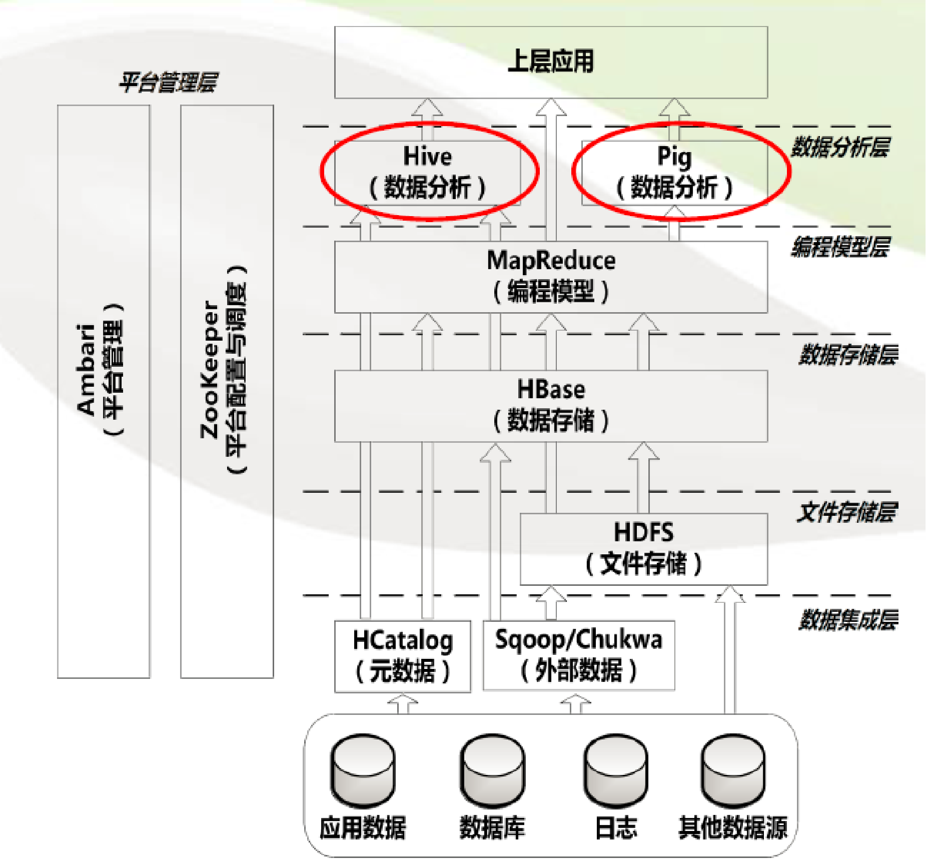
议程

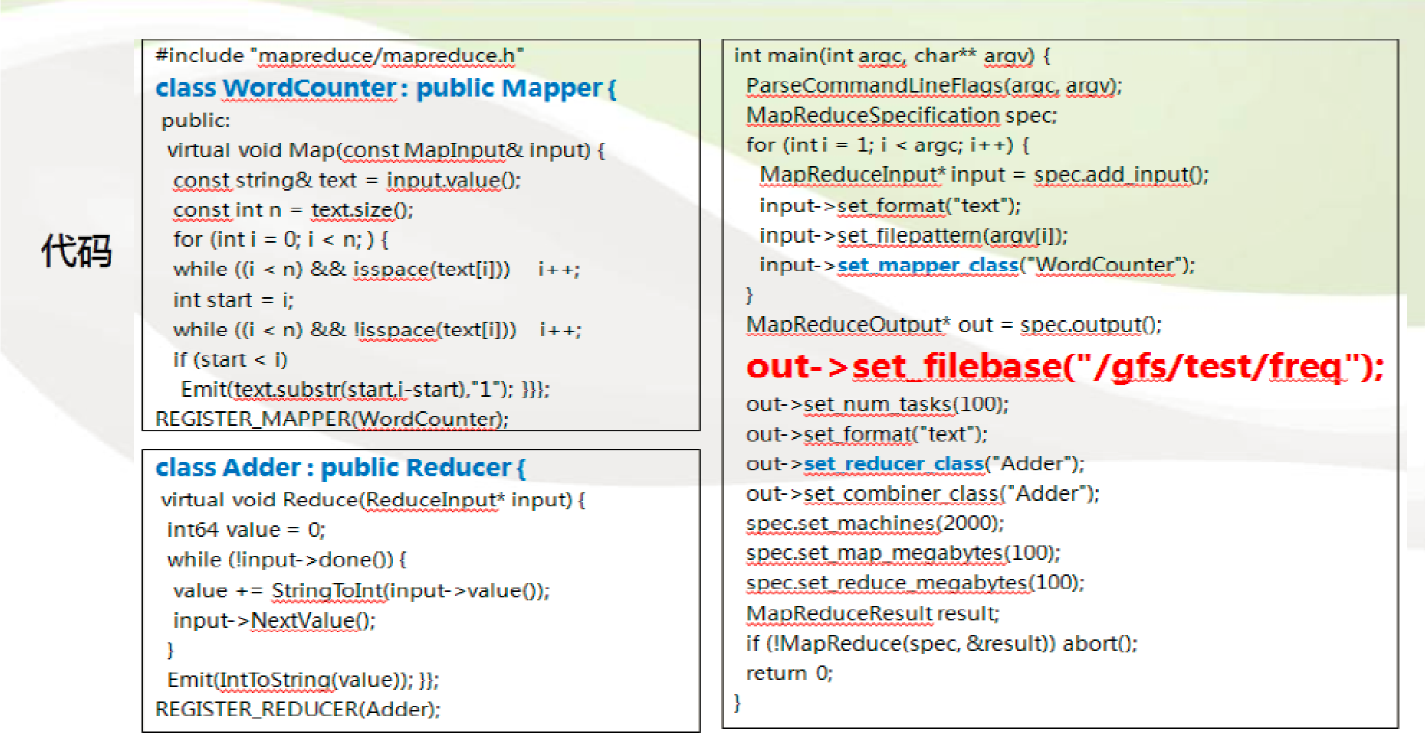
- What is Hive?
- Hive Components
- What is Hive Data Model?
- Underlying Hive Architecture
- Using Hive in Practice

What is Hive



Data warehouse infrastructure build on top of Hadoop for querying and managing large data sets

Hive存在的必要性



代码

```
#include "mapreduce/mapreduce.h"
class WordCounter : public Mapper {
 public:
  virtual void Map(const MapInput& input) {
   const string& text = input.value();
   const int n = text.size();
   for (int i = 0; i < n; ) {
   while ((i < n) && isspace(text[i]))    i++;
   int start = i;
   while ((i < n) && !isspace(text[i]))    i++;
   if (start < i)
    Emit(text.substr(start,i-start),"1"); }}};
REGISTER_MAPPER(WordCounter);
```

```
class Adder : public Reducer {
 virtual void Reduce(ReduceInput* input) {
  int64 value = 0;
  while (!input->done()) {
   value += StringToInt(input->value());
   input->NextValue();
  }
  Emit(IntToString(value)); }};
REGISTER_REDUCER(Adder);
```

```
int main(int argc, char** argv) {
  ParseCommandLineFlags(argc, argv);
  MapReduceSpecification spec;
  for (int i = 1; i < argc; i++) {
    MapReduceInput* input = spec.add_input();
    input->set_format("text");
    input->set_filepattern(argv[i]);
    input->set_mapper_class("WordCounter");
  }
  MapReduceOutput* out = spec.output();
  out->set_filebase("/gfs/test/freq");
  out->set_num_tasks(100);
  out->set_format("text");
  out->set_reducer_class("Adder");
  out->set_combiner_class("Adder");
  spec.set_machines(2000);
  spec.set_map_megabytes(100);
  spec.set_reduce_megabytes(100);
  MapReduceResult result;
  if (!MapReduce(spec, &result)) abort();
  return 0;
}
```

Hive

```
SELECT * FROM log WHERE date > '2012-12-01' ;
```

Hive的核心设计理念

A system for managing and querying unstructured data as if it were structured

- Stores schema in Database
- Uses Map-Reduce for execution
- HDFS for Storage

Hive能做什么

- **Designed for OLAP**
- **SQL type language for querying**
- **It is familiar, fast, scalable, and extensible**

Hive不适合做什么

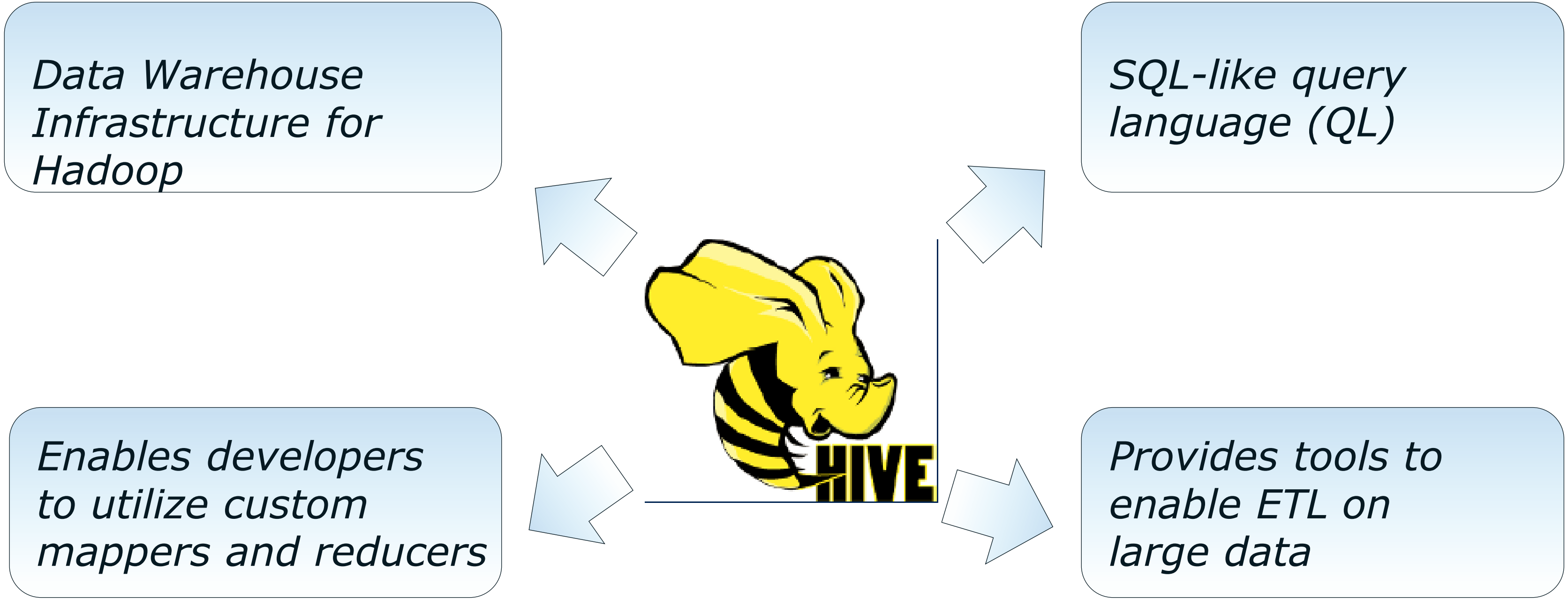
- **Relational database**
- **Designed for Online Transaction Processing (OLTP)**
- **Language for real-time queries and row-level updates**

Hive的历史

- **Early Hive development work started at Facebook in 2007**
- **Hive is an Apache project under Hadoop**

http://hive.apache.org

Hive特性



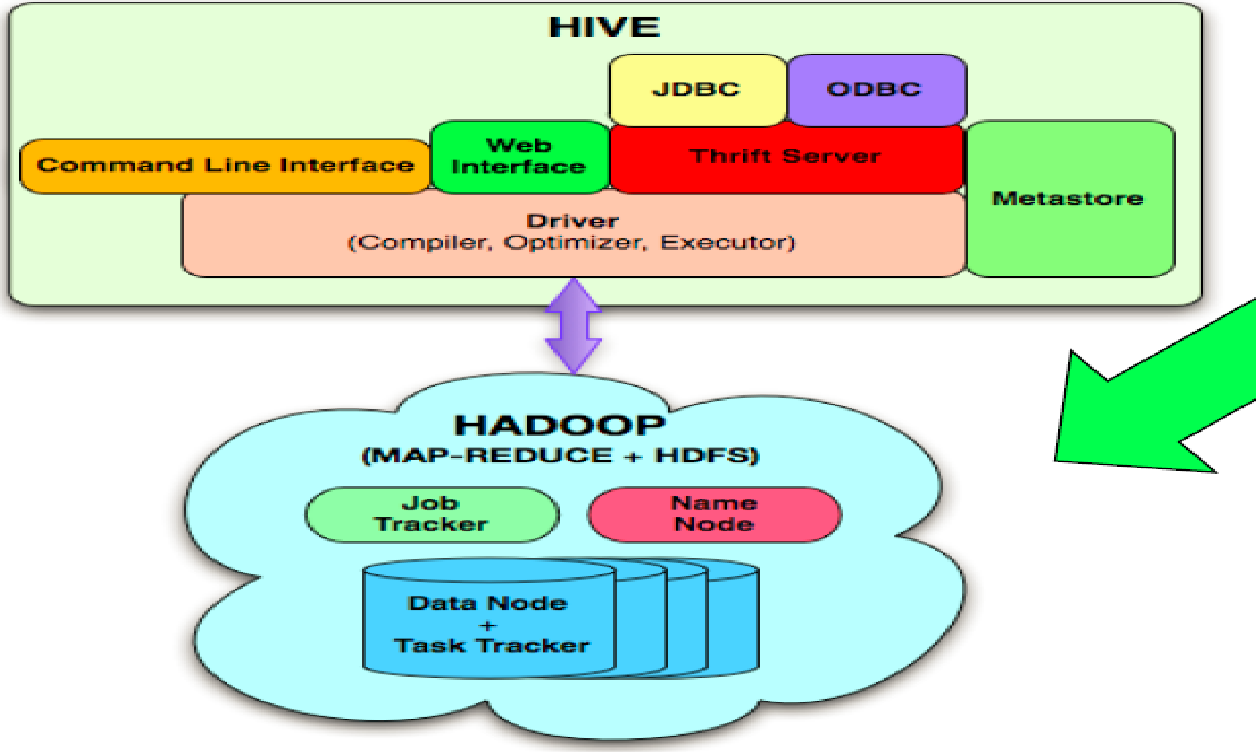
Data Warehouse Infrastructure for Hadoop

SQL-like query language (QL)

Enables developers to utilize custom mappers and reducers

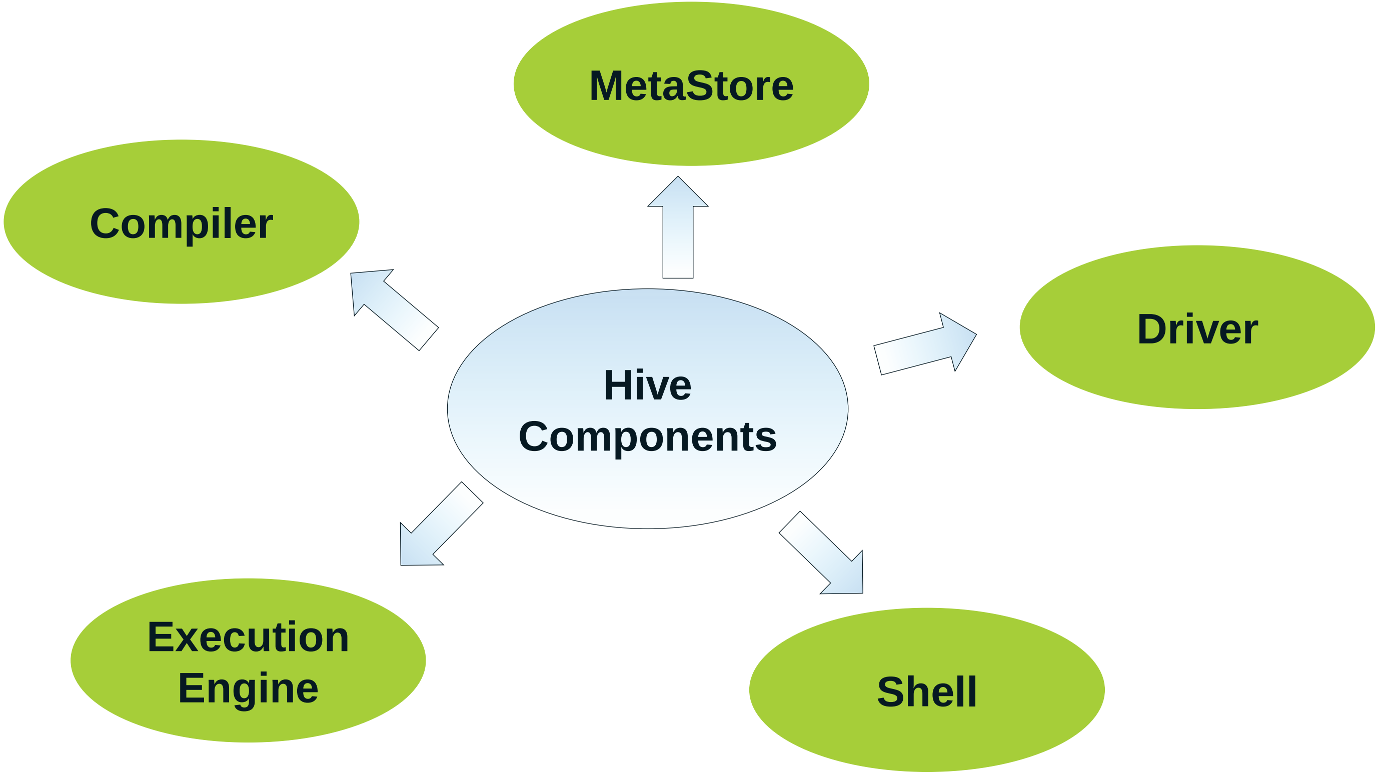
Provides tools to enable ETL on large data

Hive架构和组件

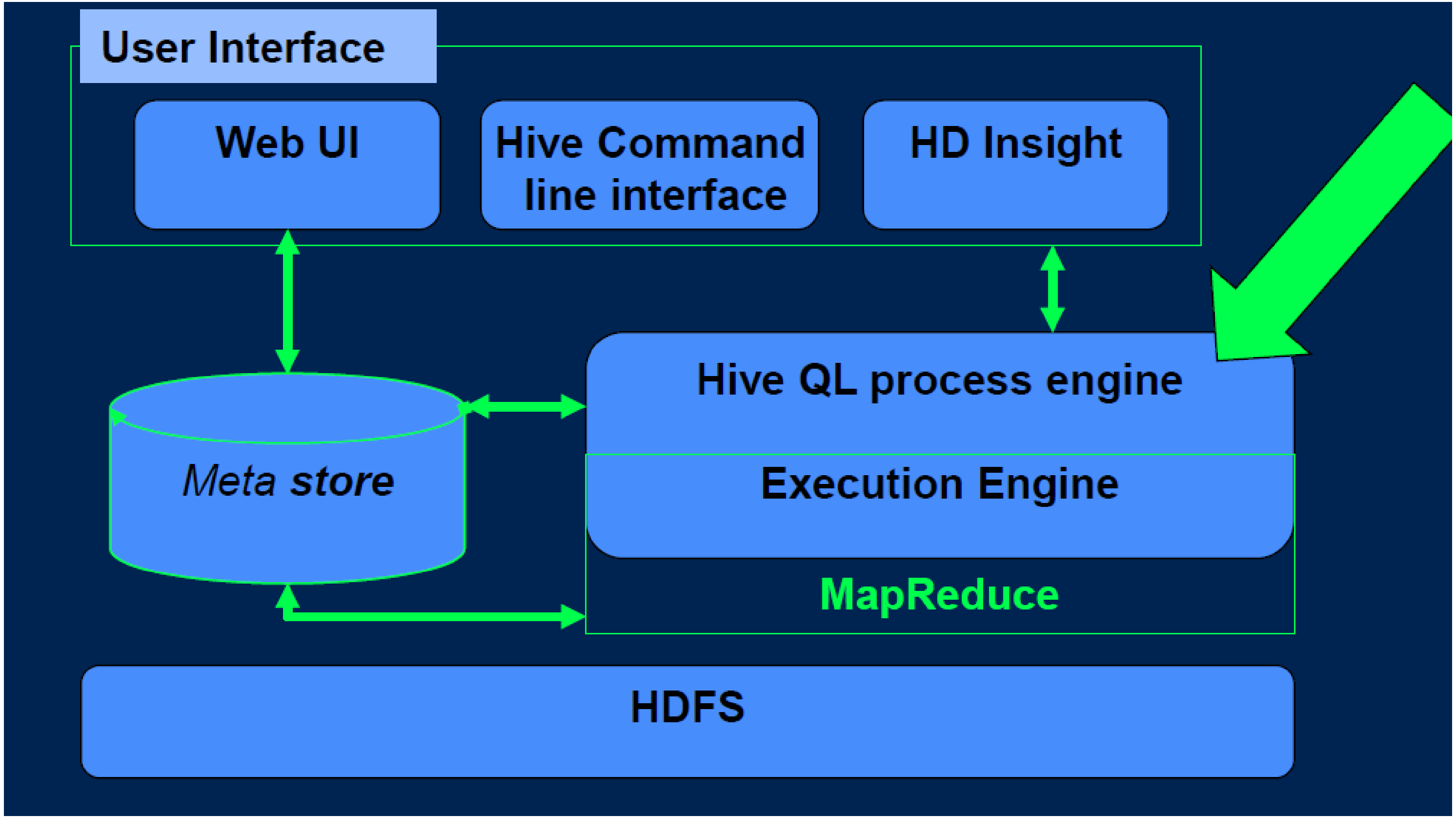


HIVE
JDBC
ODBC
Command Line Interface
Web Interface
Thrift Server
Metastore
Driver
(Compiler, Optimizer, Executor)
HADOOP
(MAP-REDUCE + HDFS)
Job Tracker
Name Node
Data Node
+
Task Tracker

Hive组件


MetaStore
Compiler
Driver
Hive Components
Execution Engine
Shell

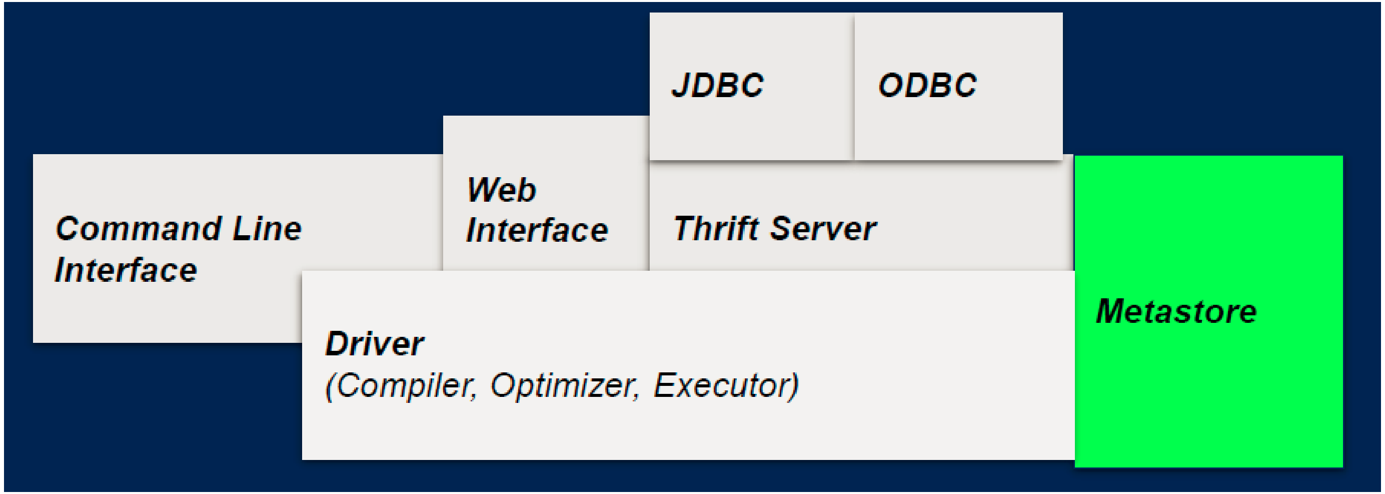
Hive架构


User Interface
Web UI
Hive Command line interface
HD Insight
Meta store
Hive QL process engine
Execution Engine
MapReduce
HDFS

Metastore

Stores the system catalog and meta data about tables, columns, partitions etc.

Stored on a traditional RDBMS



Driver

Manages the lifecycle of a HiveQL statement

Maintains a session handle and any session statistics

Query Compiler

The component that compiles HiveQL into a directed acyclic graph of map/reduce tasks

Optimizer

Consists of a chain of transformations

Performs Column Pruning , Partition Pruning, Repartitioning of Data

Execution Engine

Executes the tasks produced by the compiler in proper dependency order

Interacts with the underlying Hadoop instance

ThriftServer

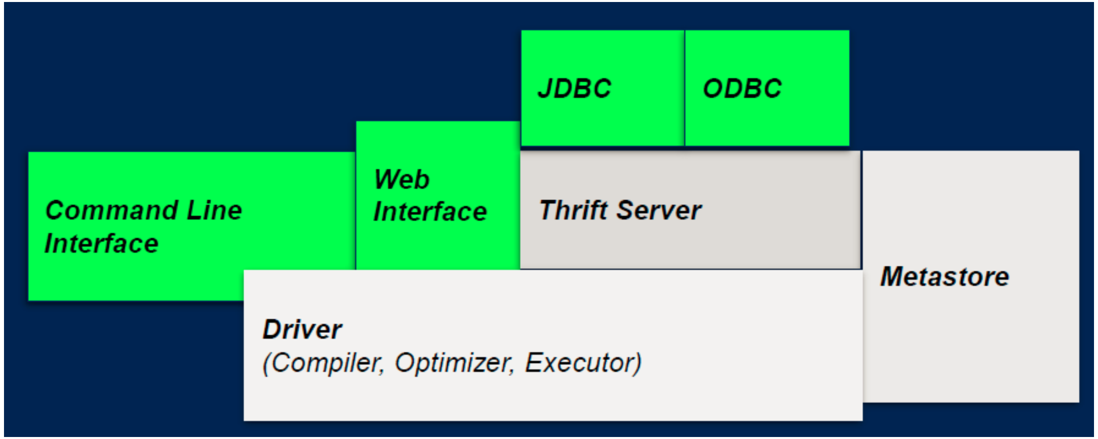
Provides a Thrift interface and a JDBC/ODBC server Enables Hive integration with other applications

Client Components

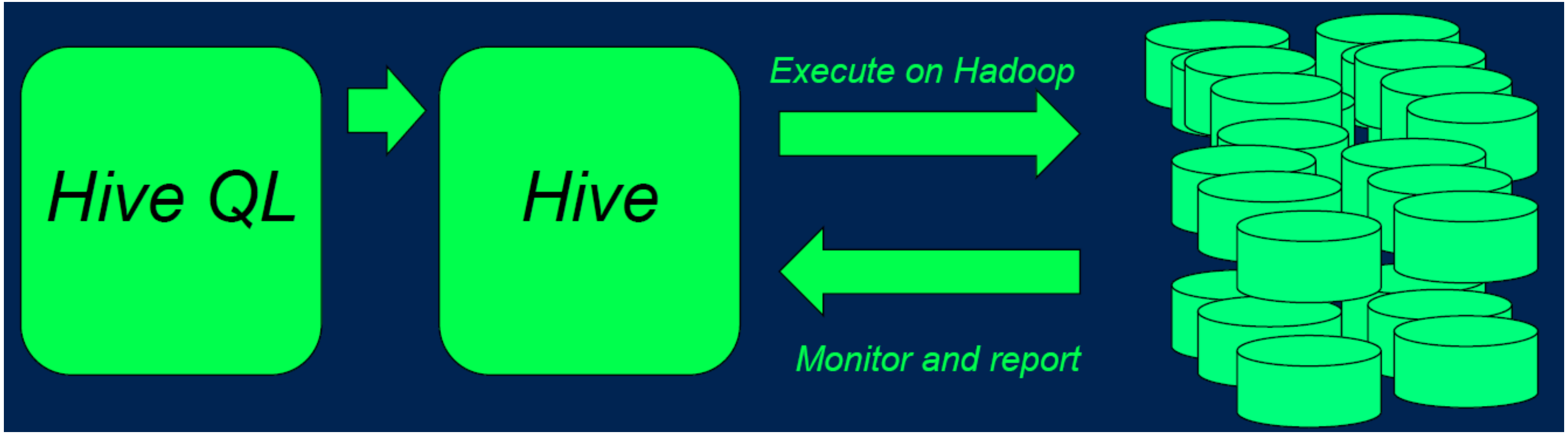
Command Line Interface(CLI)

Web UI

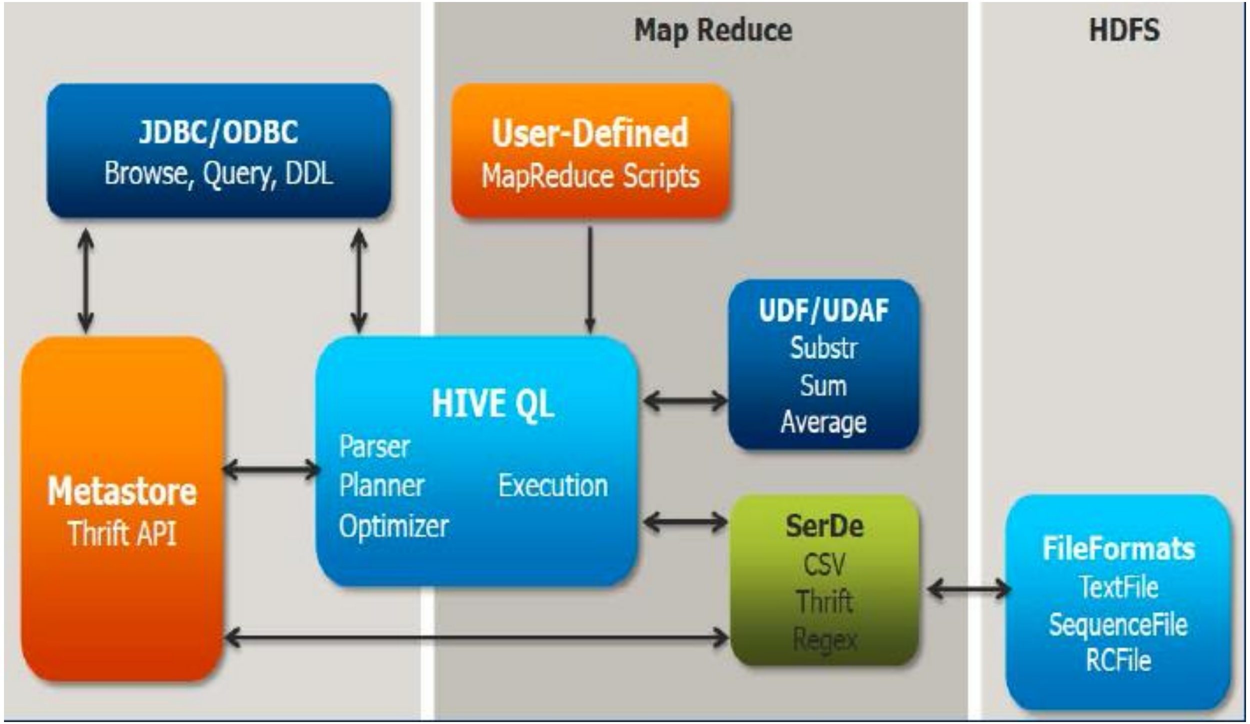
JDBC/ODBC driver



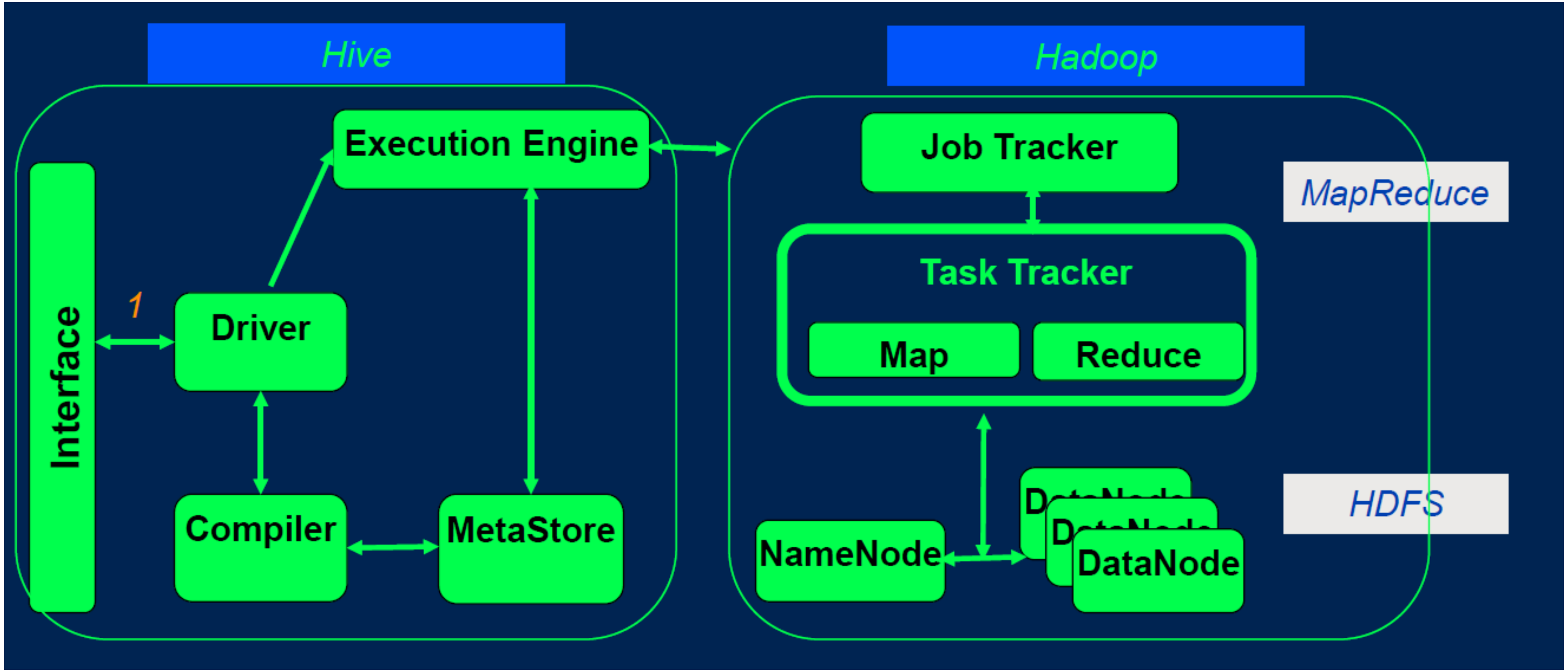
Hive执行过程



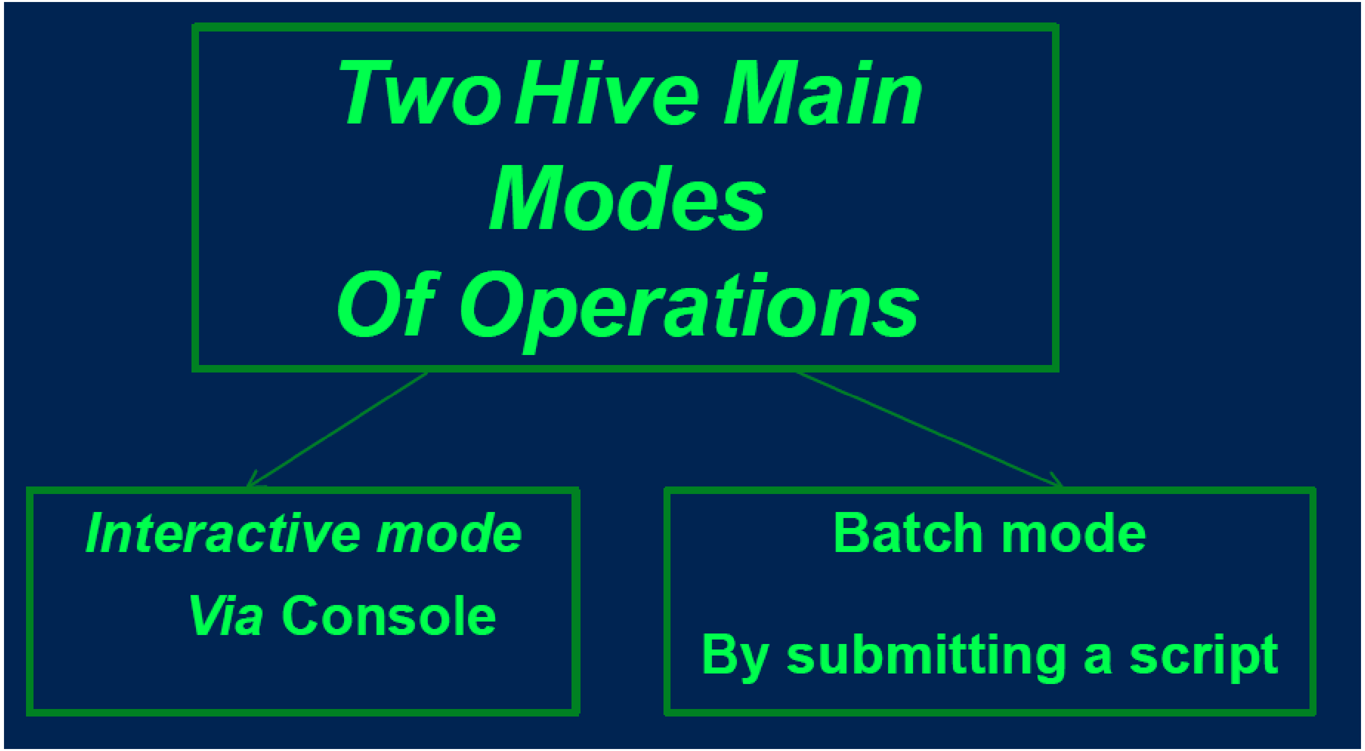
Hive工作流程


Map Reduce
HDFS
JDBC/ODBC
Browse, Query, DDL
User-Defined
MapReduce Scripts
UDF/UDAF
Substr
Sum
Average
Metastore
Thrift API
HIVE QL
Parser
Planner
Optimizer
Execution
SerDe
CSV
Thrift
Regex
FileFormats
TextFile
SequenceFile
RCFile

Hive工作流程


Hive
Hadoop
Execution Engine
Interface
1
Driver
Compiler
MetaStore
Job Tracker
MapReduce
Task Tracker
Map
Reduce
NameNode
DataNode
DataNode
DataNode
HDFS

Hive工作模式



Two Hive Main Modes Of Operations

Interactive mode

Via Console

Batch mode

By submitting a script

Hive’s Data Units

- Databases
- Tables
- Partitions
- Buckets (or clusters)

3-Levels: Tables → Partitions → Buckets

Data Model

- Table maps to a HDFS directory
- Partition maps to sub-directories under the table
- Bucket maps to files under each partition

Tables

Similar to tables in relational DBs

Each table has corresponding directory in HDFS

Partitions

- Analogous to dense indexes on partition columns
- Nested sub-directories in HDFS for each combination of partition column values
- Allows users to efficiently retrieve rows

Hive Data Structures

- Tables
- Rows
- Columns
- Partitions

Hive基础数据类型

- Integers
- Floats
- Doubles
- Strings

Hive扩展数据类型

- Associative arrays ：map<key-type, value-type>
- Lists ：list<element type>
- Structs ：struct<file name: file type…>

Hive支持的文件类型

Hive enables users store different file formats

Performance improvements

- TEXTFILE
- SEQUENCEFILE
- ORC
- RCFILE

Hive Interface

- **Command Line interface**
- **Web interface or Hue**
- **Java Database connectivity**

Hive Commands

Database

Set of Tables - name conflicts resolution

Table

Set of Rows - have the same columns

Row

A single record - a set of columns

Column

Value and type for a single value

Tables

- **SHOW TABLES**
- **CREATE TABLE**
- **ALTER TABLE**
- **DROP TABLE**

Table Commands

```
CREATE TABLE mytable (myint INT, bar STRING)
PARTITIONED BY (ds STRING);

SHOW TABLES '.*my';

ALTER TABLE mytable ADD COLUMNS (new_col
INT);

DROP TABLE mytable;
```

Hive Query Language

JOIN

- SELECT t1.a1 as c1, t2.b1 as c2 FROM t1 JOIN t2 ON (t1.a2 = t2.b2);

INSERTION

- INSERT OVERWRITE TABLE t1 SELECT * FROM t2;

Format rows

```
CREATE TABLE mypeople (id INT, name STRING)

ROW FORMAT

DELIMITED FIELDS TERMINATED BY <output format>

LINES TERMINATED BY ‘\n’;
```

Loading data into Hive

HDFS

- **LOAD DATA INPATH ‘mybigdata’ [OVERWRITE] INTO TABLE mypeople;**

Local file system

- **LOAD DATA LOCAL INPATH ‘mybigdata’ INTO TABLE mypeople;**

Partitions

- **LOAD DATA INPATH ‘myweblogs’ INTO TABLE mypeople PARTITION (dt=12-12-2020);**

BUCKETS

Set hive.enforce.bucketing property to true

```
CREATE TABLE mycustomers(id INT, purchases DOUBLE,
name STRING)

CLUSTERED BY id into 32 BUCKETS;
```

```
SELECT min(cost) FROM mysales TABLESAMPLE (BUCKET
10 OUT OF 32 ON rand());
```

VIEWS

Similar to SQL Views

Virtual table in Metastore

SHOW TABLES

JOINS

LEFT OUTER JOIN

RIGHT OUTER JOIN

FULL OUTER JOIN

```
hive> SELECT c.ID, c.NAME, c.AGE, o.AMOUNT
FROM CUSTOMERS c JOIN ORDERS o ON

(c.ID = o.CUSTOMER_ID);
```

外部表

```
CREATE EXTERNAL TABLE customers STORED AS AVRO

LOCATION

'hdfs:///user/hive/warehouse/customers'
```