Inspire…Educate…Transform.

Engineering Big Data

Spark (contd.), SQL on Hadoop

Dr. Sreerama KV Murthy
CEO, Quadratyx

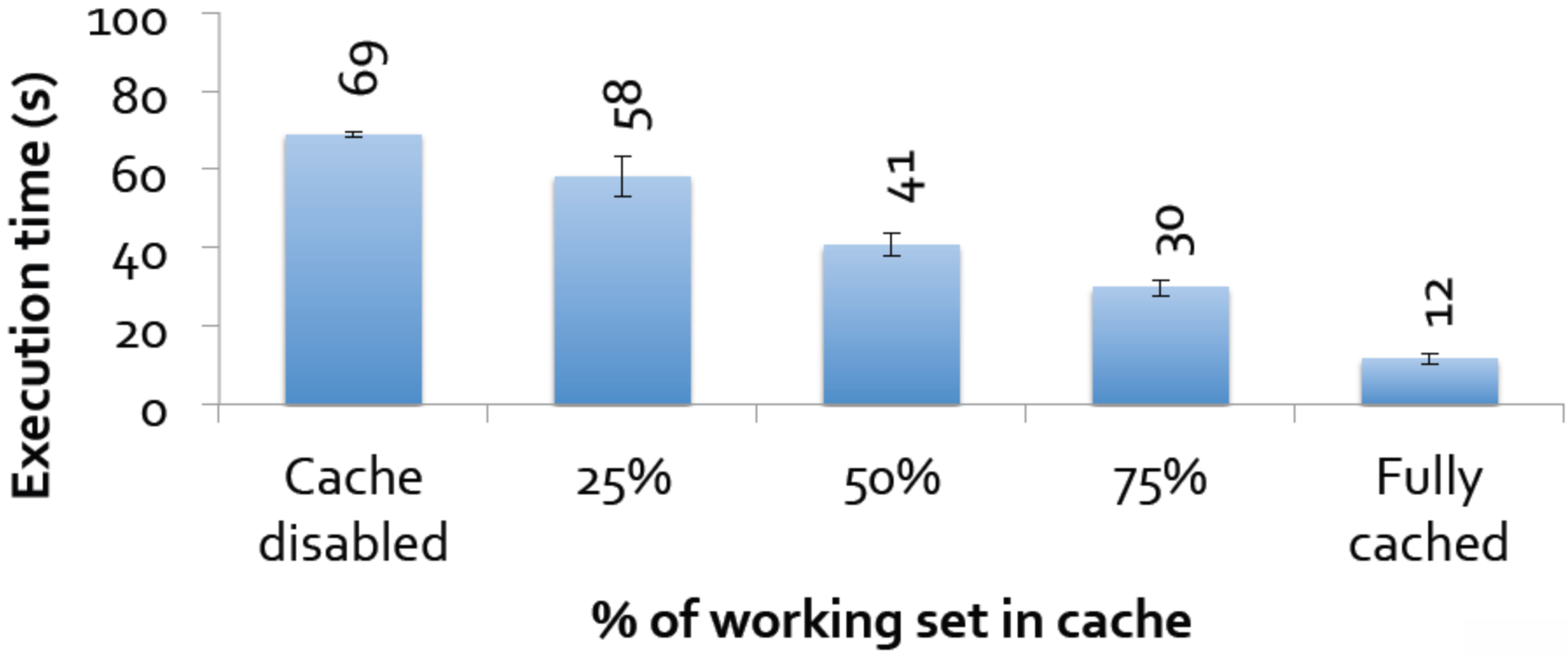
September 05, 2015

Wake-Up Quiz

Key Differences: Spark as against Map Reduce

- generalized patterns
  ⇒ unified engine for many use cases
- lazy evaluation of the lineage graph
  ⇒ reduces wait states, better pipelining
- generational differences in hardware
  ⇒ off-heap use of large memory spaces
- functional programming / ease of use
  ⇒ reduction in cost to maintain large apps
- lower overhead for starting jobs
- less expensive shuffles

Benefit of In-Memory Computing

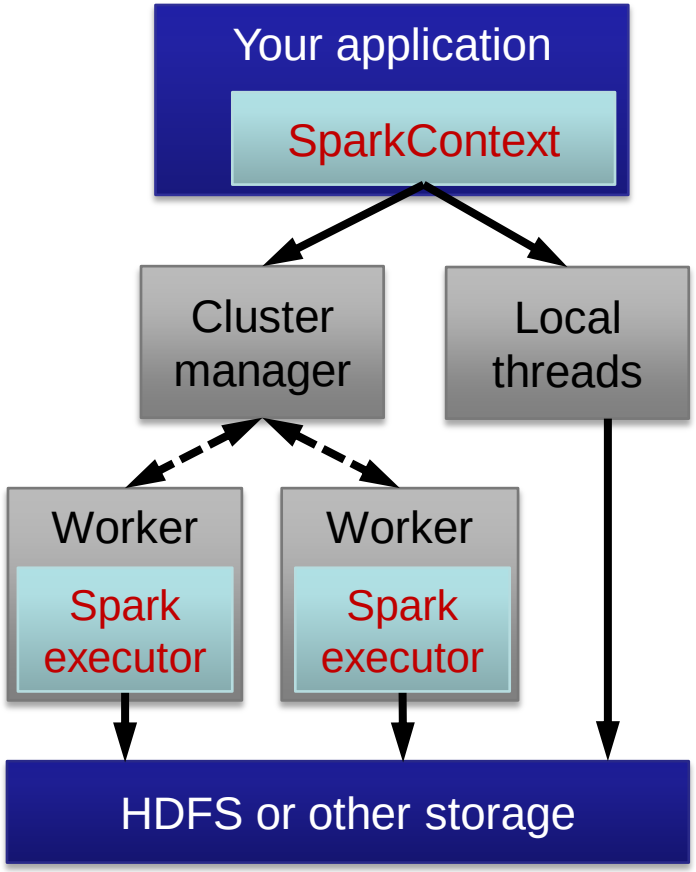


databricks.com/blog/2014/11/05/spark-officially-sets-a-new-record-in-large-scale-sorting.html

	Hadoop MR Record	Spark Record	Spark 1 PB
Data Size	102.5 TB	100 TB	1000 TB
Elapsed Time	72 mins	23 mins	234 mins
# Nodes	2100	206	190
# Cores	50400 physical	6592 virtualized	6080 virtualized
Cluster disk throughput	3150 GB/s (est.)	618 GB/s	570 GB/s
Sort Benchmark Daytona Rules	Yes	Yes	No
Network	dedicated data center, 10Gbps	virtualized (EC2) 10Gbps network	virtualized (EC2) 10Gbps network
Sort rate	**1.42 TB/min**	**4.27 TB/min**	**4.27 TB/min**
Sort rate/node	**0.67 GB/min**	**20.7 GB/min**	**22.5 GB/min**

Software Components

- Spark runs as a library in your program (one instance per app)
- Runs tasks locally or on a cluster
  - Standalone deploy cluster, Mesos or YARN
- Accesses storage via Hadoop InputFormat API
  - Can use HBase, HDFS, S3, …



Fault Recovery

RDDs track *lineage* information that can be used to efficiently recompute lost data

```
msgs = textFile.filter(lambda s: s.startsWith("ERROR"))
                .map(lambda s: s.split("\t")[2])
```

Language Support

Python

```
lines = sc.textFile(...)
lines.filter(lambda s: “ERROR” in s).count()
```

Scala

```
val lines = sc.textFile(...)
lines.filter(x => x.contains(“ERROR”)).count()
```

Java

```
JavaRDD<String> lines = sc.textFile(...);
lines.filter(new Function<String, Boolean>() {
  Boolean call(String s) {
    return s.contains(“error”);
  }
}).count();
```

Standalone Programs

- Python, Scala, & Java

Interactive Shells

- Python & Scala

Performance

- Java & Scala are faster due to static typing
- …but Python is often fine

Interactive Shell

- The Fastest Way to Learn Spark
- Available in Python and Scala
- Runs as an application on an existing Spark Cluster…
- OR Can run locally

Spark Logistic Regression Code

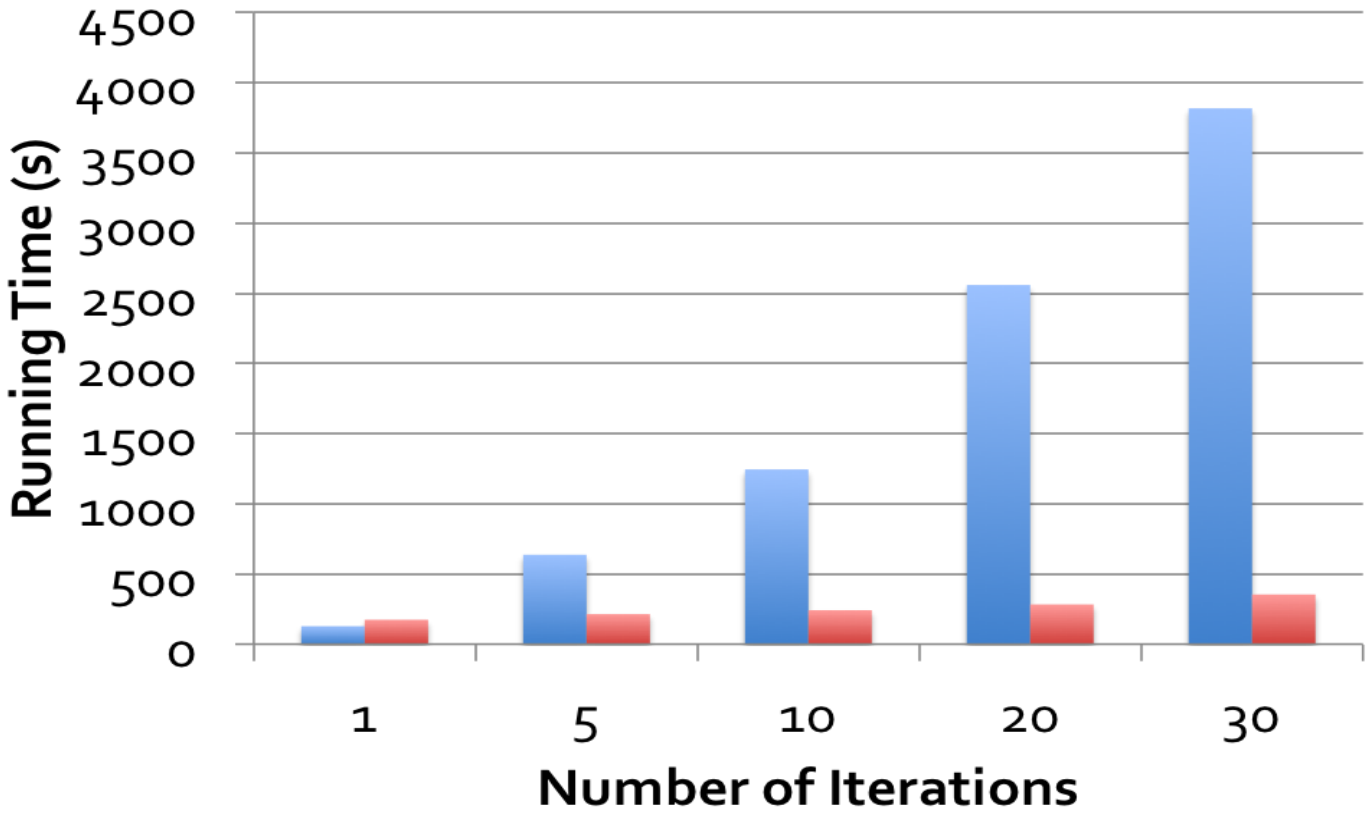
```
val data = spark.textFile(...).map(readPoint).cache()

var w = Vector.random(D)

for (i <- 1 to ITERATIONS) {
  val gradient = data.map(p =>
    (1 / (1 + exp(-p.y*(w dot p.x))) - 1) * p.y * p.x
  ).reduce(_ + _)
  w -= gradient
}

println("Final w: " + w)
```

Logistic Regression Performance

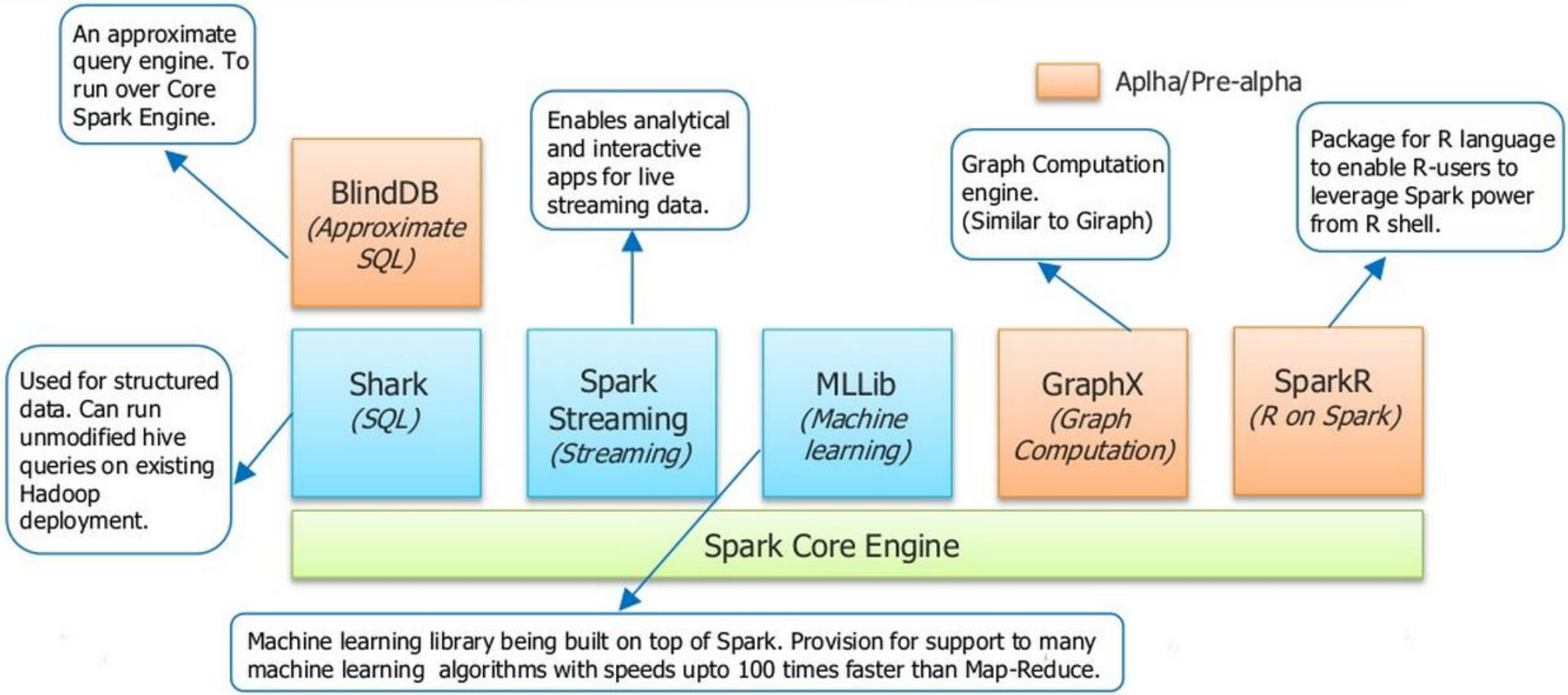


127 s / iteration

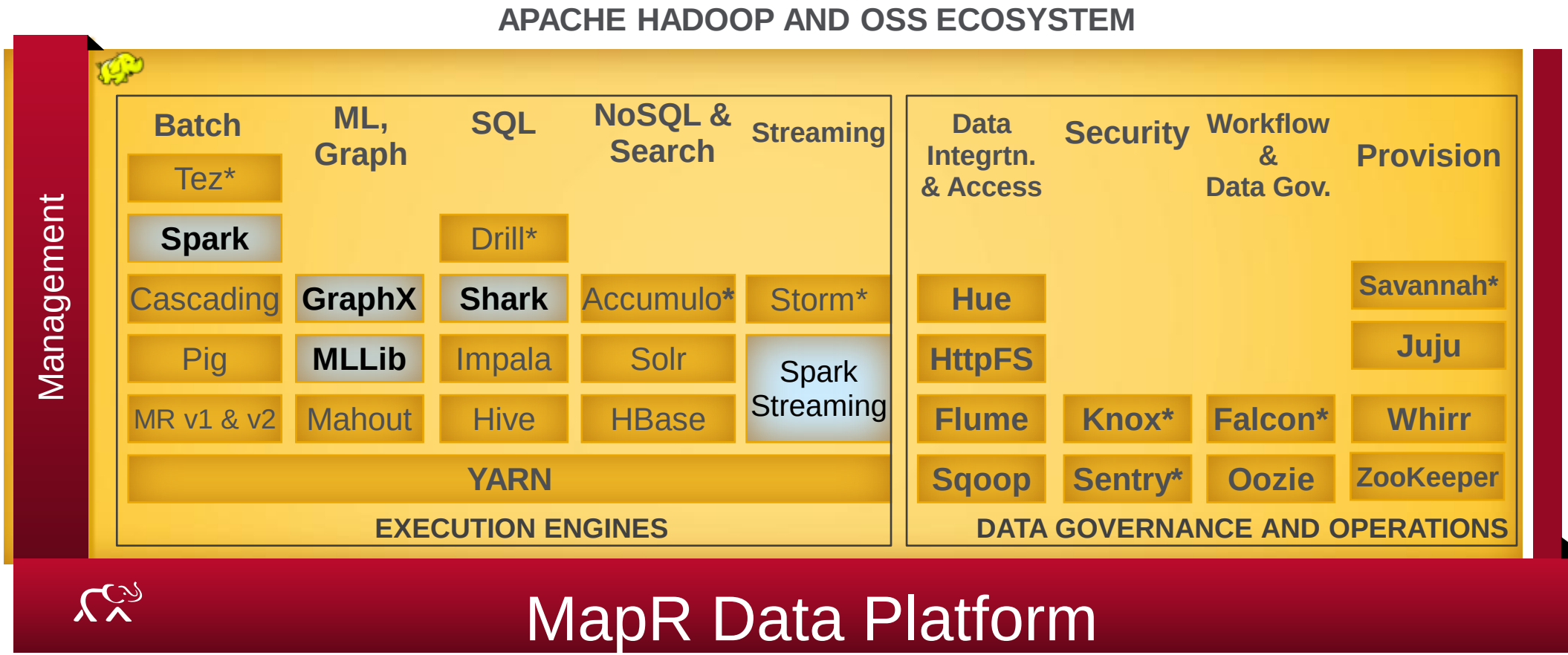
Hadoop

Spark

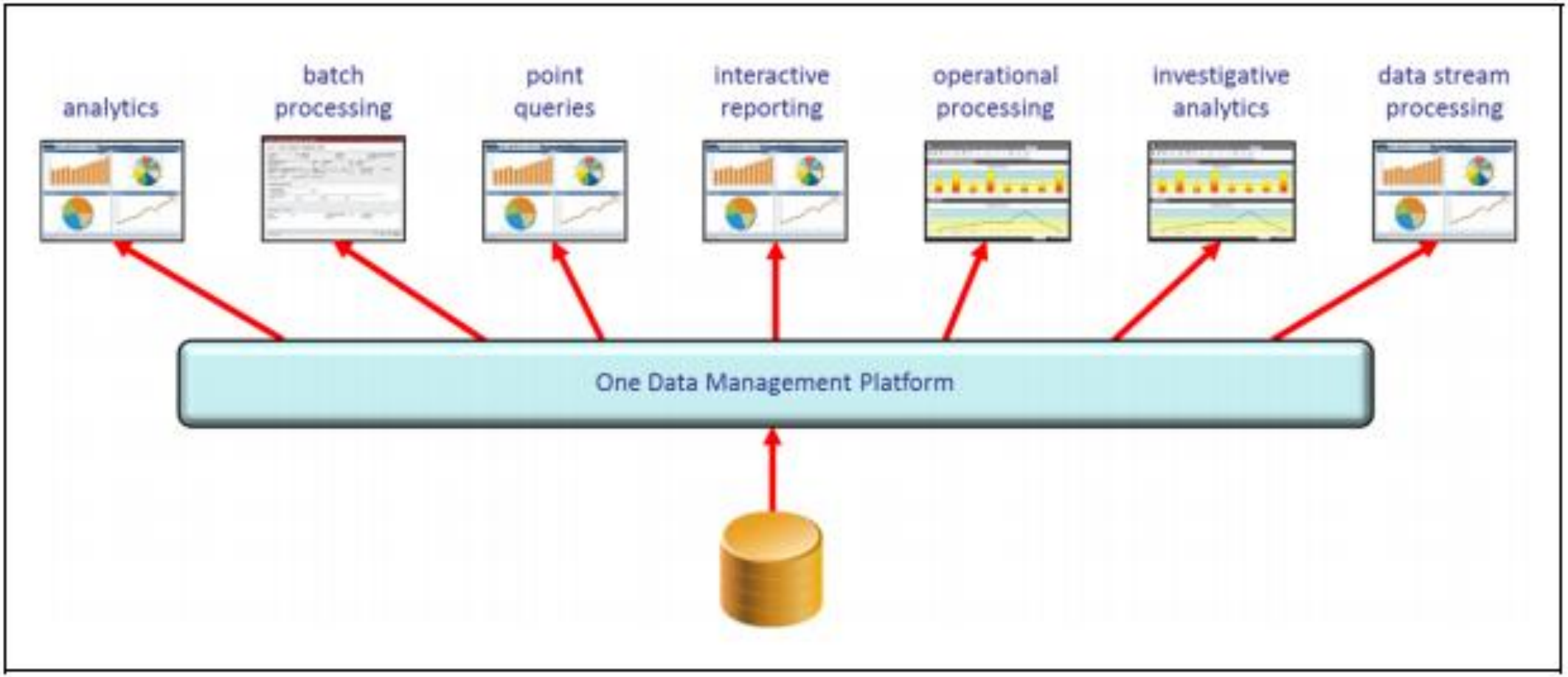
first iteration 174 s
further iterations 6 s


An approximate query engine. To run over Core Spark Engine.
BlindDB (Approximate SQL)
Enables analytical and interactive apps for live streaming data.
Aplha/Pre-alpha
Graph Computation engine. (Similar to Giraph)
Package for R language to enable R-users to leverage Spark power from R shell.
Used for structured data. Can run unmodified hive queries on existing Hadoop deployment.
Shark (SQL)
Spark Streaming (Streaming)
MLLib (Machine learning)
GraphX (Graph Computation)
SparkR (R on Spark)
Spark Core Engine
Machine learning library being built on top of Spark. Provision for support to many machine learning algorithms with speeds upto 100 times faster than Map-Reduce.

The Complete Spark Stack on Hadoop



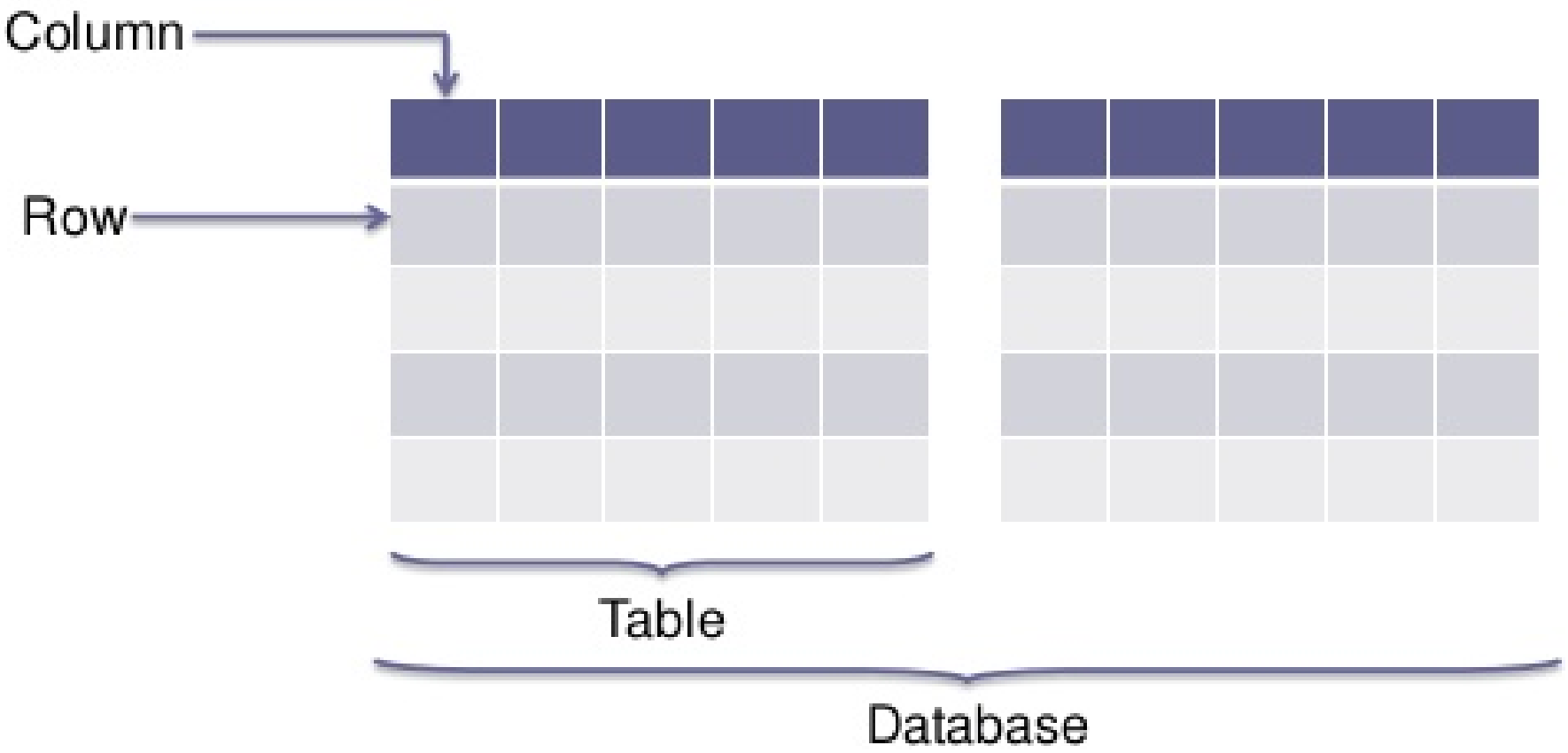
SQL ON HADOOP


analytics
batch processing
point queries
interactive reporting
operational processing
investigative analytics
data stream processing
One Data Management Platform

Relational Databases

A 1-MIN PRIMER

Basic Terminology



Relational Databases - Fundamentals

- Data is organized into tables: rows & columns
- Each row represents an instance of an entity
- Each column represents an attribute of an entity
- Metadata describes each table column
- Relationships between entities are represented by values stored in the columns of the corresponding tables (keys)
- Accessible through Standard Query Language (SQL)

Metadata & Data Table

Organism

Name	Type	Max Length	Description
Name	Alphanumeric	100	Organism name
Size	Integer	10	Genome length (bases)
Gc	Float	5	Percent GC
Accession	Alphanumeric	10	Accession number
Release	Date	8	Release date
Center	Alphanumeric	100	Genome center name
Sequence	Alphanumeric	Variable	Sequence

Name	Size	Gc	Accession	Release	Center	Sequence
Escherichia coli K12	4,640,000	50	NC_000913	09/05/1997	Univ. Wisconsin	AGCTTTTCATT…
Streptococcus pneumoniae R6	2,040,000	40	NC_003098	09/07/2001	Eli Lilly and Company	TTGAAAGAAAA…
…						

SQL

- ANSI (American National Standards Institute) standard computer language for accessing and manipulating database systems.
- SQL statements are used to retrieve and update data in a database.
- Includes:
  - Data Manipulation Language (DML)
  - Data Definition Language (DDL)

SQL Examples

DDL

```
CREATE DATABASE Microbial;

CREATE TABLE Organism (
    Name varchar(100)
    Size int(10)
    Gc   decimal(5)
    Accession varchar(10)
    Release date(8)
    Center varchar(100));

ALTER TABLE Organism ADD
    Sequence varchar;
DROP TABLE Organism;
```

DML

```
SELECT *
FROM Organism, Gene
WHERE
    Organism.Name="Escherichia coli
    K12" AND
    Organism.Accession=Gene.OAcces
    sion AND Gene.Start>=1,000,000
    AND Gene.End<=2,000,000
```

Database Management Systems

- Examples:
  - Proprietary: MS Access, MS SQL Server, DB2, Oracle, Sybase
  - Open source: MySql, PostgreSQL
- Advantages:
  - Program-data independence
  - Minimal data redundancy
  - Improved data consistency & quality
    - Access control
    - Transaction control
  - Improved accessibility & data sharing
  - Increased productivity of application development
  - Enforced standards

SQL on Hadoop: Hive and Variants

HIVE

```
class Mapper {
  buffer

  map(key, number) {
      buffer.append(number)
      if (buffer.is_full) {
          max = compute_max(buffer)
          emit(1, max)
      }
  }
}


class Reducer {
  reduce(key, list_of_local_max) {
      global_max = 0
      for local_max in list_of_local_max {
          if local_max > global_max {
              global_max = local_max
          }
      }         emit(1, global_max)
  }
}


class Combiner {
  combine(key, list_of_local_max) {
      local_max = maximum(list_of_local_max)
      emit(1, local_max)
  }
}
```

What is this code doing?

Can we do it more simply?


HIVE

Hive and Pig: Common Idea

- Provide higher-level language to facilitate large-data processing
- Higher-level language "compiles down" to Hadoop jobs

Hive

- Hive: data warehousing application in Hadoop
  - Query language is **HQL**, variant of SQL
  - Tables stored as **HDFS flat files**
  - Developed by **Facebook**, now open source

```
-- Delete the tables if they already exist
DROP TABLE myinput;
DROP TABLE wordcount;
CREATE TABLE myinput (line STRING);

-- Load the text from the local (Linux) filesystem.
-- This should be changed to HDFS for any serious usage
LOAD DATA LOCAL INPATH '/user/someperson/mytext.txt' INTO TABLE
myinput;

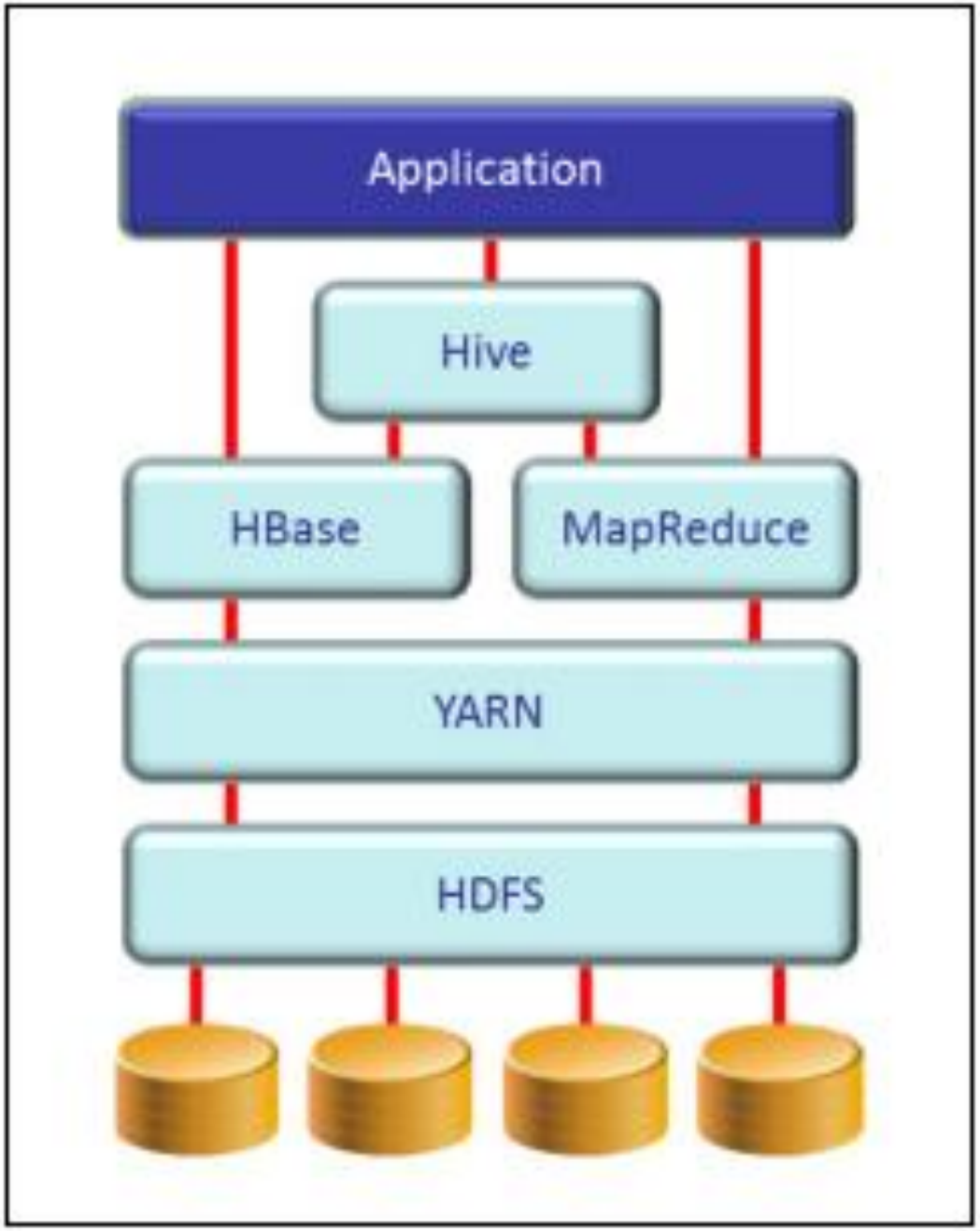
-- Create a table with the words grouped and counted.
CREATE TABLE wordcount AS
SELECT word, count(1) AS count
GROUP BY word

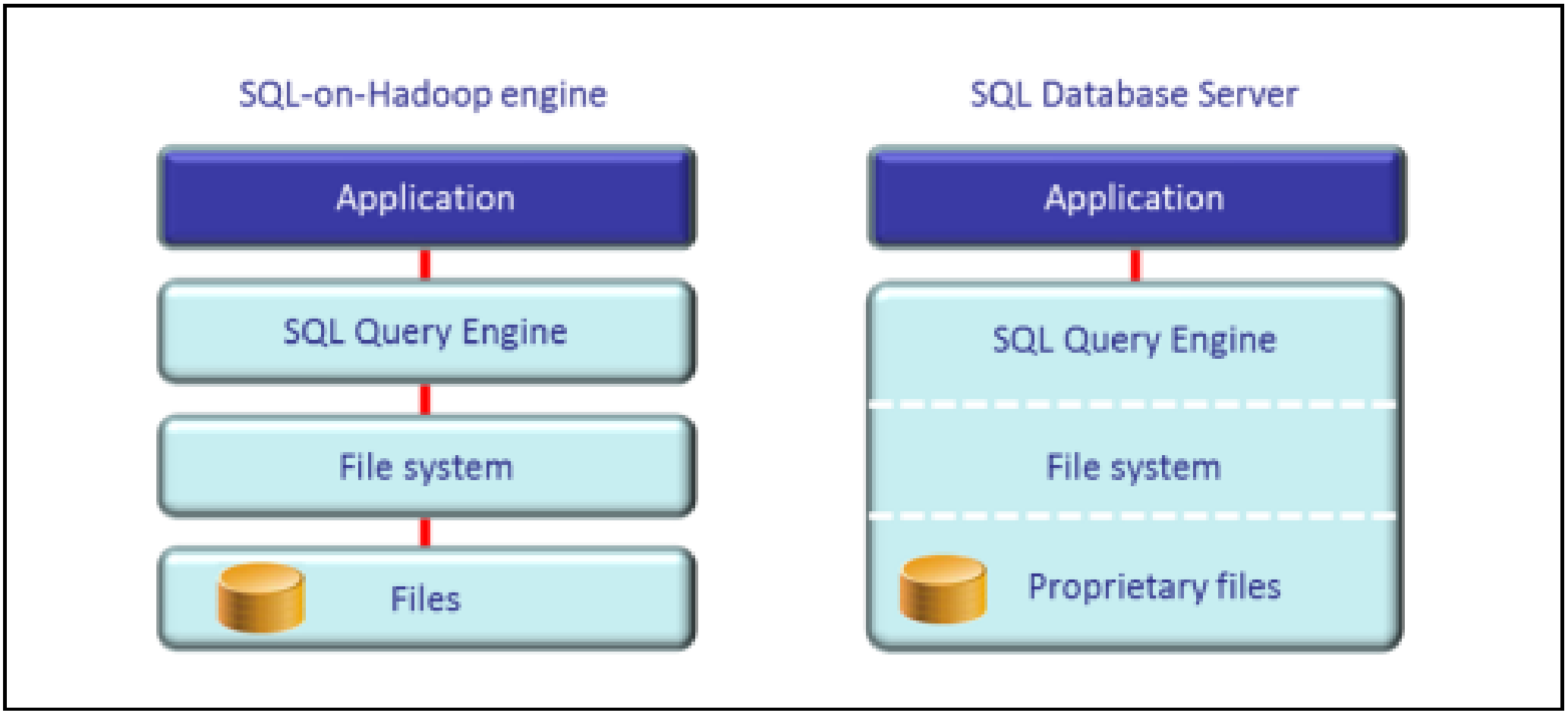
-- Sort the output by count with the highest counts first
ORDER BY count DESC, word ASC;
```

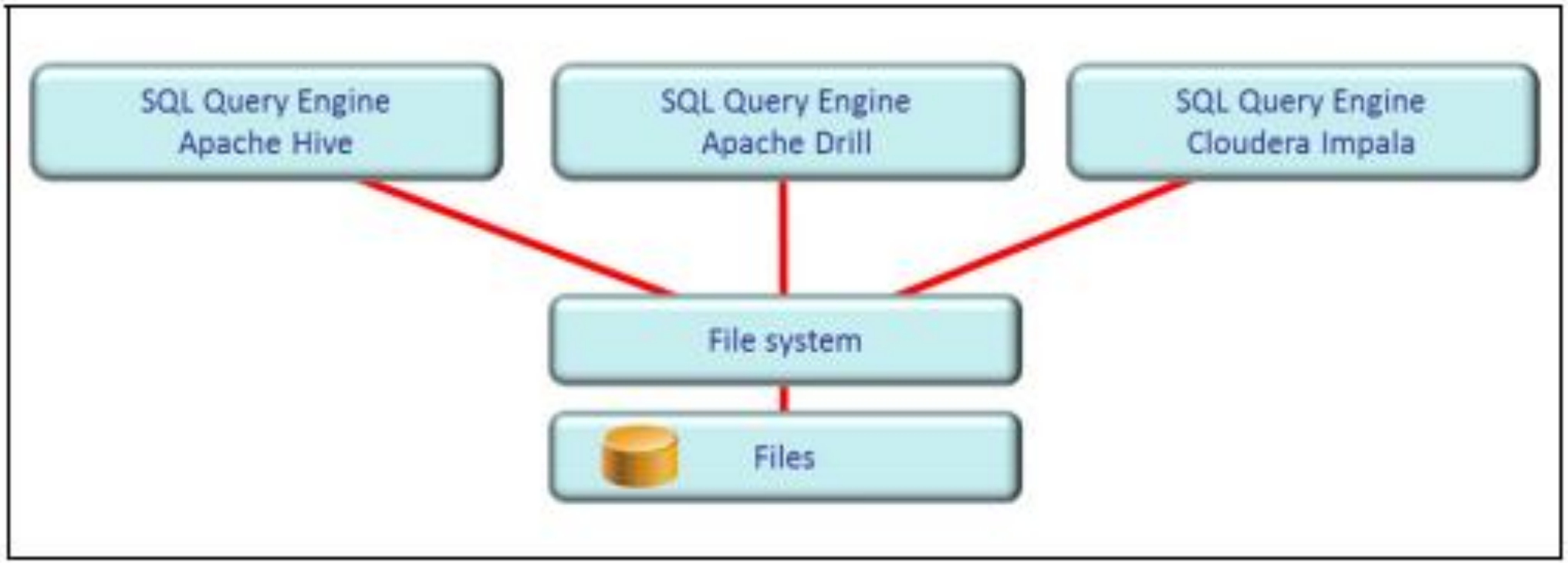
Simple WordCount Example
https://github.com/slimandslam/pig-hive-wordcount/blob/master/wordco unt.hql

Real-World Use Cases of Hive

- Bizo: reporting and ad hoc queries
- Chitika: data mining and analysis
- CNET: data mining, log analysis and ad hoc queries
- Digg: data mining, log analysis, R&D, reporting/analytics
- Grooveshark: user analytics, dataset cleaning, machine learning R&D
- Hi5: analytics, machine learning, social graph analysis.
- HubSpot: near real-time web analytics.
- Last.fm: various ad hoc queries.
- Trending Topics: log data normalization and building sample data sets for trend detection R&D.
- VideoEgg: analyze all the usage data


Application
Hive
HBase
MapReduce
YARN
HDFS


SQL-on-Hadoop engine
Application
SQL Query Engine
File system
Files
SQL Database Server
Application
SQL Query Engine
File system
Proprietary files


SQL Query Engine
Apache Hive
SQL Query Engine
Apache Drill
SQL Query Engine
Cloudera Impala
File system
Files

SQOOP

- RDBMS-Hadoop interoperability is key to Enterprise Hadoop adoption
- SQOOP provides a good general purpose tool for transferring data between any JDBC database and Hadoop
- SQOOP extensions can provide optimizations for specific targets

Sqoop

Implemented in Map-Reduce

- Import & Export
- ODBC, JDBC Data Sources
- CSV Files in HDFS

SQOOP export

- Read files in HDFS directory via MapReduce
- Bulk parallel insert into database table

SQOOP import

- Divide table into ranges using primary key max/min
- Create mappers for each range
- Mappers write to multiple HDFS nodes
- Creates text or sequence files
- Generates Java class for resulting HDFS file
- Generates Hive definition and auto-loads into HIVE

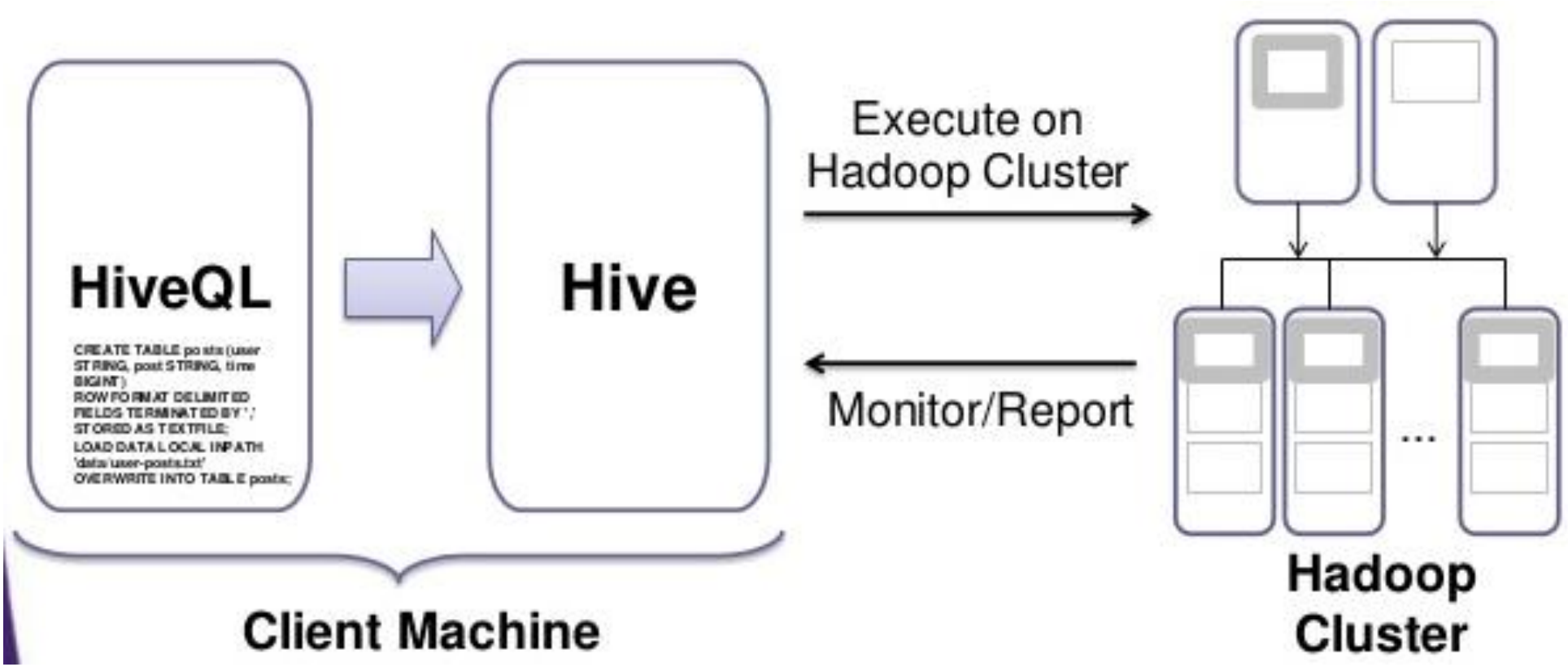
SQOOP details

- SQOOP features:
  - Compatible with almost any JDBC enabled database
  - Auto load into HIVE
  - HBase support
  - Job management
  - Cluster configuration (jar file distribution)
  - WHERE clause support
  - Open source, and included in most Hadoop distributions

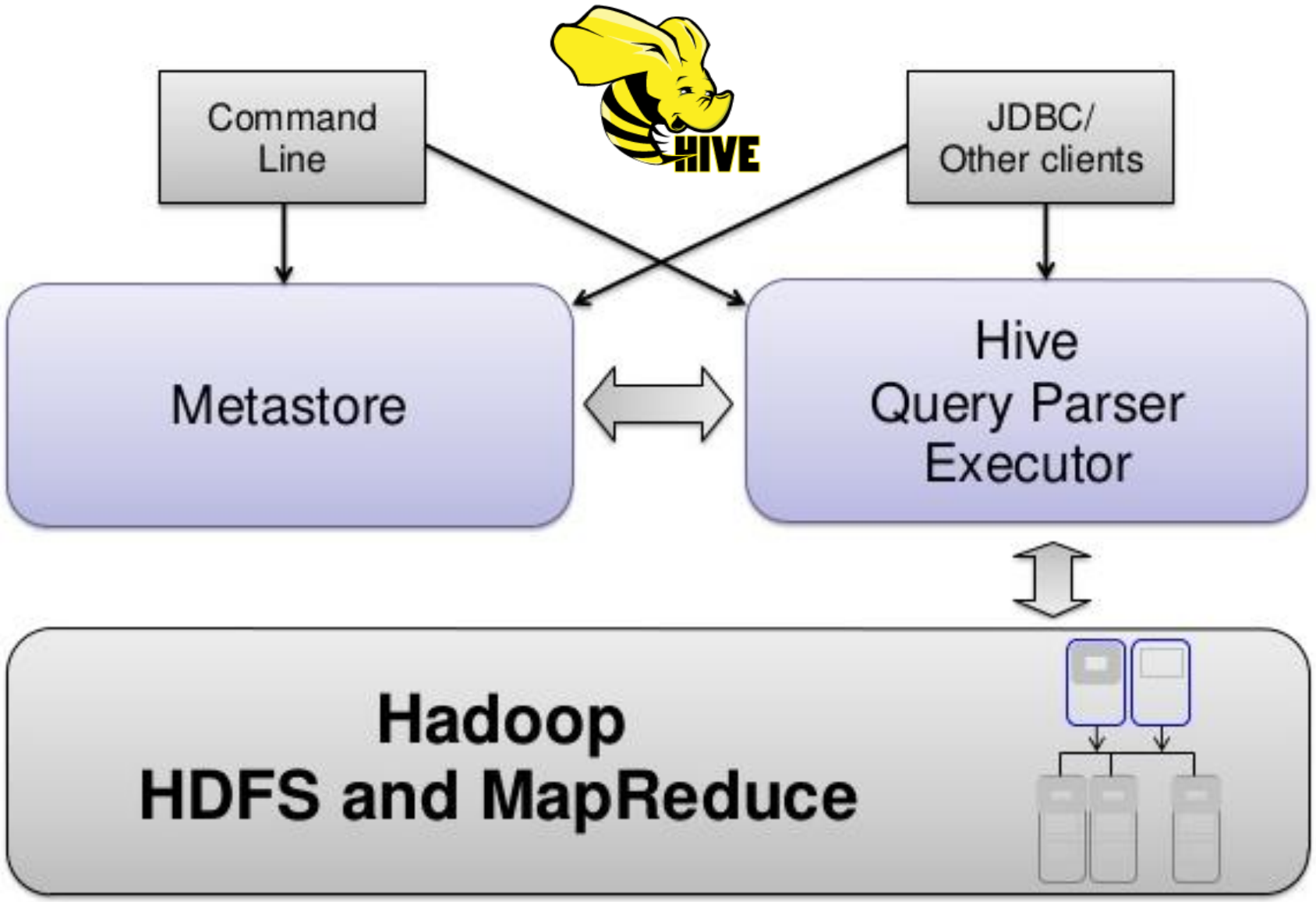
- SQOOP fast paths & plug ins


HIVE

Hive Execution Model: Translate to Map-Reduce



Hive Architecture



Hive Metastore

- To reduce the time to perform semantic checks during query execution, Hive keeps its metadata in a relational data store.
- Apache Derby, a light weight embedded SQL database
  - Can be easily replaced with Apace MySQL
- Schema not shared between users – each user has their own instance of Derby.
- Stored in Hive opening directory /metastore_db

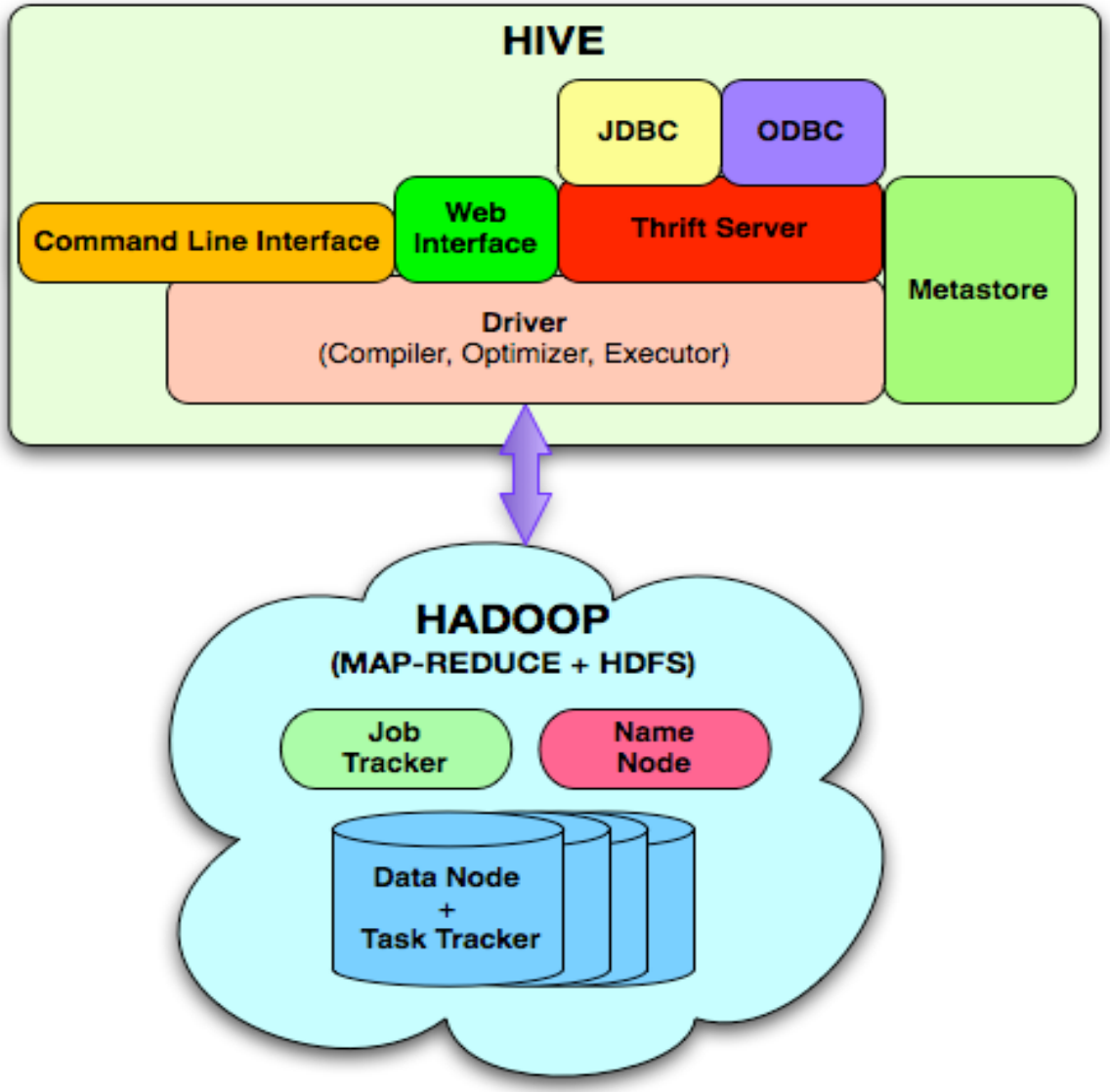
Data model

- Hive structures data into well-understood database concepts such as: tables, rows, cols, partitions
- It supports primitive types: integers, floats, doubles, and strings
- Hive also supports:
  - associative arrays: map<key-type, value-type>
  - Lists: list<element type>
  - Structs: struct<file name: file type…>
- SerDe: serialize and deserialize API is used to move data in and out of tables

Data Storage

- Tables are logical data units; table metadata associates the data in the table to hdfs directories.
- HDFS namespace: tables (HDFS directory), partition (HDFS subdirectory), buckets (subdirectories within partition)
- E.g., /user/hive/warehouse/test_table is a HDFS directory

Hive architecture



*from Facebook original paper

Hive Architectural Elements

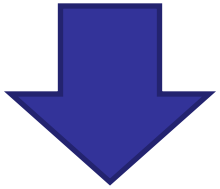
- Metastore: stores system catalog
- Driver: manages life cycle of HiveQL query as it moves through HIVE; also manages session handle and session statistics
- Query compiler: Compiles HiveQL into a directed acyclic graph of map/reduce tasks
- Execution engines: Executes the tasks in proper dependency order; interacts with Hadoop
- HiveServer: provides Thrift interface and JDBC/ODBC for integrating other applications.
- Client components: CLI, web interface, jdbc/odbc inteface
- Extensibility interface include SerDe, User Defined Functions and User Defined Aggregate Function.

List of Hive built-in and user-defined functions

HTTPS://CWIKI.APACHE.ORG/CONFLUENCE/DISPLAY/HIVE/LANGUAGEMANUAL+UDF

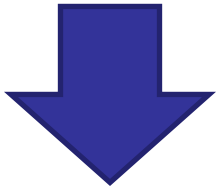
Hive: Behind the Scenes

```
SELECT s.word, s.freq, k.freq FROM shakespeare s
  JOIN bible k ON (s.word = k.word) WHERE s.freq >= 1 AND k.freq >= 1
  ORDER BY s.freq DESC LIMIT 10;
```



(Abstract Syntax Tree)


```
(TOK_QUERY (TOK_FROM (TOK_JOIN (TOK_TABREF shakespeare s) (TOK_TABREF
bible k) (= (. (TOK_TABLE_OR_COL s) word) (. (TOK_TABLE_OR_COL k) word))))
(TOK_INSERT (TOK_DESTINATION (TOK_DIR TOK_TMP_FILE)) (TOK_SELECT
(TOK_SELEXPR (. (TOK_TABLE_OR_COL s) word)) (TOK_SELEXPR (.
(TOK_TABLE_OR_COL s) freq)) (TOK_SELEXPR (. (TOK_TABLE_OR_COL k) freq)))
(TOK_WHERE (AND (>= (. (TOK_TABLE_OR_COL s) freq) 1) (>= (. (TOK_TABLE_OR_COL
k) freq) 1))) (TOK_ORDERBY (TOK_TABSORTCOLNAMEDESC (. (TOK_TABLE_OR_COL s)
freq))) (TOK_LIMIT 10)))
```



(one or more of MapReduce jobs)

Hive: Behind the Scenes… 2

```
STAGE DEPENDENCIES:
  Stage-1 is a root stage
  Stage-2 depends on stages: Stage-1
  Stage-0 is a root stage

STAGE PLANS:
  Stage: Stage-1
    Map Reduce
      Alias -> Map Operator Tree:
        s
          TableScan
            alias: s
            Filter Operator
              predicate:
                  expr: (freq >= 1)
                  type: boolean
              Reduce Output Operator
                key expressions:
                      expr: word
                      type: string
                sort order: +
                Map-reduce partition
columns:
                      expr: word
                      type: string
                tag: 0
                value expressions:
                      expr: freq
                      type: int
                      expr: word
                      type: string
        k
```

```
TableScan
        alias: k
        Filter Operator
          predicate:
              expr: (freq >= 1)
              type: boolean
          Reduce Output Operator
            key expressions:
                  expr: word
                  type: string
            sort order: +
            Map-reduce partition columns:
                  expr: word
                  type: string
            tag: 1
            value expressions:
                  expr: freq
                  type: int
```

Word Count in Hive using external UDFs

```
FROM (
MAP doctext USING 'python wc_mapper.py'
AS (word, cnt)
FROM docs
CLUSTER BY word
)
REDUCE word, cnt USING 'pythonwc_reduce.py';
```

SQL is Cool!

Batch jobs via Map Reduce

Apache Hive

✓ Fault Tolerance
✓ Scales to Petabytes
✓ Schema Flexibility

Transactional Database on Hbase?

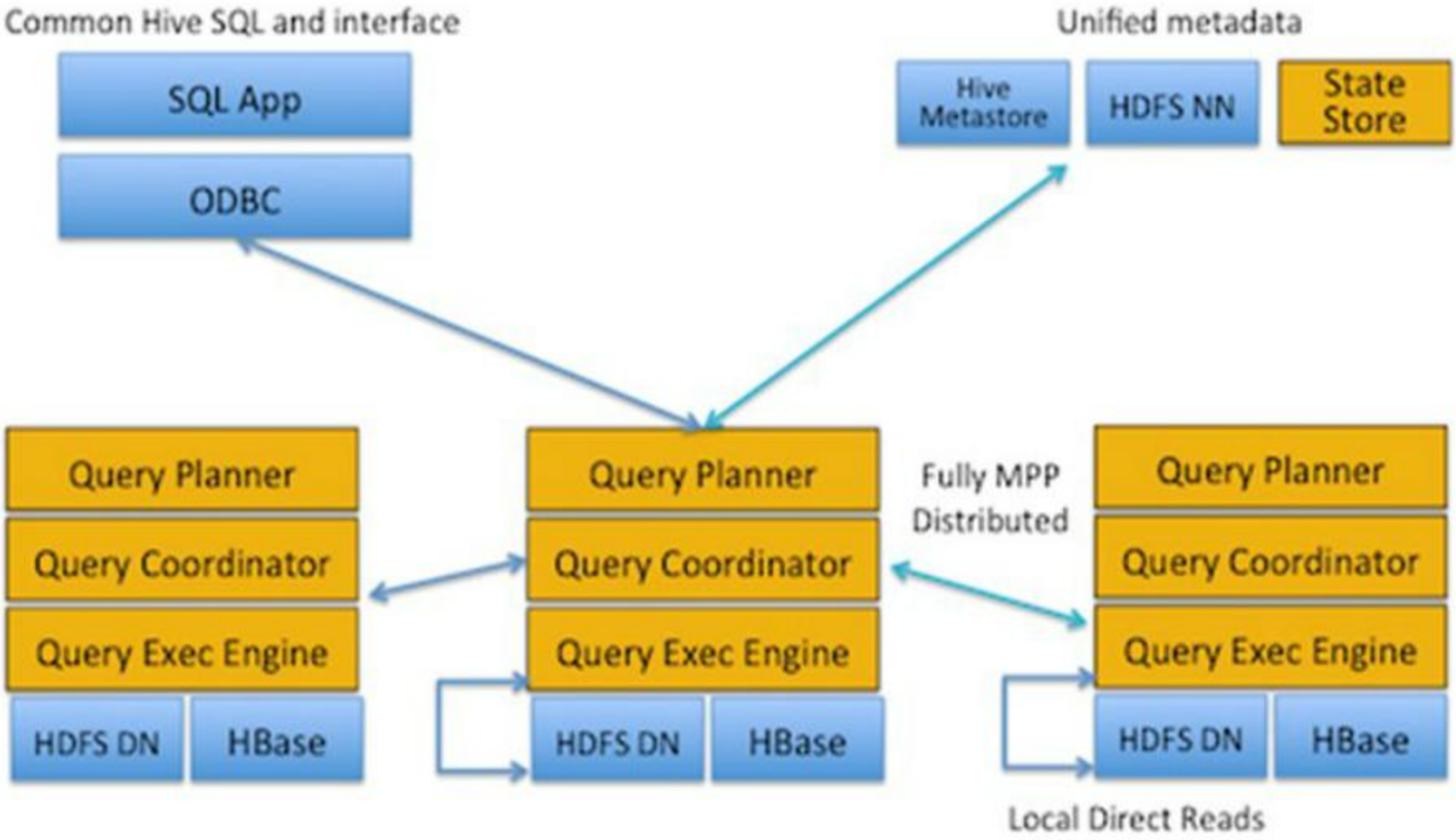
Unproven

Real-time query response
On Data Warehouse

- Cloudera Impala
- Apache
  - Drill (MapR)
  - Presto (Facebook)
  - Shark/Spark (UC Berkeley AMPLab)
  - Stinger initiative and Tez (Hortonworks)
- IBM Big SQL
- Pivotal HAWQ
- NEWSQL

? Fault Tolerance
? Scale to Petabytes
? Schema Flexibility

Cloudera Impala



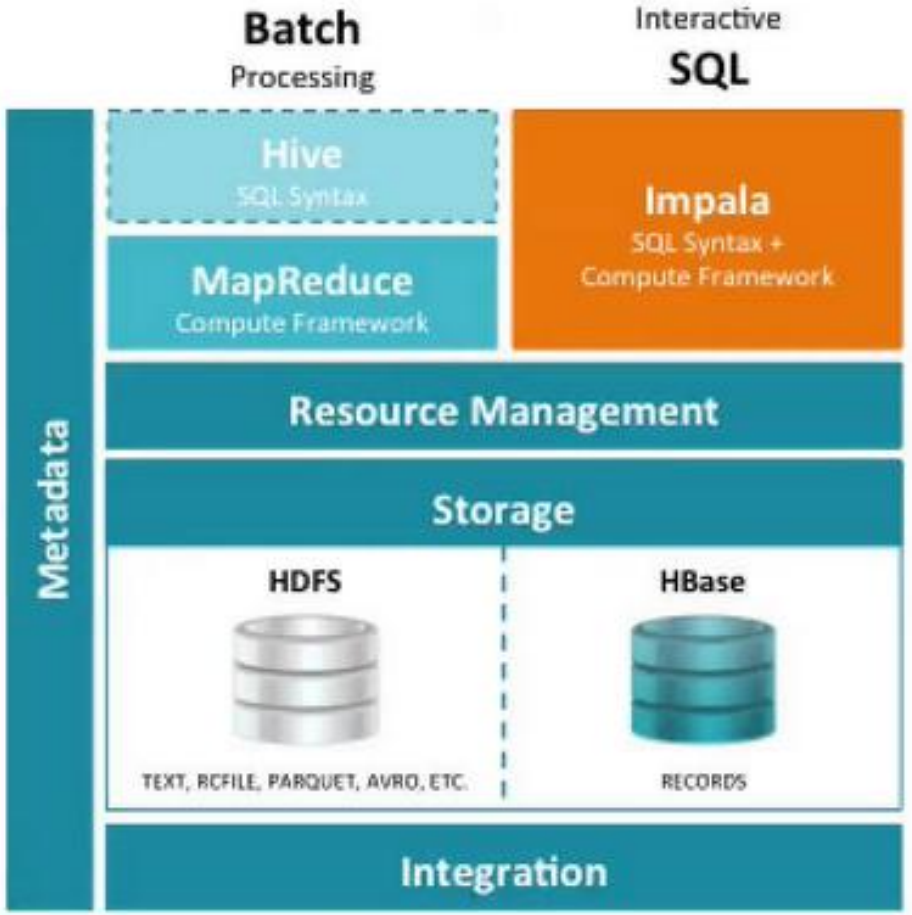
Impala and Hive

Shares Everything Client-Facing

- Metadata (table definitions)
- ODBC/JDBC drivers
- SQL syntax (Hive SQL)
- Flexible file formats
- Machine pool
- Hue GUI

But Built for Different Purposes

- **Hive:** runs on MapReduce and ideal for batch processing
- **Impala:** native MPP query engine ideal for interactive SQL



User Interfaces

- REST
- CLI (command line)
- Native API
- JDCB / ODBC

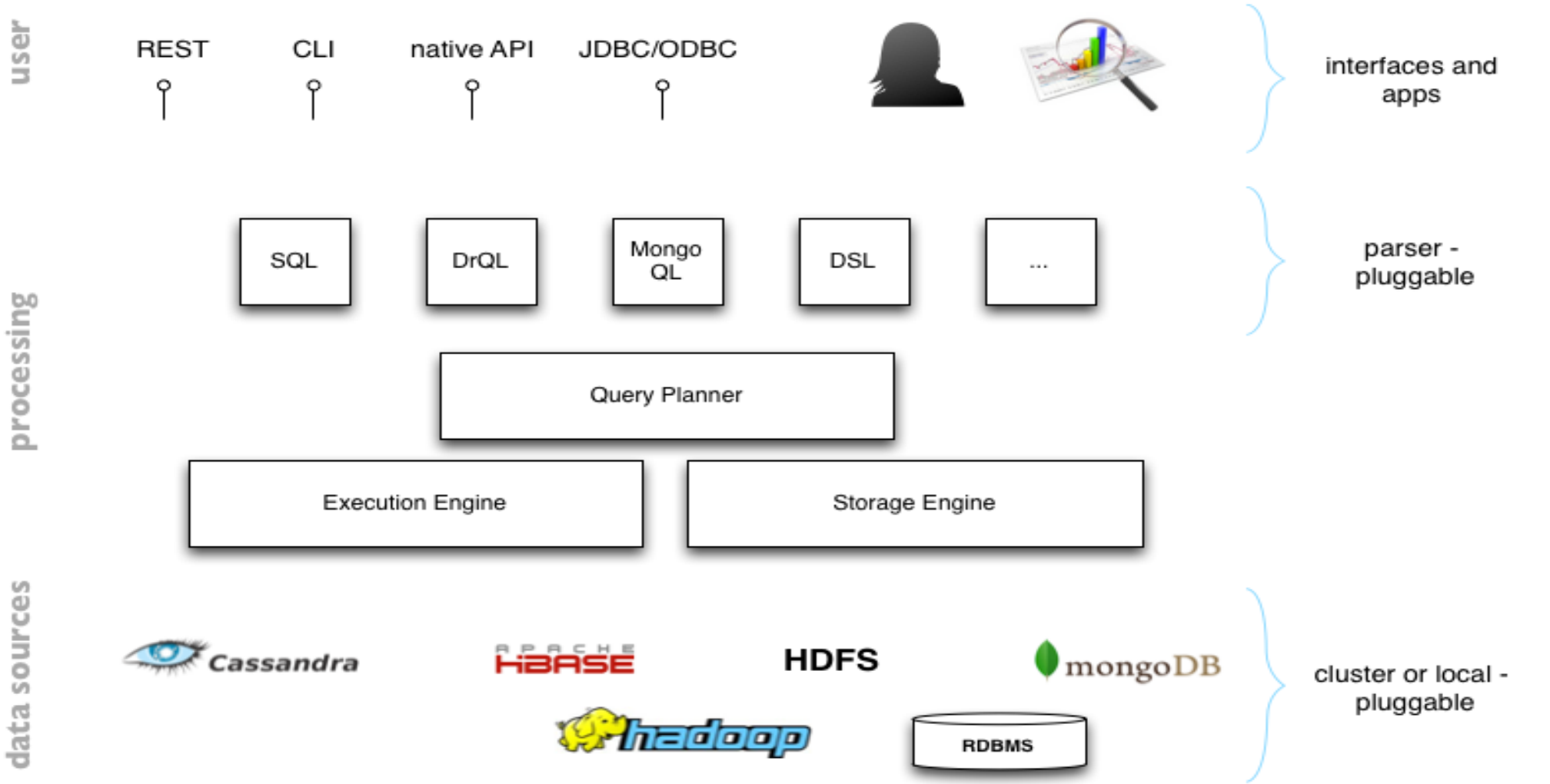
Processing Interfaces

- SQL
- DrQL (Drill query language)
- MongoQL
- ...

Data Interfaces

- Cassandra
- Hbase
- Hadoop HDFS
- MongoDB
- RDBMS
- (pluggable data sources)

Apache Drill – Architecture



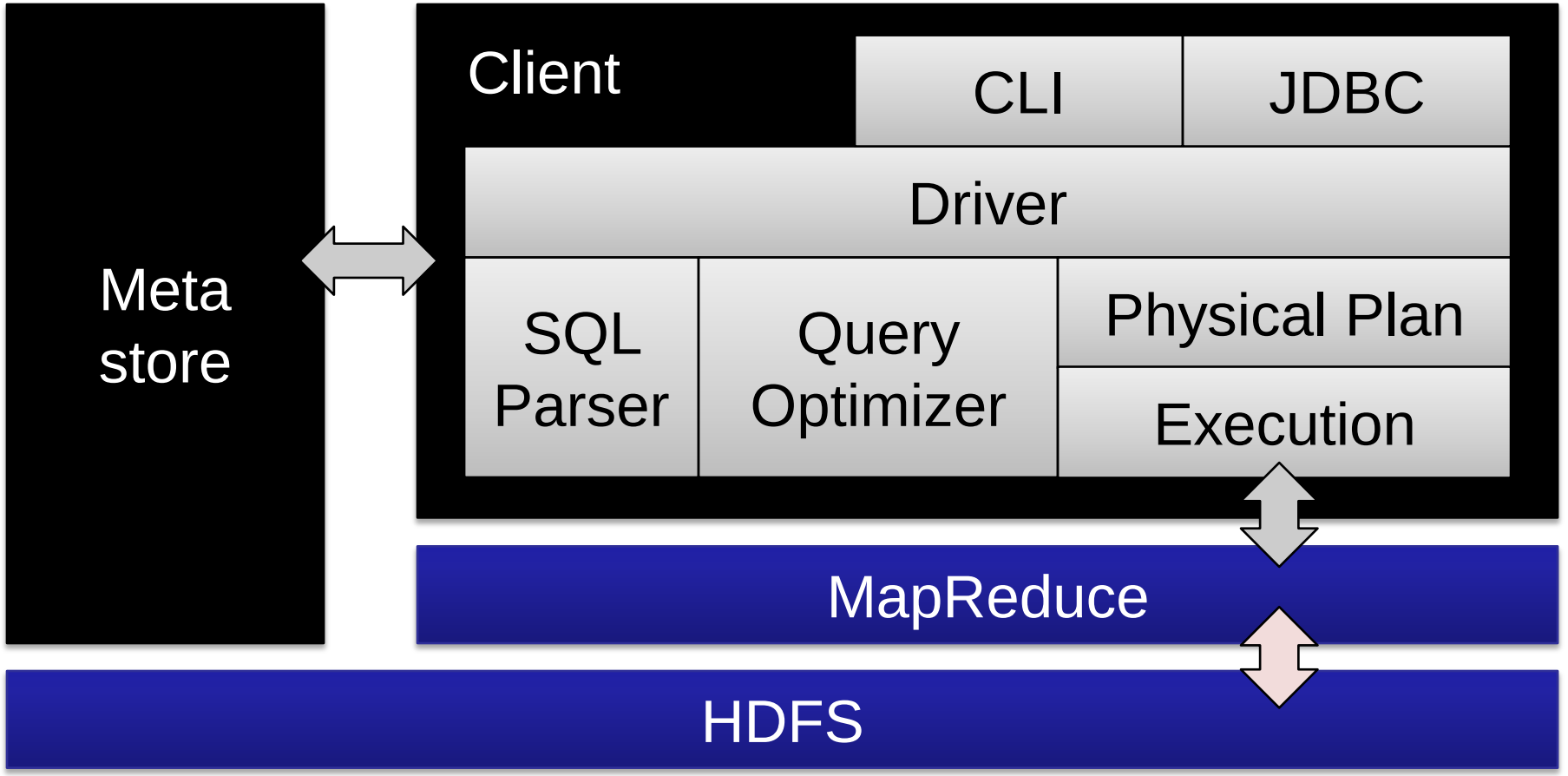
Sample Drill Query

Cross data source queries

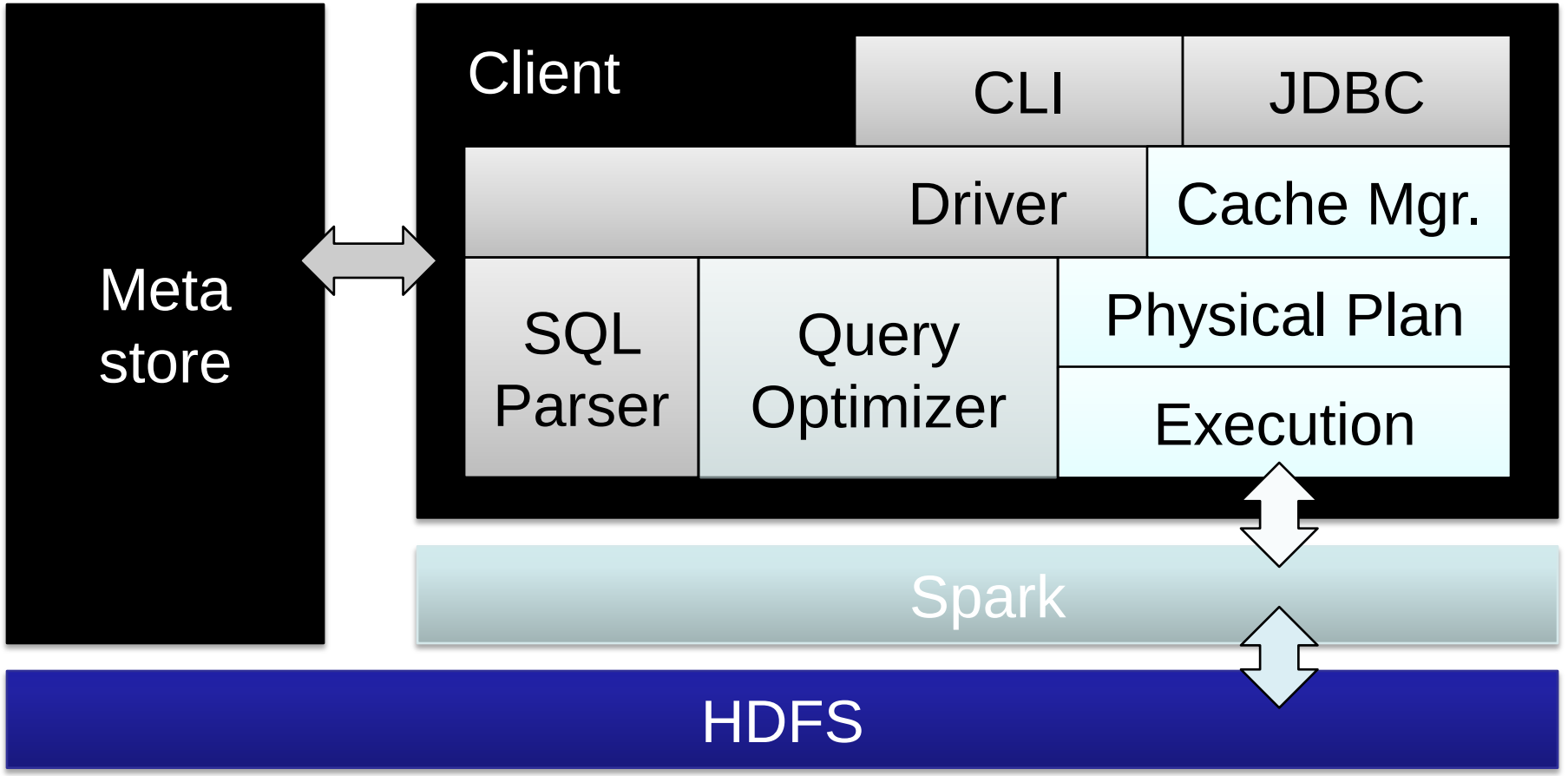
- Combine data from Files, HBase, Hive in one query
- No central metadata definitions necessary
- Example:
  - `USE HiveTest.CustomersDB`
  - `SELECT Customers.customer_name, SocialData.Tweets.Count`
    `FROM Customers`
    `JOIN HBaseCatalog.SocialData SocialData`
    `ON Customers.Customer_id = Convert_From(SocialData.rowkey, UTF-8)`

SPARK SQL

Hive Architecture



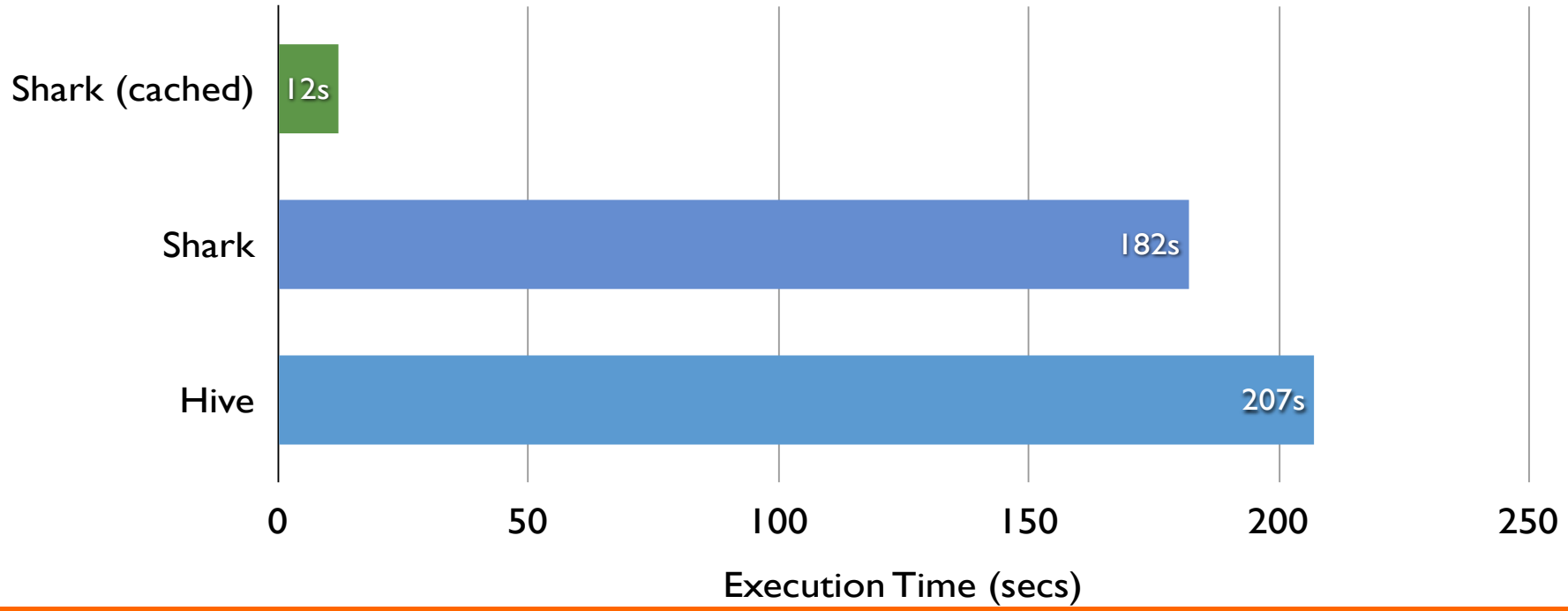
SQL on Spark Architecture



[Engle et al, SIGMOD 2012]

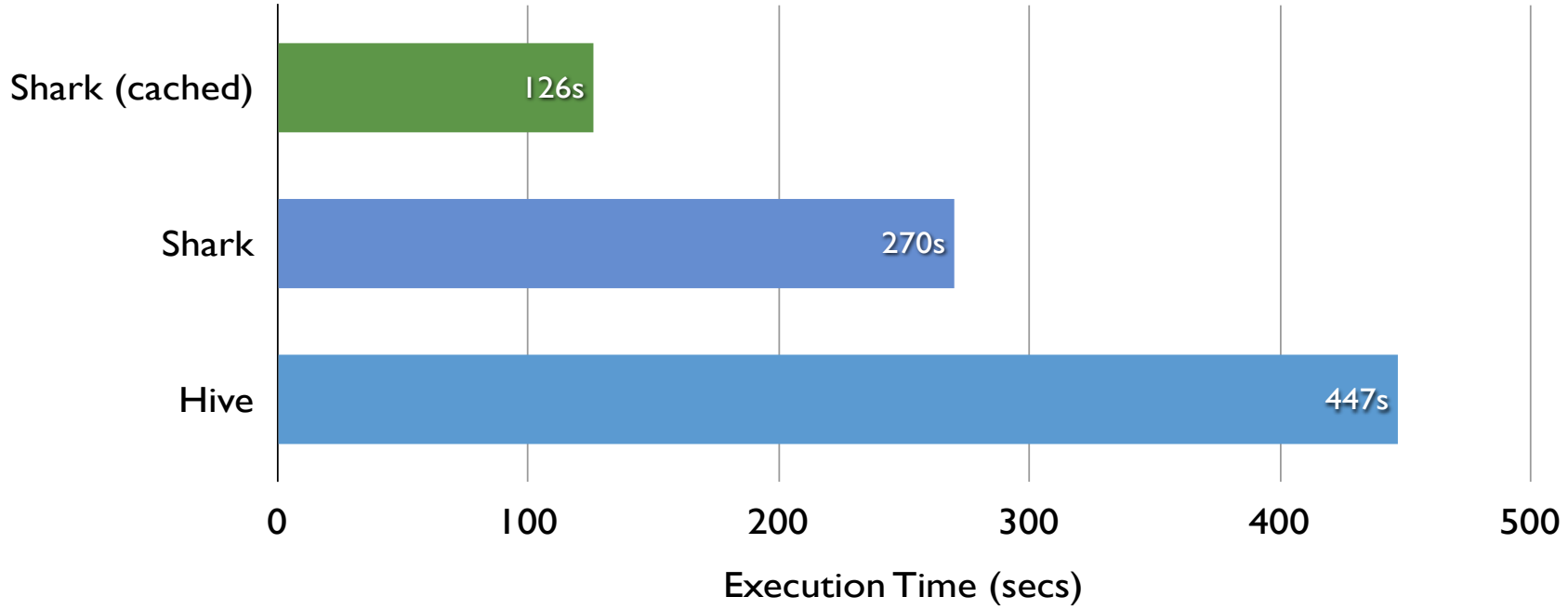
Benchmark Query 1

```
SELECT * FROM grep WHERE field LIKE '%XYZ%';
```

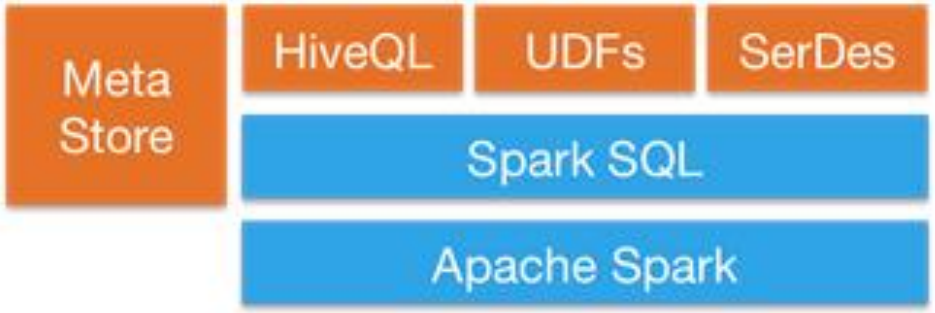


Benchmark Query 2

- ```
  SELECT sourceIP, AVG(pageRank), SUM(adRevenue) AS earnings
  FROM rankings AS R, userVisits AS V ON R.pageURL = V.destURL
  WHERE V.visitDate BETWEEN '1999-01-01' AND '2000-01-01'
  GROUP BY V.sourceIP
  ORDER BY earnings DESC
  LIMIT 1;
  ```


  ```

Spark SQL



Spark SQL can use existing Hive metastores, SerDes, and UDFs.

Use your existing BI tools to query big data.

Inspire...Educate...Transform.

International School of Engineering

Plot 63/A, 1st Floor, Road # 13, Film Nagar, Jubilee Hills, Hyderabad - 500 033

For Individuals: +91-9502334561/63 or 040-65743991

For Corporates: +91-9618483483

Web: http://www.insofe.edu.in

Facebook: https://www.facebook.com/insofe

Twitter: https://twitter.com/Insofeedu

YouTube: http://www.youtube.com/InsofeVideos

SlideShare: http://www.slideshare.net/INSOFE

LinkedIn: http://www.linkedin.com/company/international-school-of-engineering