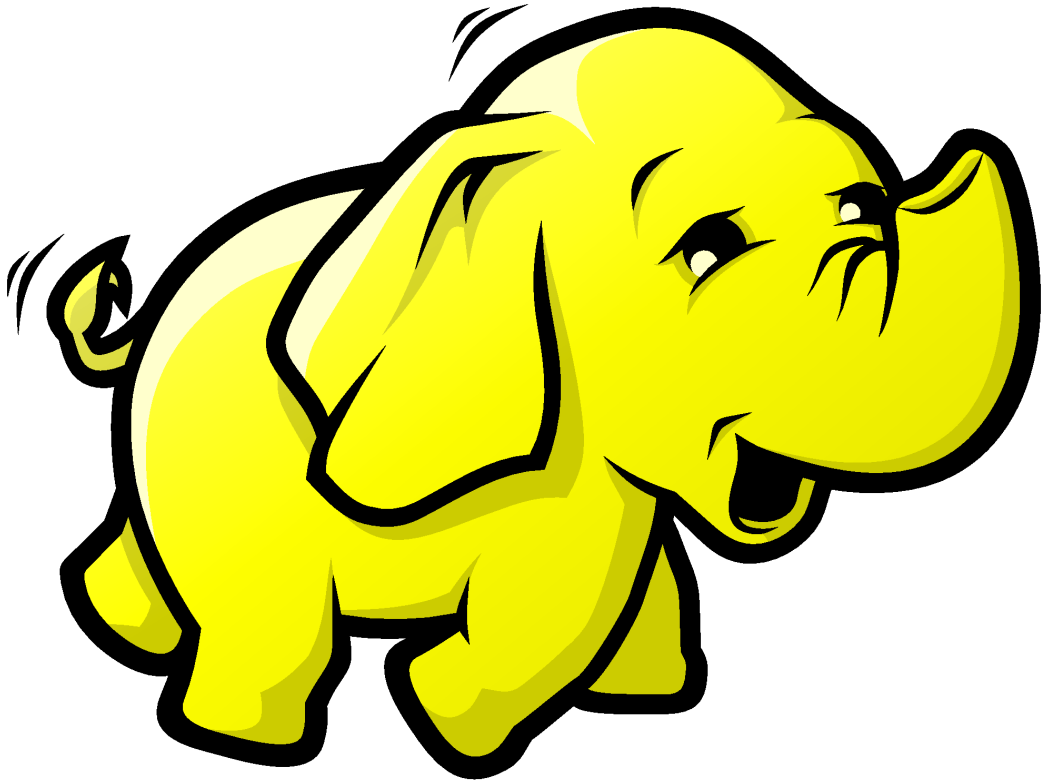
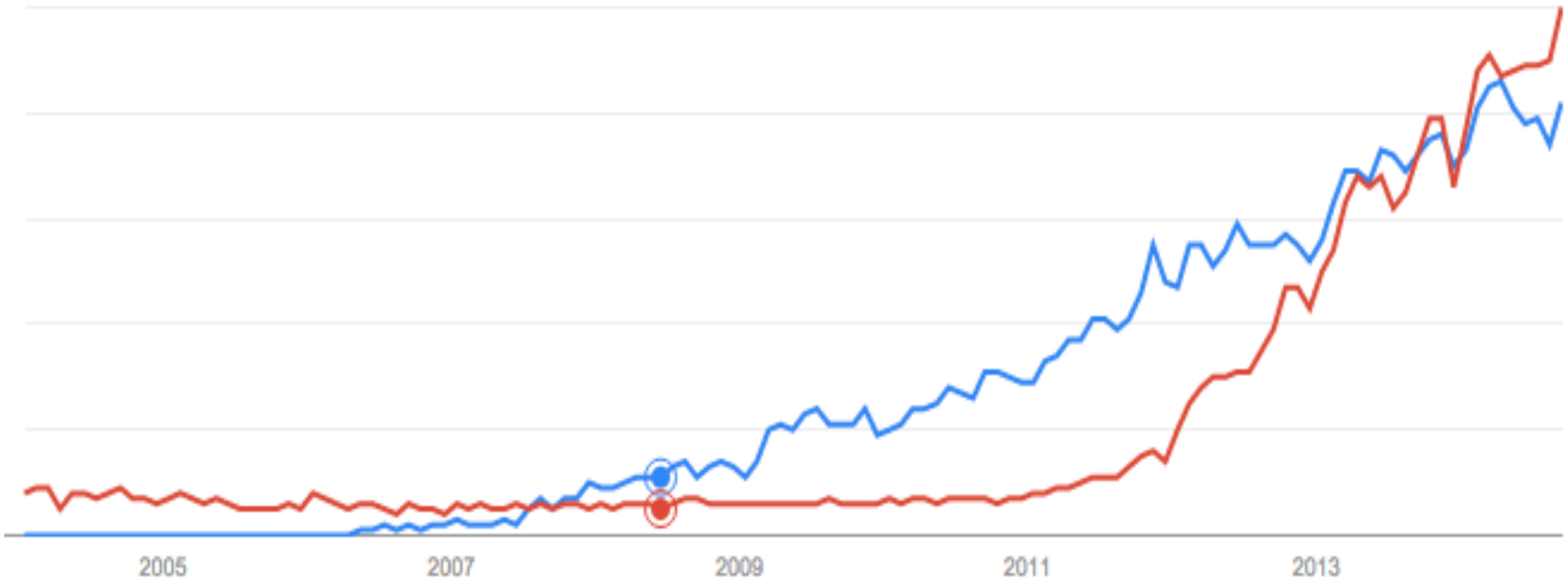
Hadoop!
Szybki start

Arkadiusz Osiński
arkadiusz.osinski@allegrogroup.com
arek.osinski@i-netpl.info

Agenda

1. Hadoop ?!
2. Architektura w słowach kilku
3. Co z tym klastrem ?
4. Jak żyć z ekosystemem ?

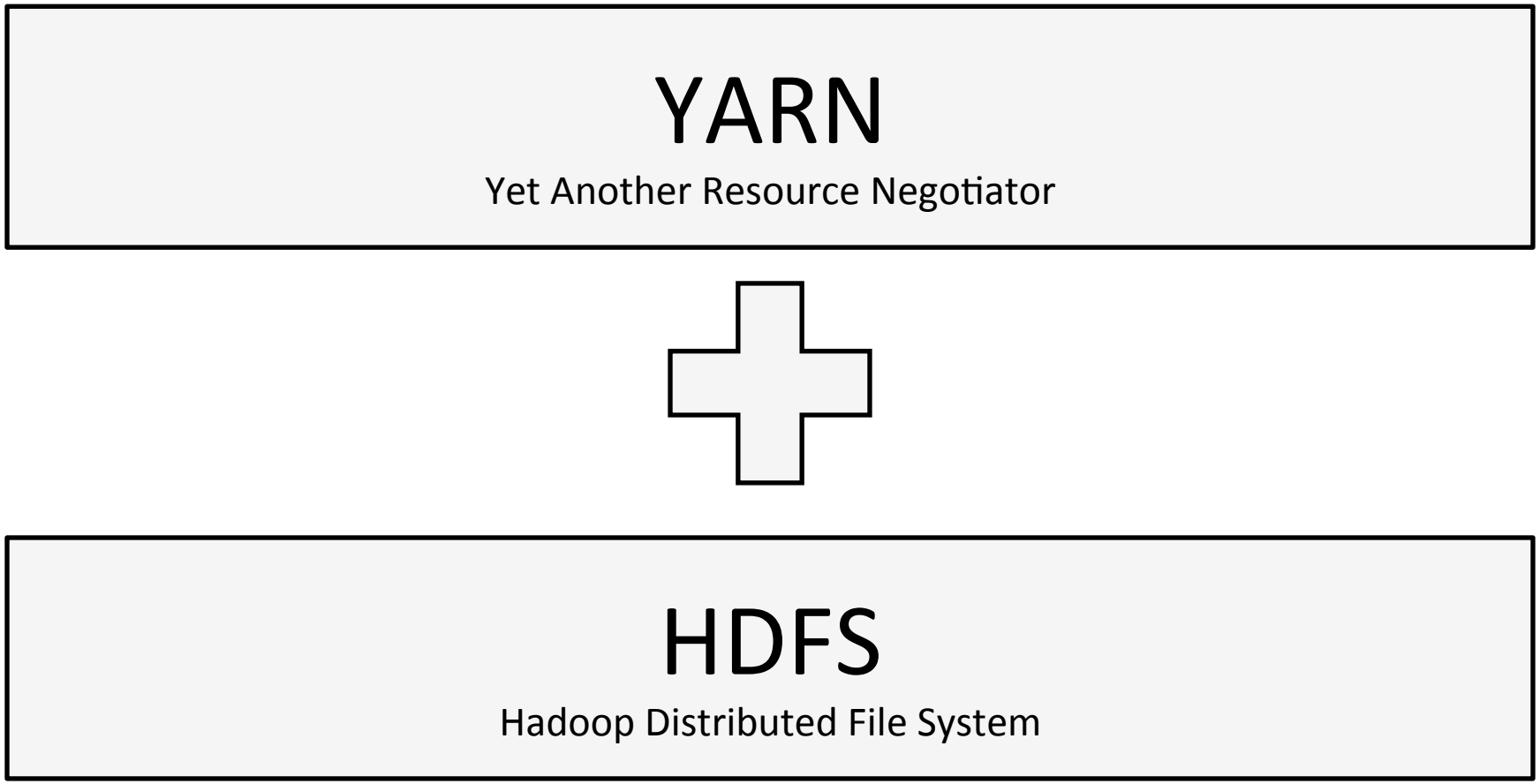




Big data

Hadoop

Hadoop


YARN
Yet Another Resource Negotiator
HDFS
Hadoop Distributed File System

HDFS

- Warstwa przechowywania danych

HDFS

- Warstwa przechowywania danych
- Rozproszony FS

HDFS

- Warstwa przechowywania danych
- Rozproszony FS
- Skalowalny

HDFS

- Warstwa przechowywania danych
- Rozproszony FS
- Skalowalny
- JBOD

HDFS

- Warstwa przechowywania danych
- Rozproszony FS
- Skalowalny
- JBOD
- Kontrola dostępu

HDFS

- Warstwa przechowywania danych
- Rozproszony FS
- Skalowalny
- JBOD
- Kontrola dostępu
- HA

YARN

- Rozproszone przetwarzanie danych

YARN

- Rozproszone przetwarzanie danych
- Zarządzanie zasobami

YARN

- Rozproszone przetwarzanie danych
- Zarządzanie zasobami
- Lokalizacja zadań

YARN

- Rozproszone przetwarzanie danych
- Zarządzanie zasobami
- Lokalizacja zadań
- MapReduce i nie tylko

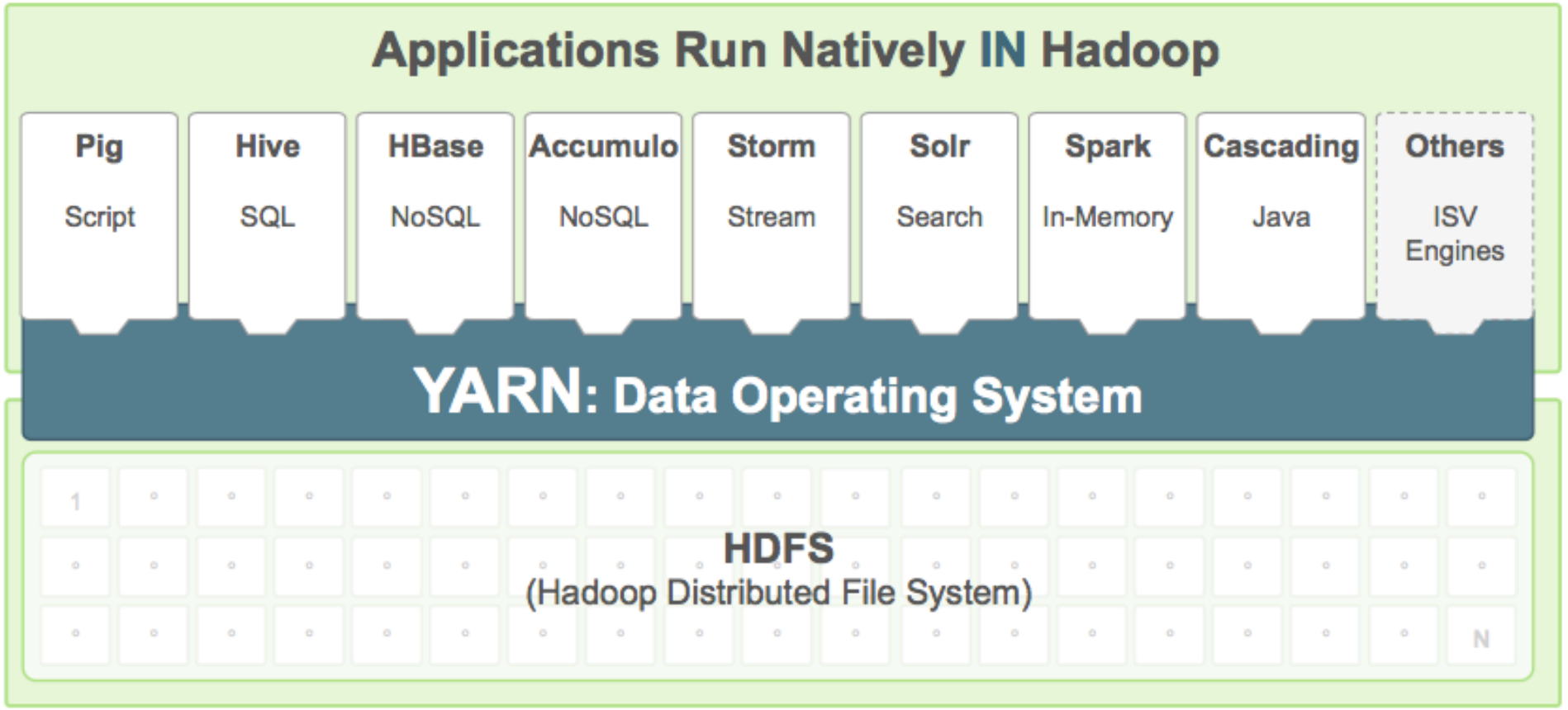
YARN

- Rozproszone przetwarzanie danych
- Zarządzanie zasobami
- Lokalizacja zadań
- MapReduce i nie tylko
- Kolejki zadań

YARN

- Rozproszone przetwarzanie danych
- Zarządzanie zasobami
- Lokalizacja zadań
- MapReduce i nie tylko
- Kolejki zadań
- HA

Hadoop



Źródło: Hortonworks

Pierwszy klaster

```
aws emr create-cluster \
--ami-version 3.2.1 \
--instance-groups
"InstanceGroupType=MASTER,InstanceCount=1,InstanceType=m1.large"
"InstanceGroupType=CORE,InstanceCount=3,InstanceType=m1.xlarge" \
--no-auto-terminate \
--name "emr-tjug1" \
--use-default-roles \
--ec2-attributes "KeyName=ao" \
--tags "Name=emr-tjug1" \
--applications "Name=HIVE"
```

Alternatywy

- Apache Ambari

Alternatywy

- Apache Ambari
- Apache Whirr

Alternatywy

- Apache Ambari
- Apache Whirr
- Apache CloudStack

Ekosystem

- Hadoop Streaming

Ekosystem

- Hadoop Streaming
- Hive

Ekosystem

- Hadoop Streaming
- Hive
- Pig

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark
- Storm

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark
- Storm
- R

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark
- Storm
- R
- Drill

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark
- Storm
- R
- Drill
- Mahout

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark
- Storm
- R
- Drill
- Mahout
- Sqoop

Ekosystem

- Hadoop Streaming
- Hive
- Pig
- Spark
- Storm
- R
- Drill
- Mahout
- Sqoop
- Hue

Hive

- Hive Query Language (a’la SQL)

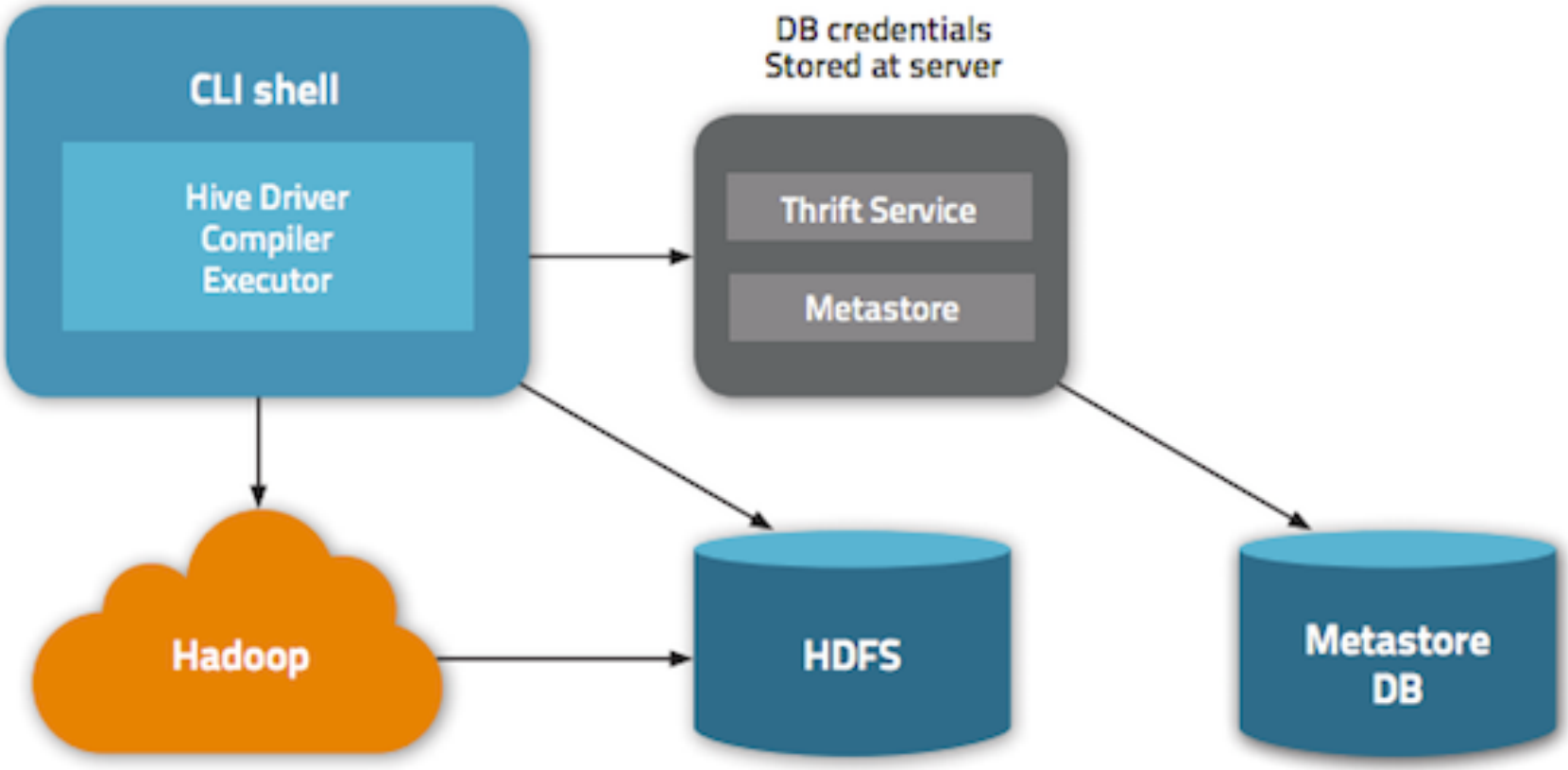
Hive

- Hive Query Language (a’la SQL)
- Metastore

Hive

- Hive Query Language (a’la SQL)
- Metastore
- JDBC

Hive



Źródło: Cloudera

Hive

```
show databases;
create database freebase;
use freebase;

create external table freebase (
subject string,
predicate string,
object string )
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t’
LINES TERMINATED BY '\012’
STORED AS TEXTFILE
LOCATION '/freebase’;

show tables;
describe formatted freebase;
```

Hive

```
select *
from freebase
where
subject like 'Albert Einstein’;

select object,collect_set(subject)
from freebase
where
object like '%Nobel %Prize %’
and
predicate = 'Award Won’
group by object;
```

Spark

- Resilient Distributed Datasets

Spark

- Resilient Distributed Datasets
- Java, Scala, Python

Spark

- Resilient Distributed Datasets
- Java, Scala, Python

- Spark SQL

Spark

- Resilient Distributed Datasets
- Java, Scala, Python

- Spark SQL
- Machine Learning Library (MLib)

Spark

- Resilient Distributed Datasets
- Java, Scala, Python

- Spark SQL
- Machine Learning Library (MLib)
- Spark Streaming

Spark

```
val tf = sc.textFile("hdfs://172.31.10.42:9000/freebase/facts.txt”)

tf.count()

tf.take(10).foreach(println)

val albert = tf.filter(line => line.contains("Albert Einstein")).collect()

albert.foreach(println)
```

Dziękuję!

Q&A

Arkadiusz Osiński
arkadiusz.osinski@allegrogroup.com
arek.osinski@i-netpl.info