Hive

Hive是建立在Hadoop上的数据仓库文件，用于查询和管理存放在分布式存储上的大规模数据集。它提供：

（1）一系列的工具，可以方便地对数据进行提取/转化/加载（ETL）；

    （2）一种可以对各种数据格式上进行结构化的机制；

    （3）存取存放在Apache HDFS或其他存储系统如Apache HBase上的文件；

    （4）通过MapReduce执行查询功能。

Hive 定义了简单的类SQL查询语言，称为QL，它允许熟悉SQL的用户查询数据。同时，这个语言也允许熟悉MapReduce的开发者开发自定义的mapper和reducer来处理内建的mapper和reducer无法完成的复杂分析工作。QL也可以被扩展，以让用户自定义标量函数（UDF's）、聚合（UDAF's）和表函数（UDTF's）。

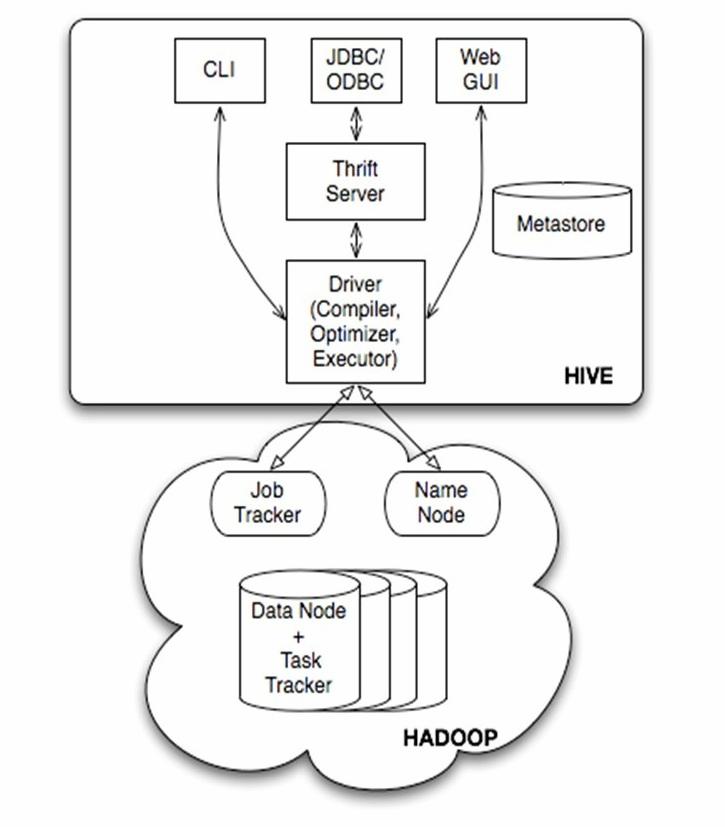
Hive并不要求使用"Hive格式"来读写数据－－没有这样的格式，Hive没有专门的数据格式。Hive可以很好的工作在Apache Thrift、控制分隔符、或用户指定的数据格式上。

  Hive并不是为OLTP工作负载设计的，也不提供实时查询和行级别的更新。它最常用于批量作业，比较适合工作在只追加的大数据集上（如Web日志）。Hive的核心设计思想是可伸缩性（通过动态添加到Hadoop集群的机器来进行横向扩展）、可扩展性（使用MapReduce框架和UDF/UDAF/UDTF）、容错性、与输入格式的松耦合性。

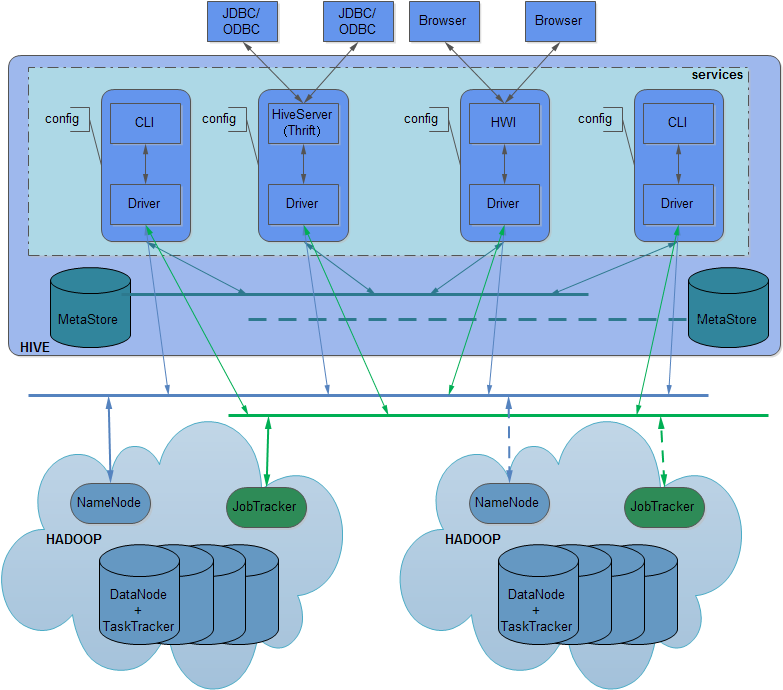
# 1、Hive的体系结构

## 1.1 体系结构

Hive的结构如下图所示：



Hive体系结构



Hive的体系结构可以分成以下几个部分：

1）用户接口CLI，Client和WUI，其中最常用的是CLI，Cli启动的时候，会同时启动一个Hive副本。Client是Hive的客户端，用户连接至Hive Server。在启动Client模式的时候，需要指出Hive Server所在节点，并且在该节点启动Hive Server。WUI是通过浏览器访问Hive。

2）Hive将元数据存储在数据库中，如mysql,derby（默认）。Hive中的元数包括表的名字，表的列和分区以及属性，表的属性（是否为外部表等），表的数据所在目录等。

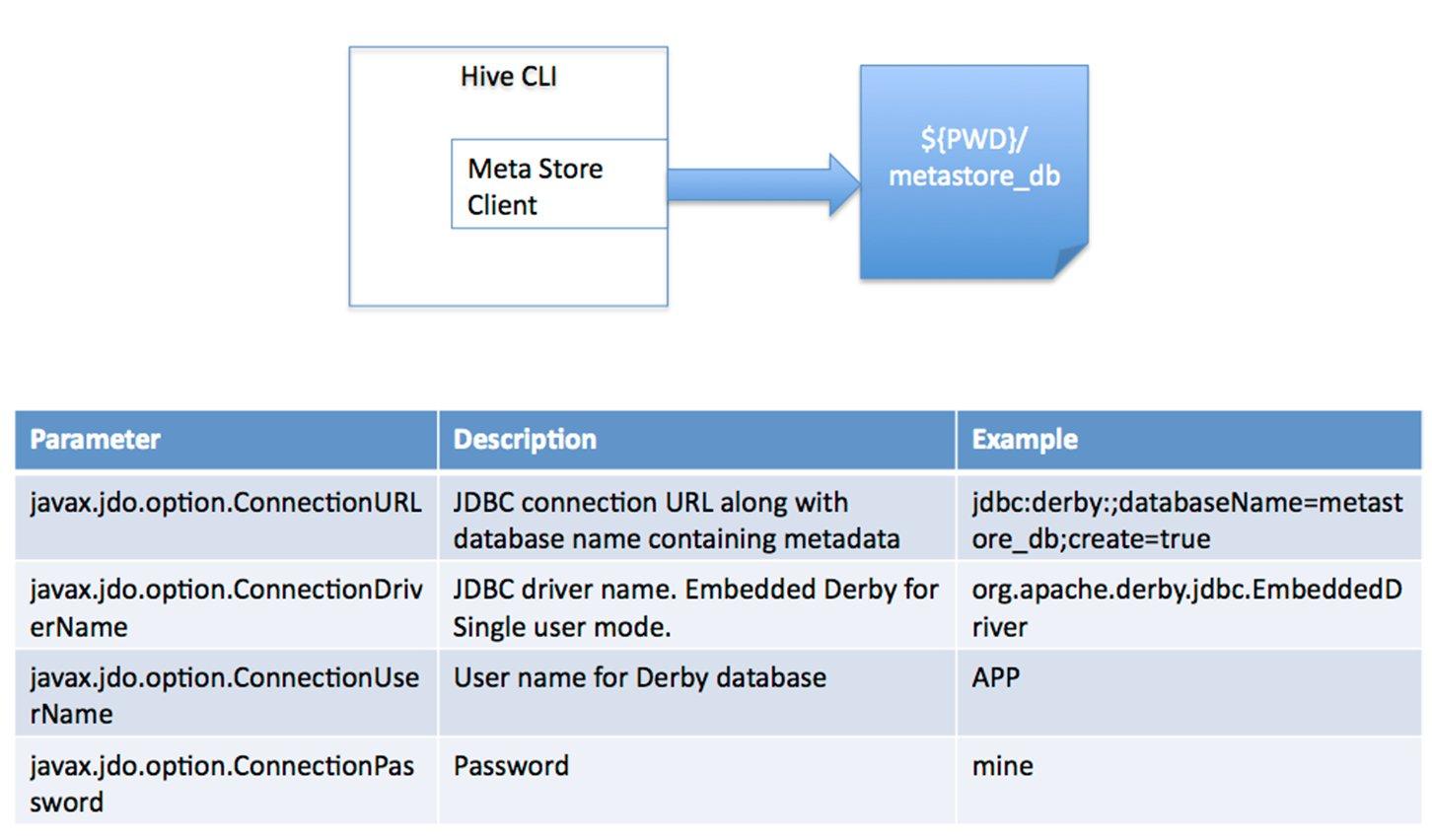
3）解析器、编译器、优化器完成HQL查询语句从词法、语法、编译、优化以及查询计划的生成。生成的查询计划存储在HDFS中，并在随后由Mapreduce调用执行。

4）Hive的数据存储在HDFS中，大部分的查询、计算由Mapreduce完成（包括\*查询）

## 1.2 运行模式

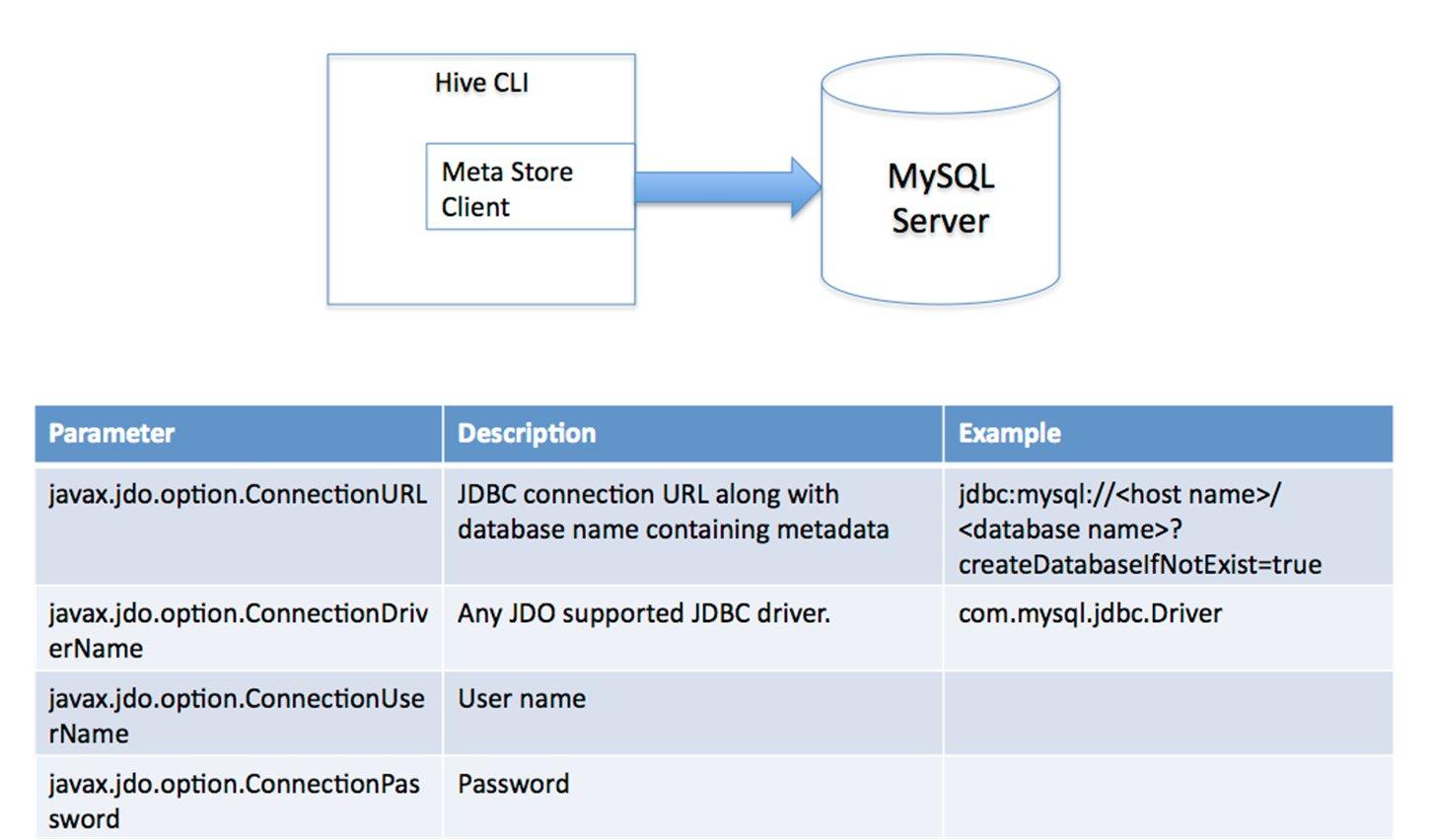
1）单用户模式

此模式连接到一个In-memory 的数据库Derby，一般用于Unit Test。



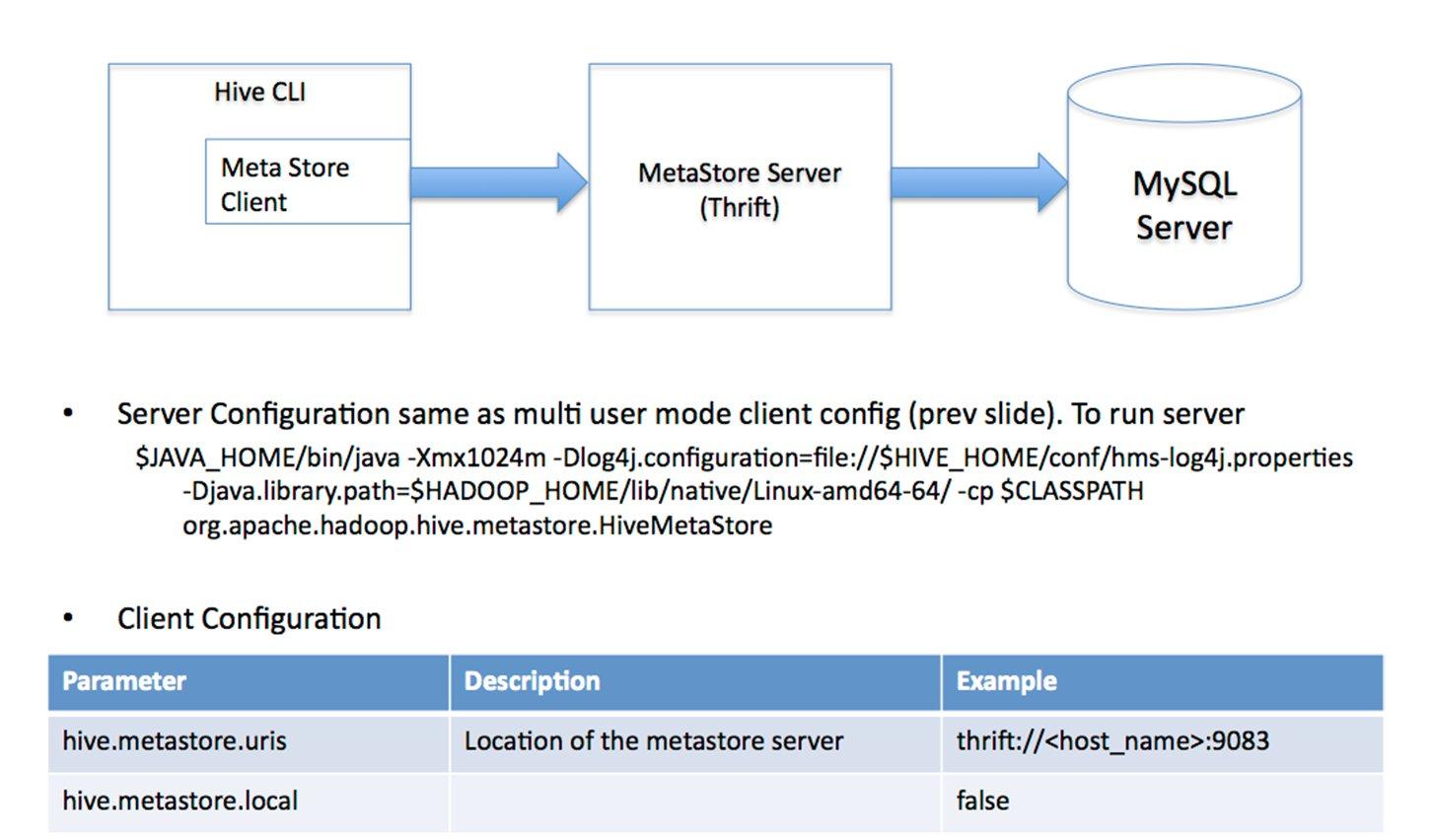
2）多用户模式

通过网络连接到一个数据库中，是最经常使用到的模式



3）远程服务器模式

用于非Java客户端访问元数据库，在服务端启动MetaStoreServer，客户端利用Thrift协议通过MetaStoreServer访问元数据库。



对于数据存储，Hive没有专门的数据存储格式，也没有为数据建立索引，用户可以非常自由的组织Hive中的表，只需要在创建表的时候告诉Hive数据中的列分隔符和行分隔符，Hive就可以解析数据。Hive中所有的数据都存储在HDFS中，存储结构主要包括数据库、文件、表和视图。

## 1.3 Hive数据模型

Hive中包含以下数据模型：Table内部表，External Table外部表，Partition分区，Bucket桶。Hive默认可加载seqence file,RCFile。

Hive的数据模型介绍如下：

    （1）Hive数据库

    类似传统数据库的DataBase，在第三方数据库里实际是一张表。简单示例命令行 hive > create database test\_database;

    （2）内部表

    Hive的内部表与数据库中的Table在概念上是类似。每一个Table在Hive中都有一个相应的目录存储数据。例如一个表pvs，它在HDFS中的路径为/wh/pvs，其中wh是在hive-site.xml中由${hive.metastore.warehouse.dir} 指定的数据仓库的目录，所有的Table数据（不包括External Table）都保存在这个目录中。删除表时，元数据与数据都会被删除。

    内部表简单示例：  
    创建数据文件：test\_inner\_table.txt

    创建表：create table test\_inner\_table (key string)  
    加载数据：LOAD DATA LOCAL INPATH ‘filepath’ INTO TABLE test\_inner\_table  
    查看数据：select \* from test\_inner\_table;  select count(\*) from test\_inner\_table  
    删除表：drop table test\_inner\_table

    （3）外部表

    外部表指向已经在HDFS中存在的数据，可以创建Partition。它和内部表在元数据的组织上是相同的，而实际数据的存储则有较大的差异。内部表的创建过程和数据加载过程这两个过程可以分别独立完成，也可以在同一个语句中完成，在加载数据的过程中，实际数据会被移动到数据仓库目录中；之后对数据对访问将会直接在数据仓库目录中完成。删除表时，表中的数据和元数据将会被同时删除。而外部表只有一个过程，加载数据和创建表同时完成（CREATE EXTERNAL TABLE ……LOCATION），实际数据是存储在LOCATION后面指定的 HDFS 路径中，并不会移动到数据仓库目录中。当删除一个External Table时，仅删除该链接。

外部表简单示例：

创建数据文件：

test\_external\_table.txt

创建表：create external table test\_external\_table (key string)  
    加载数据：LOAD DATA INPATH ‘filepath’ INTO TABLE test\_inner\_table  
    查看数据：select \* from test\_external\_table;  •select count(\*) from test\_external\_table  
    删除表：drop table test\_external\_table

    （4）分区

    Partition对应于数据库中的Partition列的密集索引，但是Hive中Partition的组织方式和数据库中的很不相同。在Hive中，表中的一个Partition对应于表下的一个目录，所有的Partition的数据都存储在对应的目录中。例如pvs表中包含ds和city两个Partition，则对应于ds = 20090801, ctry = US 的HDFS子目录为/wh/pvs/ds=20090801/ctry=US；对应于 ds = 20090801, ctry = CA 的HDFS子目录为/wh/pvs/ds=20090801/ctry=CA。

    分区表简单示例：  
    创建数据文件：test\_partition\_table.txt  
    创建表：create table test\_partition\_table (key string) partitioned by (dt string)  
    加载数据：LOAD DATA INPATH ‘filepath’ INTO TABLE test\_partition\_table partition (dt=‘2006’)  
    查看数据：select \* from test\_partition\_table;  select count(\*) from test\_partition\_table  
    删除表：drop table test\_partition\_table

    （5）桶

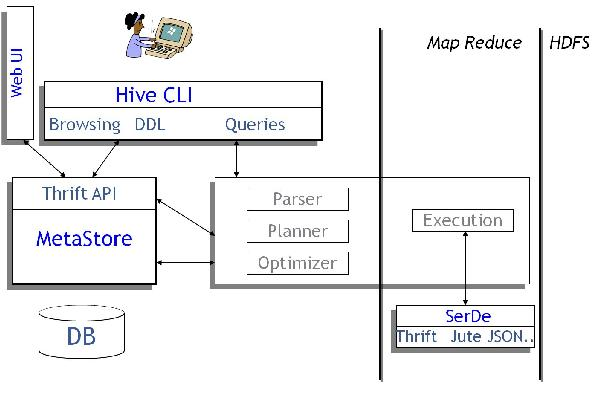
    Buckets是将表的列通过Hash算法进一步分解成不同的文件存储。它对指定列计算hash，根据hash值切分数据，目的是为了并行，每一个Bucket对应一个文件。例如将user列分散至32个bucket，首先对user列的值计算hash，对应hash值为0的HDFS目录为/wh/pvs/ds=20090801/ctry=US/part-00000；hash值为20的HDFS目录为/wh/pvs/ds=20090801/ctry=US/part-00020。如果想应用很多的Map任务这样是不错的选择。

    桶的简单示例：  
    创建数据文件：test\_bucket\_table.txt  
    创建表：create table test\_bucket\_table (key string) clustered by (key) into 20 buckets  
    加载数据：LOAD DATA INPATH ‘filepath’ INTO TABLE test\_bucket\_table  
    查看数据：select \* from test\_bucket\_table;  set hive.enforce.bucketing = true;

    （6）Hive的视图

    视图与传统数据库的视图类似。视图是只读的，它基于的基本表，如果改变，数据增加不会影响视图的呈现；如果删除，会出现问题。•如果不指定视图的列，会根据select语句后的生成。  
    示例：create view test\_view as select \* from test

# 2、Hive执行原理



Hive构建在Hadoop之上：

HQL中对查询语句的解析、优化、生成查询计划是由Hive完成的

所有的数据都存储在Hadoop中

查询计划被转化为MapReduce任务，在Hadoop中执行（有些查询没有MR任务，如Select \* from table）

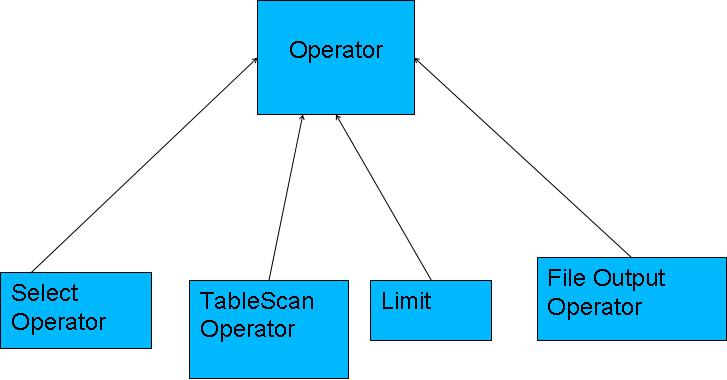
Hadoop和Hive都是用UTF-8编码

Hive编译器将一个Hive QL转换操作符，操作符Operator是Hive的最小的处理单元，每个操作符代表HDFS的一个操作或者移到MapReduce作业。

Operator都是Hive定义的一个处理过程，其定义有：

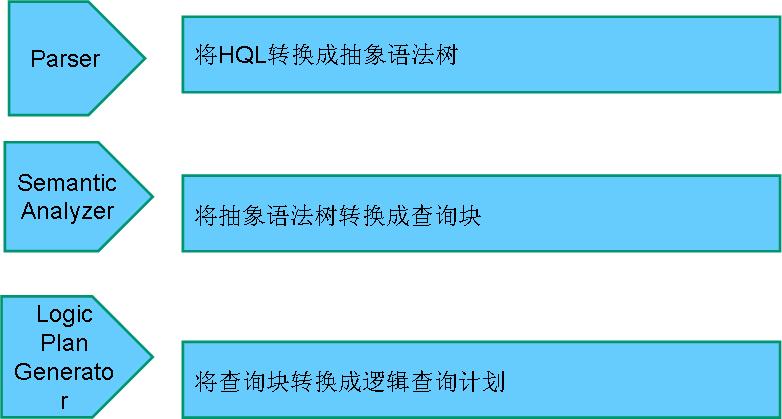
protected List <Operator<? extends Serializable >> childOperators;   
 protected List <Operator<? extends Serializable >> parentOperators;   
 protected boolean done;   // 初始化值为false

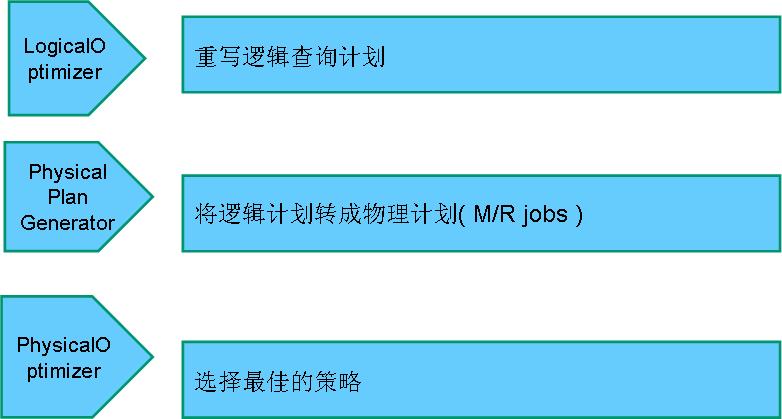
所有操作构成了Operator图，Hive正是基于这些图关系来处理诸如limit,group by,join等操作：



Hive通过ExecMapper和ExecReducer执行MapReduce任务，在执行MapReduce时有两种模式，即本地模式和分布式模式。

Hive的编译器组成：





# 3、Hive安装与配置

下载Hive安装包

svn co http://192.168.33.2/svn/hadoop/trunk/hive-0.12.0-cdh5.1.0/

采用第二种运行模式，使用Mysql存储元数据，安装Mysql并设置root密码，创建数据库hive\_metadata

yum -y install mysql-server

配置hive，主要配置hive-env.sh（环境），hive-site.xml（运行参数）及hive-log4j.properties

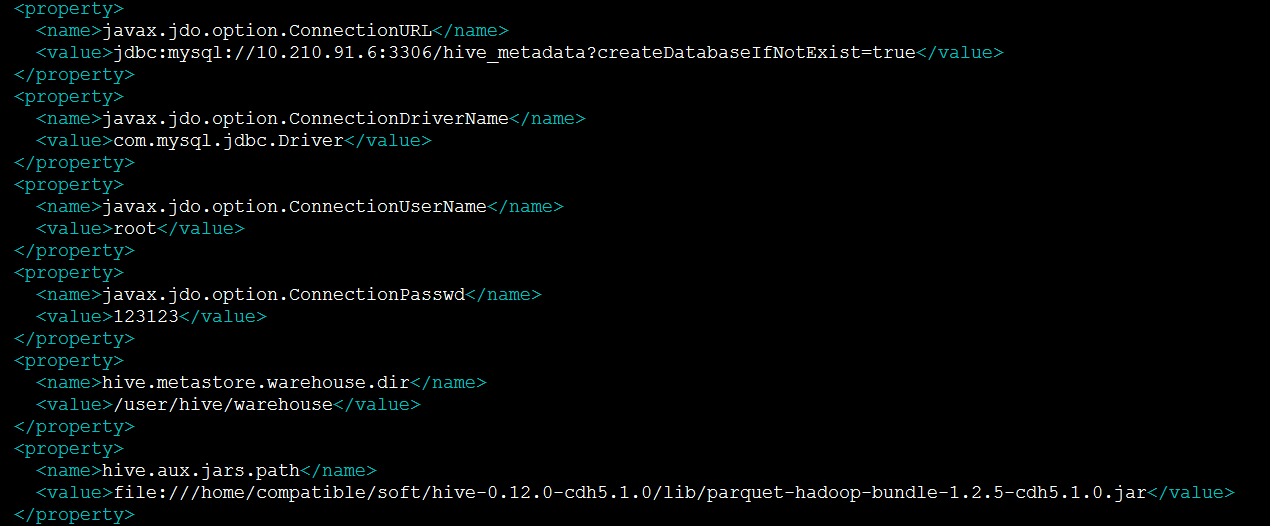
# Set HADOOP\_HOME to point to a specific hadoop install directory

HADOOP\_HOME=/home/compatible/soft/hadoop-2.6.0/

# Hive Configuration Directory can be controlled by:

export HIVE\_CONF\_DIR=/home/compatible/soft/hive-0.12.0-cdh5.1.0/conf/

针对hive-site.xml，主要设置hive.metastore.warehouse.dir及metadata的存储数据库参数，如下图所示：



# 4、Hive使用

上面进行了Hive的安装与配置，下面介绍其使用：

1）Hive的启动：

./hiveserver2

启动thrift server

2）查看metadata元数据

$mysql -u root -p

mysql> use hive\_metadata;

mysql> show tables;

数据库中会显示创建的表：

Tables\_in\_hive\_metadata |

+---------------------------+

| BUCKETING\_COLS |

| CDS |

| COLUMNS\_V2 |

| DATABASE\_PARAMS |

| DBS |

| PARTITION\_KEYS |

| SDS |

| SD\_PARAMS |

| SEQUENCE\_TABLE |

| SERDES |

| SERDE\_PARAMS |

| SKEWED\_COL\_NAMES |

| SKEWED\_COL\_VALUE\_LOC\_MAP |

| SKEWED\_STRING\_LIST |

| SKEWED\_STRING\_LIST\_VALUES |

| SKEWED\_VALUES |

| SORT\_COLS |

| TABLE\_PARAMS |

| TBLS |

| VERSION |

3）Hive启动成功，hive创建表的命令如下：

CREATE TABLE t\_hive(a int ,b int,c int) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

4）生成数据，并导入表

以一定格式，生成数据文件t\_hive.txt，以tab键为间隔

LOAD DATA LOCAL INPATH '/../t\_hive.txt' INTO TABLE t\_hive

5）从HDFS中查找刚刚导入的数据

hdfs dfs -cat /user/hive/warehouse/t\_hive.txt

6）直接从表中查找数据

hive> selest \* from t\_hive

会返回查询到的结果

# 5、Hive功能测试

## 5.1创建表测试

|  |  |  |  |
| --- | --- | --- | --- |
| 项目: | 功能测试 | 分项目: | Table create测试 |
| 用例编号: | TESTHIVE\_TABLE\_CREATE | 版本: |  |
| 用例分数 |  | 适用场景 | 通用 |
| 测试目的: | 测试TABLE的创建 | | |
| 预置条件: |  | | |
| 测试步骤: | 1、创建表，pokes，以tab键为间隔生成列  创建分区表 Invites，以某一个列为标志  2、查看表，describe | | |
| SQL语句 | 1、create table pokes(id int,cont string) row format delimited fields terminated by '\t' stored as textfile  2、create bale invites (yymm string,id int) partition by(dt string) | | |
| 预期结果: | 返回ok | | |
| 测试结果: |  | | |

## 5.2修改

|  |  |  |  |
| --- | --- | --- | --- |
| 项目: | 功能测试 | 分项目: | Table Alter测试 |
| 用例编号: | TESTHIVE\_TABLE\_Alter | 版本: |  |
| 用例分数 |  | 适用场景 | 通用 |
| 测试目的: | 测试TABLE的修改 | | |
| 预置条件: |  | | |
| 测试步骤: | 1、向表中增加分区/删除分区  2、重命名表  3、修改，增加及覆盖列 | | |
| SQL语句 |  | | |
| 预期结果: | 返回ok | | |
| 测试结果: |  | | |

## 5.3视图

|  |  |  |  |
| --- | --- | --- | --- |
| 项目: | 功能测试 | 分项目: | Table VIEW测试 |
| 用例编号: | TESTHIVE\_TABLE\_VIEW | 版本: |  |
| 用例分数 |  | 适用场景 | 通用 |
| 测试目的: | 创建视图 | | |
| 预置条件: |  | | |
| 测试步骤: | 1、create view  2、查看视图中的数据 | | |
| SQL语句 |  | | |
| 预期结果: | 返回ok | | |
| 测试结果: |  | | |

## 5.4 加载数据

|  |  |  |  |
| --- | --- | --- | --- |
| 项目: | 功能测试 | 分项目: | Table LOAD测试 |
| 用例编号: | TESTHIVE\_TABLE\_LOAD | 版本: |  |
| 用例分数 |  | 适用场景 | 通用 |
| 测试目的: | 从本地及HDFS中导入数据到表格 | | |
| 预置条件: |  | | |
| 测试步骤: | 1、创建表  2、从本地文件导入数据，查看表中的数据  3、从hdfs文件导入数据，查看表中的数据 | | |
| SQL语句 |  | | |
| 预期结果: | 返回ok | | |
| 测试结果: |  | | |

## 5.5 Hive的查询（Select）

|  |  |  |  |
| --- | --- | --- | --- |
| 项目: | 功能测试 | 分项目: | Table Select |
| 用例编号: | TESTHIVE\_TABLE\_SELECT | 版本: |  |
| 用例分数 |  | 适用场景 | 通用 |
| 测试目的: |  | | |
| 预置条件: |  | | |
| 测试步骤: | 1、从创建的数据表中，查询所有数据  2、top k 查询  3、join 查询 | | |
| SQL语句 |  | | |
| 预期结果: | 返回ok | | |
| 测试结果: |  | | |

## 5.6 导出文件到本地

|  |  |  |  |
| --- | --- | --- | --- |
| 项目: | 功能测试 | 分项目: | Table Insert |
| 用例编号: | TESTHIVE\_TABLE\_LOAD | 版本: |  |
| 用例分数 |  | 适用场景 | 通用 |
| 测试目的: | 从本地及HDFS中导入数据到表格 | | |
| 预置条件: |  | | |
| 测试步骤: | 将上一步中创建的数据表，导出到本地 | | |
| SQL语句 |  | | |
| 预期结果: | 返回ok | | |
| 测试结果: |  | | |