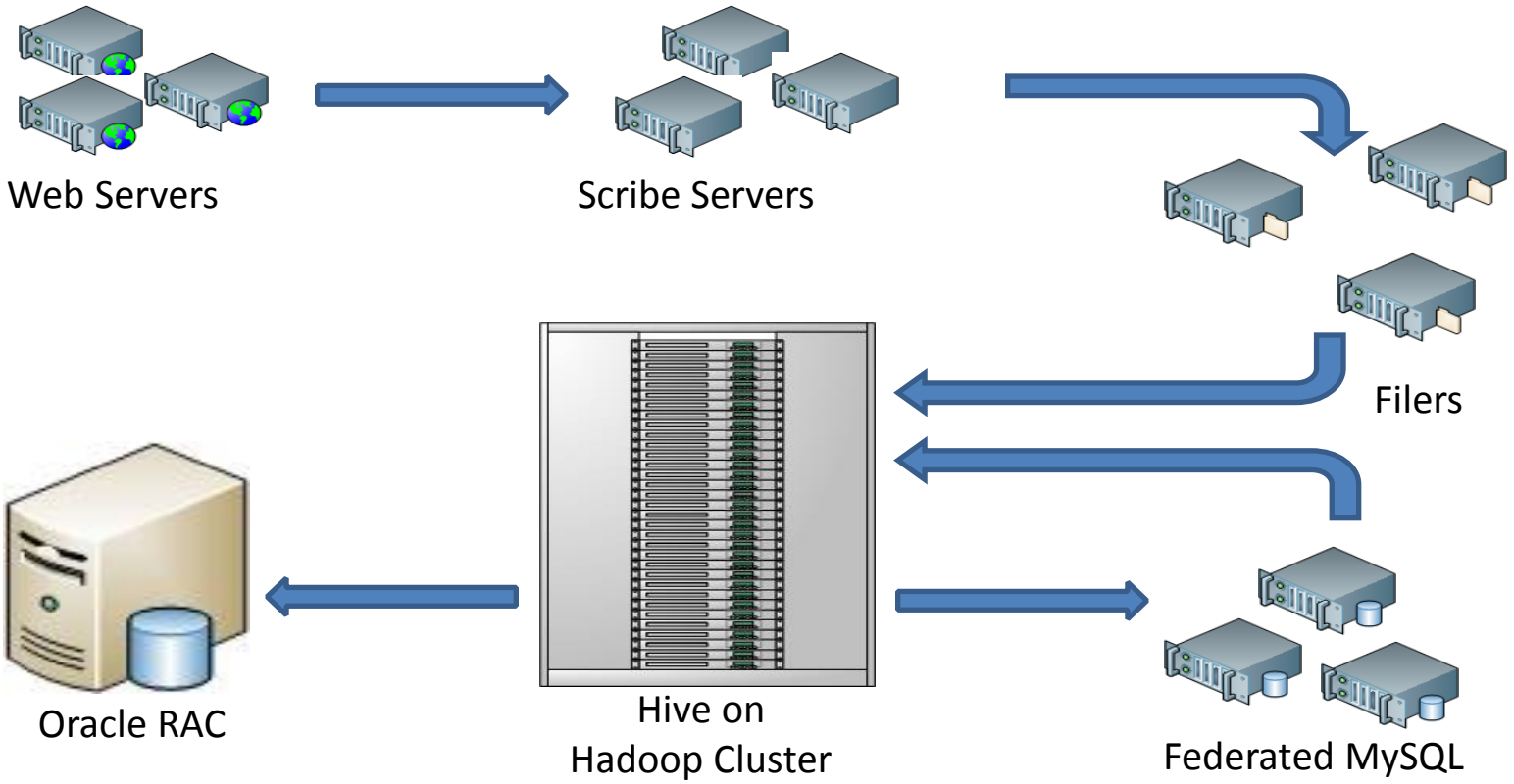
Hive

- 什么是Hive
- Hive的原理
- Hive简介
- Hive的表
- Hive的元数据
- 分区
- 执行计划
- Hive的UDF和UDAF
- Hive与MapReduce比较

Hive 数据仓库

Data Warehousing at Facebook



What is Hive

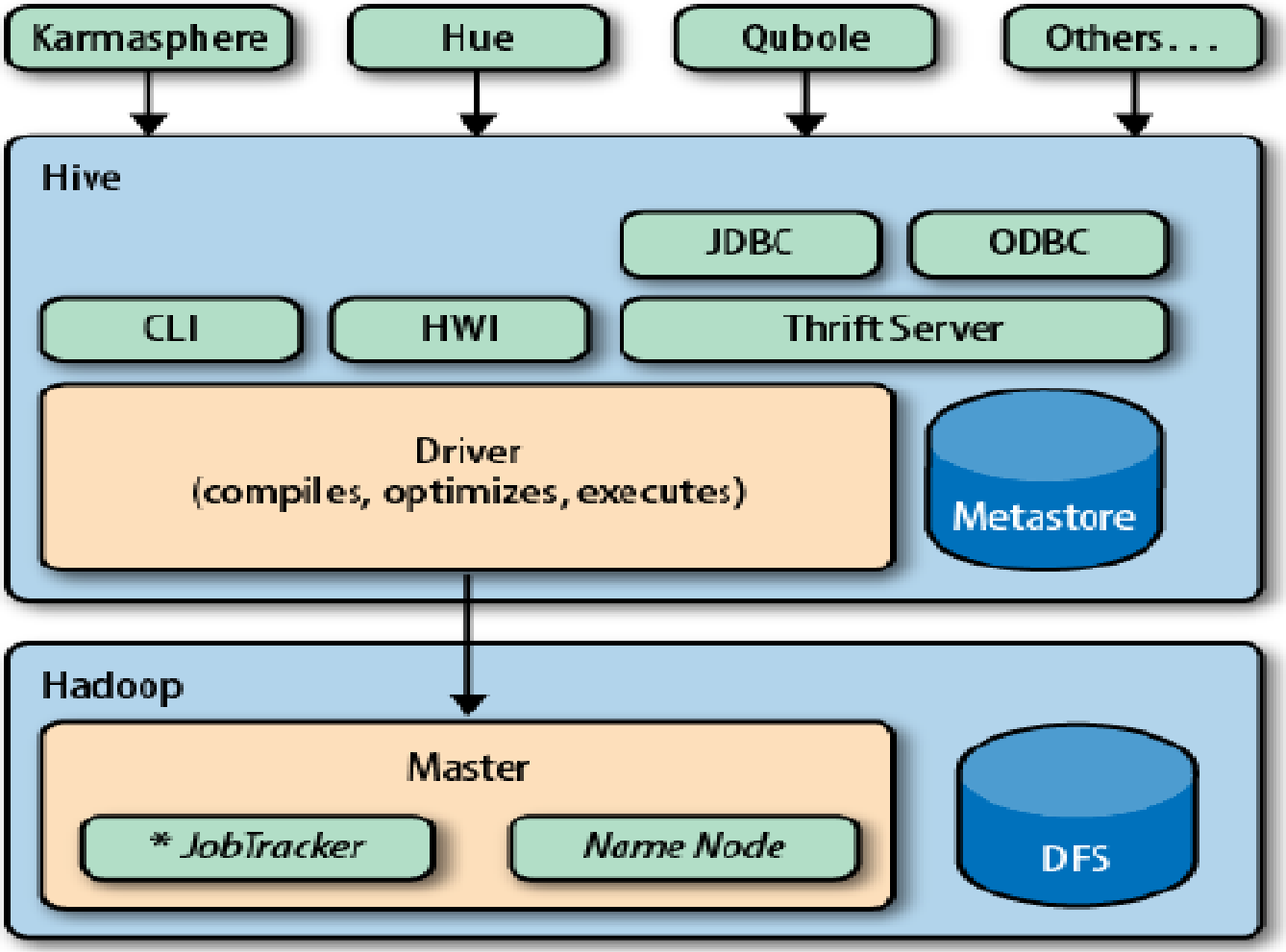
构建于hadoop的hdfs和mapred之上，用于管理和查询结构化/非结构化数据的数据仓库。

- 使用HQL作为查询接口
- 使用HDFS作为底层存储
- 使用MapRed作为执行层
- ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

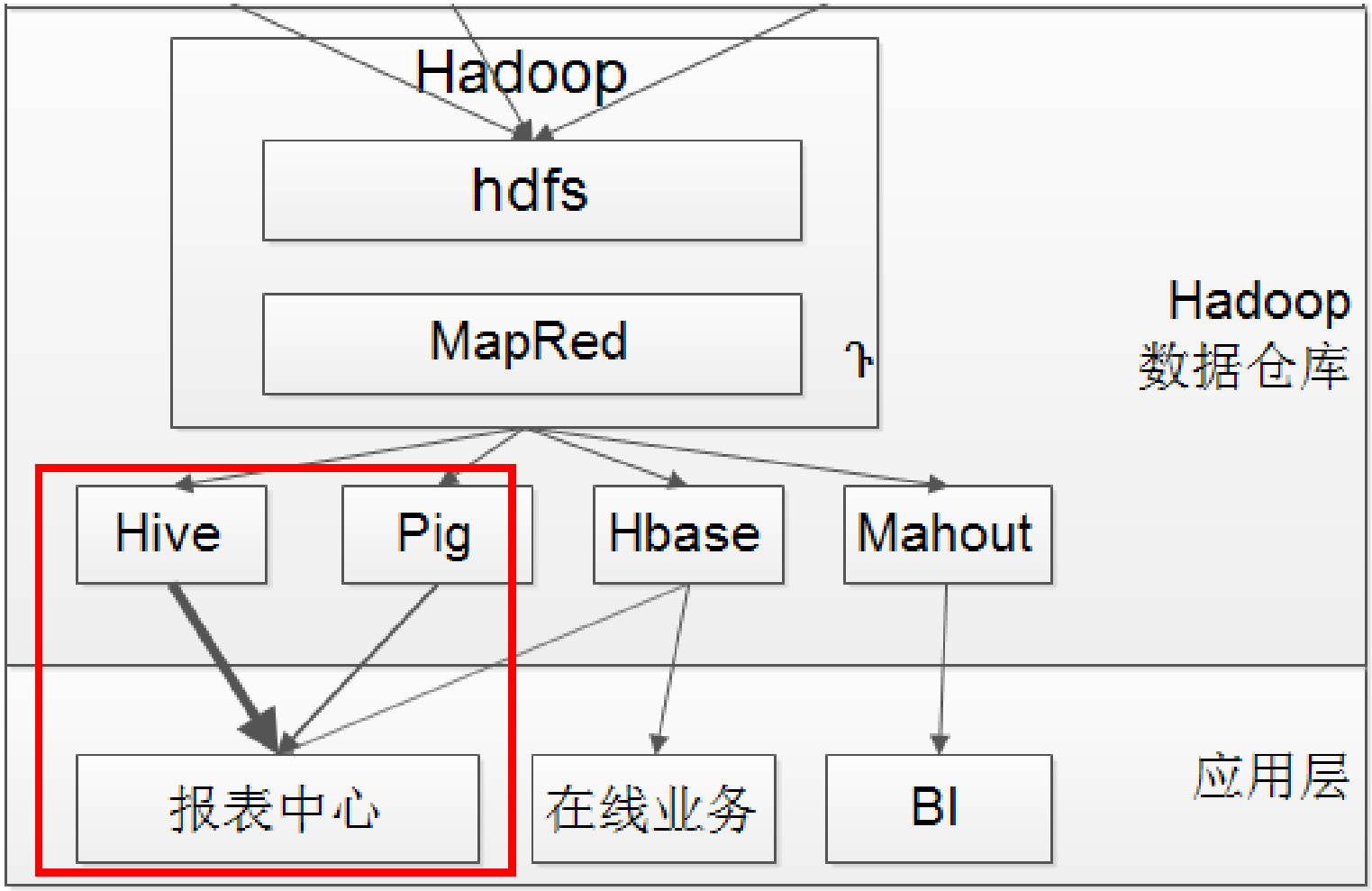
介绍两个非常棒的工具

- Impala
- Presto

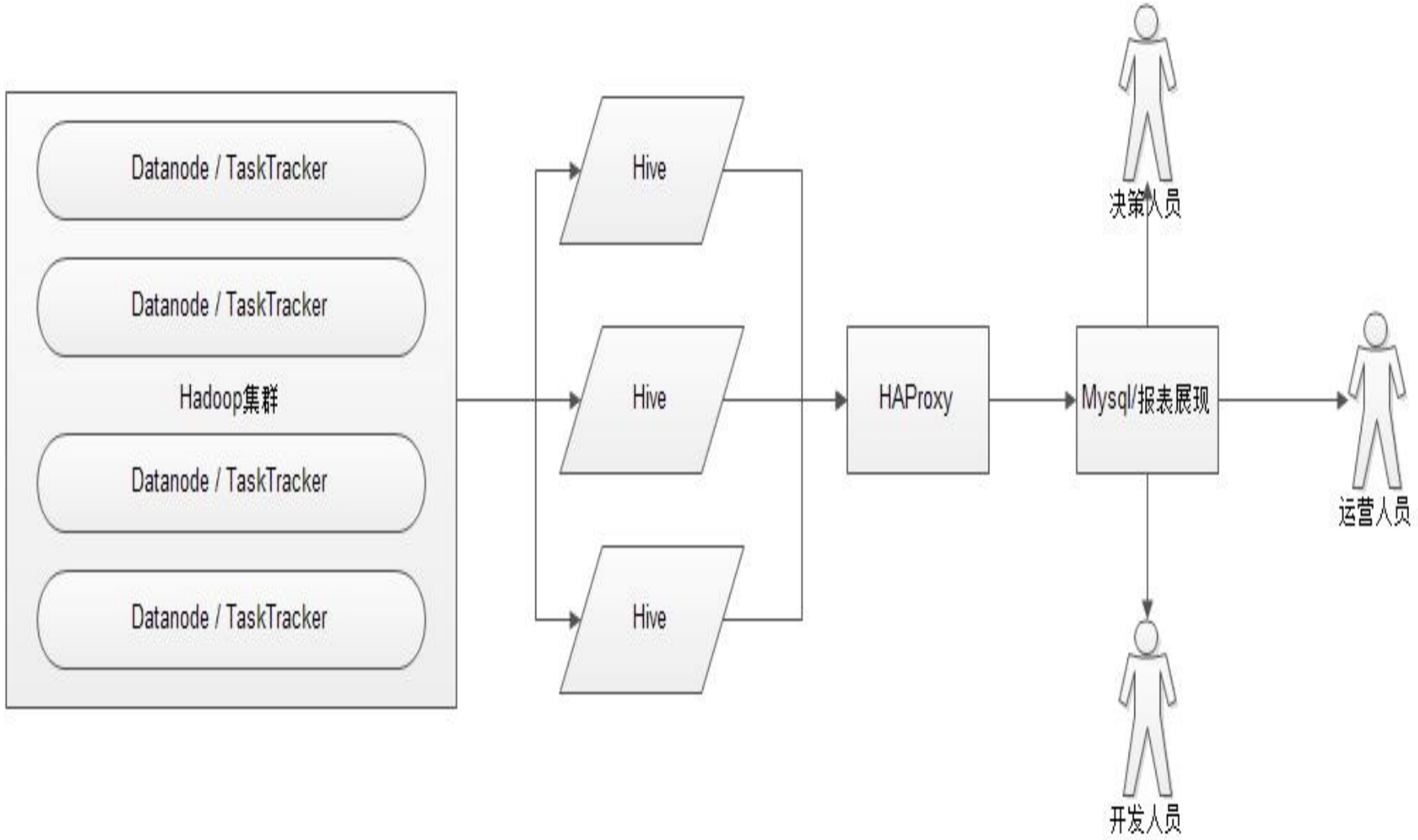
What is Hive


Karmasphere
Hue
Qubole
Others...
Hive
JDBC
ODBC
CLI
HWI
Thrift Server
Driver
(compiles, optimizes, executes)
Metastore
Hadoop
Master
* JobTracker
Name Node
DFS

What is Hive

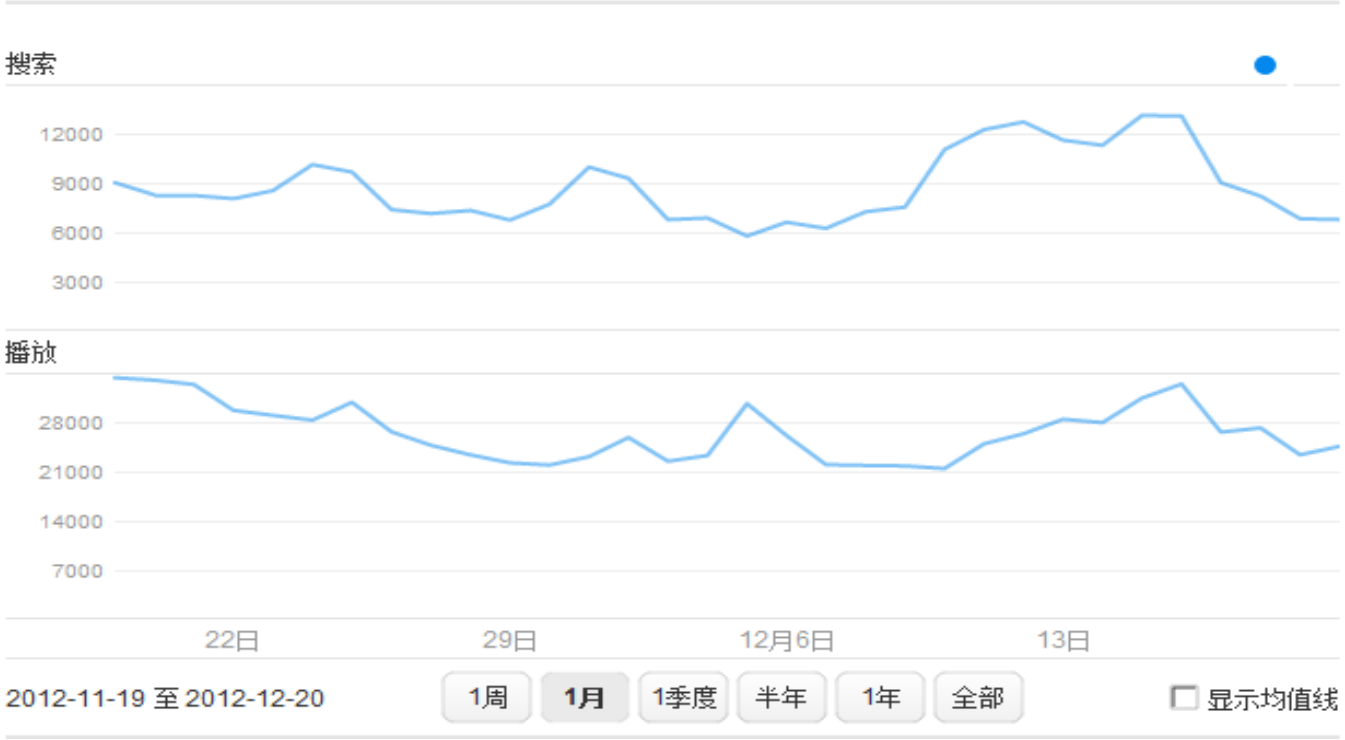

Hadoop
hdfs
MapRed
Hadoop
数据仓库
Hive
Pig
Hbase
Mahout
报表中心
在线业务
BI
应用层

Hive 应用


Datanode / TaskTracker
Datanode / TaskTracker
Hadoop集群
Datanode / TaskTracker
Datanode / TaskTracker
Hive
Hive
Hive
HAProxy
Mysql/报表展现
决策人员
运营人员
开发人员

Hive Application Cases

某视频播放和搜索指数



HiveQL vs SQL

对比项目	Hive	SQL
数据插入	支持批量导入	支持单条和批量导入
数据更新	不支持	支持
索引	支持	支持
分区	支持	支持
执行延迟	高	低
扩展性	好	有限

Hive部署及安装

1. 下载tar.gz，解压缩

2. 设置HADOOP_HOME

3. 为什么使用JDBC连接数据库做Metastore

4. 访问Hive 通过 thrift或JDBC

5. Hive与其他数据库的对接与数据ETL

Hive部署及安装

- hive 的元数据存储
  - hive默认使用内存数据库derby存储元数据，使用时不需要修改任何配置，缺点：hive server重启后所有的元数据都会丢失
  - hive还执行mysql、oracle等任何支持JDBC连接方式的数据库来存储元数据，需要修改相应的配置项

Hive部署及安装

```
<property>
  <name>hive.metastore.local</name>
  <value>false</value>
</property>
```

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://db1.mydomain.pvt/hive_db?createDatabaseIfNotExist=true</value>
</property>
```

```
<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
</property>
<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>database_user</value>
</property>
<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>database_pass</value>
</property>
```

Hivede 的访问

(1) cli这个就是Command Line Interface的简写，是Hive的命令行界面，用的比较多。这是默认的服务，直接可以在命令行里面使用。

（2）hiveserver：这个可以让Hive以提供Trift服务的服务器形式来运行，可以允许许多不同语言编写的客户端进行通信。使用需要启动HiveServer服务以和客户端联系，我们可以通过设置HIVE_PORT环境变量来设置服务器所监听的端口号，在默认的情况下，端口为10000。可以通过下面方式来启动hiveserve:

[hive@master ~]$ bin/hive --service hiveserver -p

（3）hwi：其实就是hive web interface的缩写，它是Hive的Web接口，是hive cli的一个web替换方案。

HiveQL

1. Select * from db.table1
2. Select count distinct uid from db.table1
3. 支持select、union all、join（left、right、mapjoin）

   like、where、各种聚合函数、支持json解析
4. 类SQL，并不完全支持

HiveQL

1. Hive建表语法

Create Table

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name
  [(col_name data_type [COMMENT col_comment], ...)]
  [COMMENT table_comment]
  [PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]
  [CLUSTERED BY (col_name, col_name, ...) [SORTED BY (col_name [ASC|DESC], ...)] INTO num_buckets BUCKETS]
  [SKEWED BY (col_name, col_name, ...) ON ([(col_value, col_value, ...), ...|col_value, col_value, ...]) (N
  [
   [ROW FORMAT row_format] [STORED AS file_format]
   | STORED BY 'storage.handler.class.name' [WITH SERDEPROPERTIES (...)]  (Note: only available starting wi
  ]
  [LOCATION hdfs_path]
  [TBLPROPERTIES (property_name=property_value, ...)]  (Note: only available starting with 0.6.0)
  [AS select_statement]  (Note: this feature is only available starting with 0.5.0, and is not supported wh
```

HiveQL

hive建表语法格式

- external 外部表，类似于mysql的csv引擎
- partitioned by 指定分区字段
- clustered by sorted by 可以对表和分区 对某个列进行分桶操作，也可以利用sorted by对某个字段进行排序
- row format 指定数据行中字段间的分隔符 和数据行分隔符
- stored as 指定数据文件格式：textfile sequence rcfile inputformat (自定义的inputformat 类)
- location 指定数据文件存放的hdfs目录

HiveQL

hive建表语句

```
CREATE TABLE page_view
 (viewTime INT, userid BIGINT, page_url STRING, referrer_url     STRING, ip
  STRING COMMENT 'IP Address of the User')
 COMMENT 'This is the page view table'
 PARTITIONED BY(dt STRING, country STRING)
 CLUSTERED BY(userid) SORTED BY(viewTime) INTO 32 BUCKETS
 ROW FORMAT DELIMITED
   FIELDS TERMINATED BY '\001'
  COLLECTION ITEMS TERMINATED BY '\002'
  MAP KEYS TERMINATED BY '\003'
STORED AS SEQUENCEFILE;
```

HiveQL

删除表

- drop table [IF EXISTS] table_name
- 删除内部表时会删除元数据和表数据文件
- 删除外部表（external）时只删除元数据

HiveQL

修改表 增加分区

```
ALTER TABLE page_view ADD PARTITION (dt='2008-08-08', country='us')
  location '/path/to/us/part080808' PARTITION (dt='2008-08-09',
  country='us') location '/path/to/us/part080809';
```

修改表 删除分区

```
ALTER TABLE page_view DROP PARTITION (dt='2008-08-08', country='us');
```

修改表 重命名表

```
ALTER TABLE table_name RENAME TO new_table_name
```

修改表 修改字段

```
ALTER TABLE test_change CHANGE a a1 STRING AFTER b;
```

HiveQL

加载数据

```
LOAD DATA INPATH '/user/myname/kv2.txt' OVERWRITE INTO TABLE
 invites PARTITION (ds='2008-08-15');
```

加载（本地、hdfs）文件到指定的表 分区

```
FROM src
    INSERT OVERWRITE TABLE dest1 SELECT src.* WHERE src.key < 100
    INSERT OVERWRITE TABLE dest2 SELECT src.key, src.value WHERE
 src.key >= 100 and src.key < 200
```

从指定表中选取数据插入到其他表中

HiveQL

select 语法结构

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
    FROM table_reference
    [WHERE where_condition]
    [GROUP BY col_list]
    [ CLUSTER BY col_list | [DISTRIBUTE BY col_list] [SORT BY col_list] ]
    [LIMIT number]
```

HiveQL

select 语法结构

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
    FROM table_reference
    [WHERE where_condition]
    [GROUP BY col_list]
    [ CLUSTER BY col_list | [DISTRIBUTE BY col_list] [SORT BY col_list] ]
    [LIMIT number]
```

HiveQL

select 案例

```
SELECT pv.pageid, u.age FROM page_view p JOIN user u ON (pv.userid =
 u.userid) JOIN newuser x on (u.age = x.age);
```

Hive Extension

User-defined Function

UDF 作用于单个数据行，输出一个数据 ，如字符处理函数
UDAF 作用于多个数据行，输出一个数据，如count，sum函数
UDTF 作用于单个数据行，输出多个数据行
支持用户使用java自定义开发UDF函数

Hive streaming

支持用户在hive QL语句中嵌入自定义的streaming处理脚本

Hive UDF

```
add jar /tmp/helloUDF.jar;
create temporary function helloworld as
    'com.hrj.hive.udf.HelloUDF';
select helloworld(t.col1) from t limit 10;
drop temporary function helloworld;
```

Hive 优化策略思路

- 使用 Partition 减少扫描数量
- 使用Map端Join
- 配置Reduce数量
- xml,json 提取 适用脚本提取,而非使用函数
- 使用 INSERT INTO LOCAL DIRECTORY ‘/home/me/pv_age_sum.dir’, 而非适用 HiveServer 。
- 使用 LZO 压缩存储数据
- 适用外部表,而非内部表
- hive.exec.compress.output = false,true
- 适用队列管理任务执行

Hive的SQL

- 处理结构化的数据，将数据列映射成字段

```
# 数据
Zhangsan 25 male
Lisi 20 female
Wangwu 66 mail
```

```
#表定义
create table user (
  name STRING,
  age int,
  gender STRING )
```

- 像操作SQL一样操作数据
  - Select * from user where name = ‘Zhangsan’;
  - Select gender, count(*) from user group by gender;

Hive的Table类型

表类型	语句	是否copy数据到warehouse	删除table，是否删除数据
普通表	create table	Y	Y
外部表	create external table	N	N

Hive Metastore

- 作用：保存表、分区、Index、视图信息
- 默认使用Derby作为数据库，缺点是单一session访问
- 可以变更为mysql或者其他JDO接口的数据库
- sqoop

Parition（分区）的概念

- 数据通过目录划分分区，分区字段是特殊字段
  - 目录结构：/pub/{pub_date}/{user_id}/{channel_id}
  - 添加分区的语句：ALTER TABLE fc ADD PARTITION (pub_date='20110803', user_id = '1', channel_id='5042') location '/pub/20110803/1/5042';
- 使用分区进行查询
  - Select * from fc where pub_date = ‘20110805’ and channel_id=‘250’
  - 日期字段（字符串类型）可以进行比较，如：t.pub_data>‘20110205’

Hive的UDF和UDAF

- UDF函数可以直接应用于select语句，对查询结构做格式化处理后，再输出内容。
- 编写UDF函数的时候需要注意一下几点：

  a）自定义UDF需要继承org.apache.hadoop.hive.ql.UDF。

  b）需要实现evaluate函。

  c）evaluate函数支持重载。

Hive的UDF案例

```
public final class Add extends UDF {
public Integer evaluate(Integer a, Integer b) {
        if (null == a || null == b) {
                return null;
        } return a + b;
}
public Double evaluate(Double a, Double b) {
        if (a == null || b == null)
                return null;
                return a + b;
        }
public Integer evaluate(Integer... a) {
        int total = 0;
        for (int i = 0; i < a.length; i++)
                if (a[i] != null)
                        total += a[i];
                         return total;
                }
}
```

Hive的UDF使用

4、使用UDF步骤

a）把程序打包放到目标机器上去；

b）进入hive客户端，添加jar包：hive>add jar /run/jar/udf_test.jar;

c）创建临时函数：hive>CREATE TEMPORARY FUNCTION add_example AS 'hive.udf.Add';

d）查询HQL语句：

```
SELECT add_example(8, 9) FROM scores;
SELECT add_example(scores.math, scores.art) FROM scores;
SELECT add_example(6, 7, 8, 6.8) FROM scores;
```

e）销毁临时函数：hive> DROP TEMPORARY FUNCTION add_example;

Hive的文件类型

- ◆TEXTFILE
- ◆SEQUEUEFILE
- ◆RCFILE

- RCFile是Facebook开发的一个集行存储和列存储的优点于一身，压缩比更高，读取列更快，它MapReduce环境中大规模数据处理中扮演着重要的角色。
- RCFile一种行列存储相结合的存储方式。首先，将数据按行分块，保证同一个record在一个块上，避免读一个记录需要读取多个Block。

Hive复合类型

- maps，
- array，
- struct

复合类型	功能解释
map	Map类型构建
struct	Struct类型构建
array	array类型构建

案例

日志格式

2014-03-03 12:22:34#127.0.0.1#get#amap#src=123&code=456&cookie=789#status=success&time=2s

2014-03-03 11:22:34#127.0.0.1#get#autonavi#src=123&code=456#status=success&time=2s&cookie

Hive transform

Transform的使用案例

Map

Reduce

案例：

```
insert overwrite table snowman_service_rc partition(dt='2013-11-21',service='shift') select transform(log) using 'python hive_shift_parse.py' as sm_datetime,sm_appid,sm_language,sm_iosMaxVersion,sm_iosMinVersion,sm_messageid,sm_logtype,sm_request,sm_response,sm_status,sm_responsetime,sm_ip,sm_province,sm_city,sm_town,sm_day_time from snowman_service_raw partition(dt='2013-11-21',service='shift');
```

Hive的执行计划

1) explain select * from table1；

2) explain select name from table1；

3) explain select * from table1 where name=“jayliu”

分析执行结果？

Hive JDBC

1）启动Hive远程服务接口：

bin/hive --service hiveserver

2） JDBC 驱动

```
Class.forName(driverName);
Connection con = DriverManager.getConnection("jdbc:hive://1127.0.0.1:10000/default", "", "");
Statement stmt = con.createStatement();
```

1）查询Hive表数据

```
String tableName = "testHiveDriverTable";
        stmt.executeQuery("drop table " + tableName);
        ResultSet res = stmt.executeQuery("create table "
    + tableName + " (key int, value string)");
        String sql = "show tables '" + tableName + "'";
        System.out.println("Running: " + sql);
        res = stmt.executeQuery(sql);
        if (res.next()) {
          System.out.println(res.getString(1));
        }
```

Hive与MapReduce的比较

Hive

1) 标准的数据格式

2）支持的算法有限

3）不可控，

4）开发周期短，成本底

MapReduce

1）非标准的数据格式

2）算法

3）可以随便改变，灵活

4）开发周期长