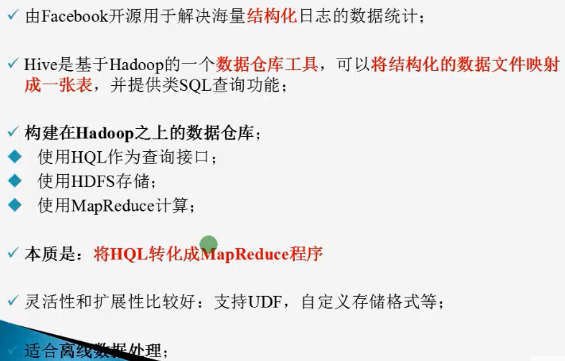
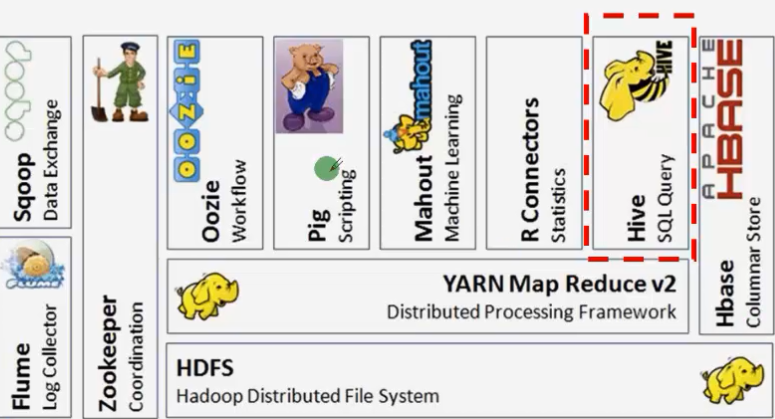
Hive

# Hive 简介

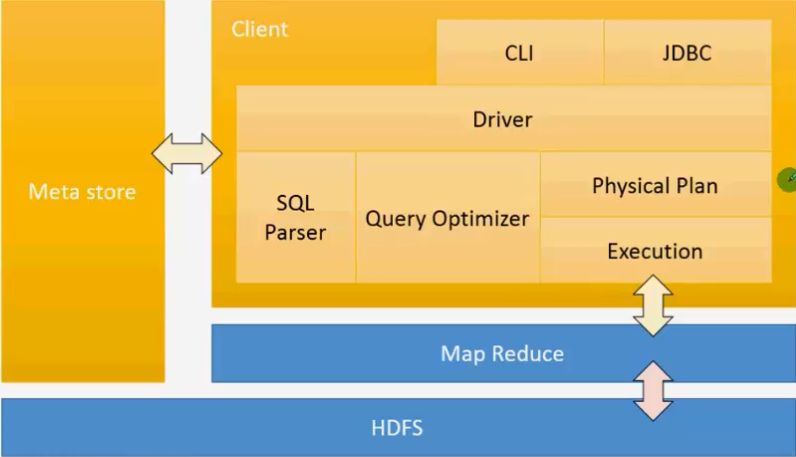
## 什么是Hive



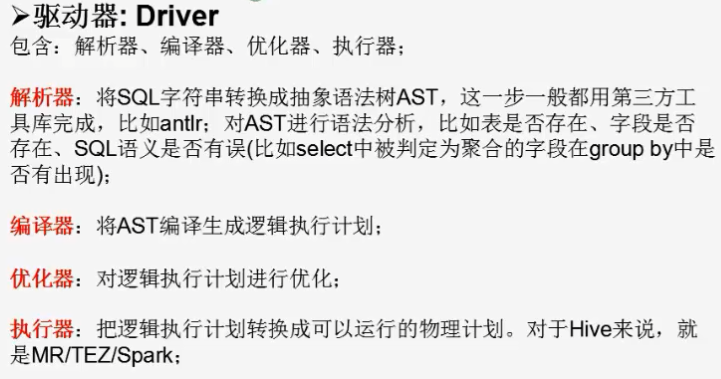
Hive 在生态系统的位置：



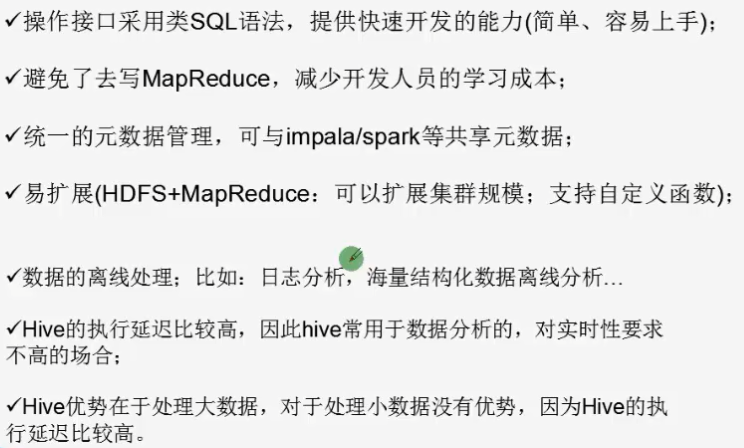
## Hive架构







## Hive 优点与使用场景



#### Compile Hive on master

To build the current Hive code from the master branch:

$ git clone https://git-wip-us.apache.org/repos/asf/hive.git

$ cd hive

$ mvn clean package -Pdist

$ cd packaging/target/apache-hive-{version}-SNAPSHOT-bin/apache-hive-{version}-SNAPSHOT-bin

$ ls

## Hive 运行三种方式

1. 命令行

### Web

问题1：ls: cannot access /opt/cdh/hive-0.13.1-cdh5.3.6/lib/hive-hwi-\*.war: No such file or directory

解决方案：

[beifeng@bigdata-spark01 hive-0.13.1-cdh5.3.6]$ cd hwi/

[beifeng@bigdata-spark01 hwi]$ ls

hive-hwi.iml pom.xml src web

[beifeng@bigdata-spark01 hwi]$ jar cvfM0 hive-0.13.1.war -C web/ .

[beifeng@bigdata-spark01 hwi]$ ls

hive-0.13.1.war hive-hwi.iml pom.xml src web

[beifeng@bigdata-spark01 hwi]$ cp hive-0.13.1.war /opt/cdh/hive-0.13.1-cdh5.3.6/lib/

修改配置文件：搜索 Hive Web Interface--->Configuration

<property>

<name>hive.hwi.listen.host</name>

<value>bigdata-spark01.ibeifeng.com</value>

</property>

<property>

<name>hive.hwi.listen.port</name>

<value>9999</value>

</property>

<property>

<name>hive.hwi.war.file</name>

<value>lib/hive-hwi-0.13.1.war</value>

</property>

[beifeng@bigdata-spark01 hive-0.13.1-cdh5.3.6]$ bin/hive --service hwi

问题2:Perhaps JAVA\_HOME does not point to the JDK.It is currently set to "/opt/modules/jdk1.7.0\_67/jre"

解决方案：

cp /opt/modules/jdk1.7.0\_67/lib/tools.jar /opt/cdh/hive-0.13.1-cdh5.3.6/lib/

重启服务：

[beifeng@bigdata-spark01 hive-0.13.1-cdh5.3.6]$ bin/hive --service hwi

访问：

http://bigdata-spark01.ibeifeng.com:9999/hwi/

### 远程服务

应用场景：

以JDBC or ODBC的程序登录到hive中操作数据的时候，必须要选用这种远程服务启动方式

启动方式：

bin/hive --service hiverserver &

# Hive 环境搭建



<https://cwiki.apache.org/confluence/display/Hive/Home;jsessionid=6FF03F77C1DFE3AFC97651F27966A128>

<https://cwiki.apache.org/confluence/display/Hive/GettingStarted>

<https://github.com/apache/hive>

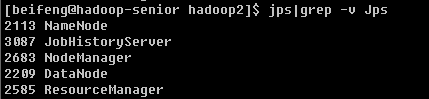
## 基分伪分布式的安装

[beifeng@hadoop-senior hadoop2]$ sbin/start-dfs.sh

[beifeng@hadoop-senior hadoop2]$ sbin/start-yarn.sh

[beifeng@hadoop-senior hadoop2]$ sbin/hadoop-daemon.sh stop secondarynamenode

[beifeng@hadoop-senior hadoop2]$ sbin/mr-jobhistory-daemon.sh start historyserver



### 解压hive

[beifeng@hadoop-senior softwares]$ tar -zxvf apache-hive-0.13.1-bin.tar.gz -C /opt/modules/

[beifeng@hadoop-senior modules]$ mv apache-hive-0.13.1-bin/ hive-0.13.1

[beifeng@hadoop-senior modules]$ ln -s /opt/modules/hive-0.13.1/ hive

### 安装配置

[beifeng@hadoop-senior conf]$ cp hive-env.sh.template hive-env.sh

HADOOP\_HOME=/opt/modules/hadoop-2.5.0

HIVE\_CONF\_DIR=/opt/modules/hive-0.13.1/conf

In addition, you must create /tmp and /user/hive/warehouse (aka hive.metastore.warehouse.dir) and set them chmod g+w in HDFS before you can create a table in Hive.

Commands to perform this setup:

$ $HADOOP\_HOME/bin/hadoop fs -mkdir /tmp

$ $HADOOP\_HOME/bin/hadoop fs -mkdir /user/hive/warehouse

$ $HADOOP\_HOME/bin/hadoop fs -chmod g+w /tmp

$ $HADOOP\_HOME/bin/hadoop fs -chmod g+w /user/hive/warehouse

#### Hive数据仓库位置配置

default

/user/hive/warehouse

注意事项

\* 在仓库目录下，没有对默认的数据库default创建文件夹

\* 如果某张表属于default数据库，直接在数据仓库目录下创建一个文件夹

<property>

<name>hive.metastore.warehouse.dir</name>

<value>/user/hive/warehouse</value>

</property>

### HIVE CLI

-d,–define <key=value> 定义一个变量值，这个变量可以在Hive交互Shell中引用，后面会介绍用法，比如：-d A=B

–database <databasename> 进入Hive交互Shell时候指定数据库，默认进入default数据库

-e <quoted-query-string> 命令行执行一段SQL语句

-f <filename> filename文件中保存HQL语句，执行其中的语句

-H,–help 显示帮助信息

-h <hostname> 连接远程Hive Server，后续介绍

–hiveconf <property=value> 在命令行中设置Hive的运行时配置参数，优先级高于hive-site.xml,但低于Hive交互Shell中使用Set命令设置。

–hivevar <key=value> 同—define

-i <filename> 进入Hive交互Shell时候先执行filename中的HQL语句

-p <port> 连接远程Hive Server的端口号

-S,–silent 静默模式，指定后不显示执行进度信息，最后只显示结果

-v,–verbose 冗余模式，额外打印出执行的HQL语句

[beifeng@hadoop-senior hive-0.13.1]$ bin/hive

建表，字段的分隔符为制表符：

create table student(id int, name string) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

导入数据：

hive> load data local inpath '/opt/datas/student.txt' into table student;

### mysql安装

[root@hadoop-senior mysql-libs]# rpm -qa|grep mysql

mysql-libs-5.1.66-2.el6\_3.x86\_64

[root@hadoop-senior mysql-libs]# rpm -e --nodeps mysql-libs-5.1.66-2.el6\_3.x86\_64

[root@hadoop-senior mysql-libs]# rpm -ivh MySQL-server-5.6.24-1.el6.x86\_64.rpm

[root@hadoop-senior mysql-libs]# cat /root/.mysql\_secret

[root@hadoop-senior mysql-libs]# rpm -ivh MySQL-client-5.6.24-1.el6.x86\_64.rpm

修改root用户的密码

mysql> set PASSWORD=PASSWORD('root');

mysql> update mysql.user set host='%' where host='localhost';

mysql> select user,host,host,password from mysql.user;

### 配置hive metastore

拷贝mysql驱动jar包，到Hive安装目录的lib下

[beifeng@hadoop-senior mysql-connector-java-5.1.27]$ cp mysql-connector-java-5.1.27-bin.jar /opt/modules/hive-0.13.1/lib/

[beifeng@hadoop-senior conf]$ cp hive-default.xml.template hive-site.xml

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:mysql://hadoop-senior.ibeifeng.com:3306/metastore?createDatabaseIfNotExist=true</value>

<description>JDBC connect string for a JDBC metastore</description>

</property>

<property>

<name>javax.jdo.option.ConnectionDriverName</name>

<value>com.mysql.jdbc.Driver</value>

<description>Driver class name for a JDBC metastore</description>

</property>

<property>

<name>javax.jdo.option.ConnectionUserName</name>

<value>root</value>

<description>username to use against metastore database</description>

</property>

<property>

<name>javax.jdo.option.ConnectionPassword</name>

<value>root</value>

<description>password to use against metastore database</description>

</property>

<property>

<name>hive.cli.print.header</name>

<value>true</value>

<description>Whether to print the names of the columns in query output.</description>

</property>

<property>

<name>hive.cli.print.current.db</name>

<value>true</value>

<description>Whether to include the current database in the Hive prompt.</description>

</property>

### 元数据介绍

表名 说明 关联键

TBLS 所有hive表的基本信息 TBL\_ID,SD\_ID

TABLE\_PARAM 表级属性，如是否外部表，表注释等 TBL\_ID

COLUMNS Hive表分区信息表字段信息(字段注释，字段名，字段类型，字段序号)SD\_ID

SDS 所有hive表、表分区所对应的hdfs数据目录和数据格式 SD\_ID,SERDE\_ID

SERDE\_PARAM 序列化反序列化信息，如行分隔符、列分隔符、NULL的表示字符等 SERDE\_ID

PARTITIONS Hive表分区信息 PART\_ID,SD\_ID,TBL\_ID

PARTITION\_KEYS Hive分区表分区键 TBL\_ID

PARTITION\_KEY\_VALS Hive表分区名(键值) PART\_ID

TBLS表:

元数据表字段 说明 示例数据

TBL\_ID 表ID 1

CREATE\_TIME 创建时间 1436317071

DB\_ID 数据库ID 2，对应DBS中的DB\_ID

LAST\_ACCESS\_TIME 上次访问时间 1436317071

OWNER 所有者 liuxiaowen

RETENTION 保留字段 0

SD\_ID 序列化配置信息 86，对应SDS表中的SD\_ID

TBL\_NAME 表名 lxw1234

TBL\_TYPE 表类型 MANAGED\_TABLE、EXTERNAL\_TABLE、INDEX\_TABLE、VIRTUAL\_VIEW

VIEW\_EXPANDED\_TEXT 视图的详细HQL语句 select `lxw1234`.`pt`, `lxw1234`.`pcid` from `liuxiaowen`.`lxw1234`

VIEW\_ORIGINAL\_TEXT 视图的原始HQL语句 select \* fr

COLUMNS\_V2表:

元数据表字段 说明 示例数据

CD\_ID 字段信息ID 1

COMMENT 字段注释

COLUMN\_NAME 字段名 pt

TYPE\_NAME 字段类型 string

INTEGER\_IDX 字段顺序 2

### Hive基本操作

show databases ;

create database db\_hive ;

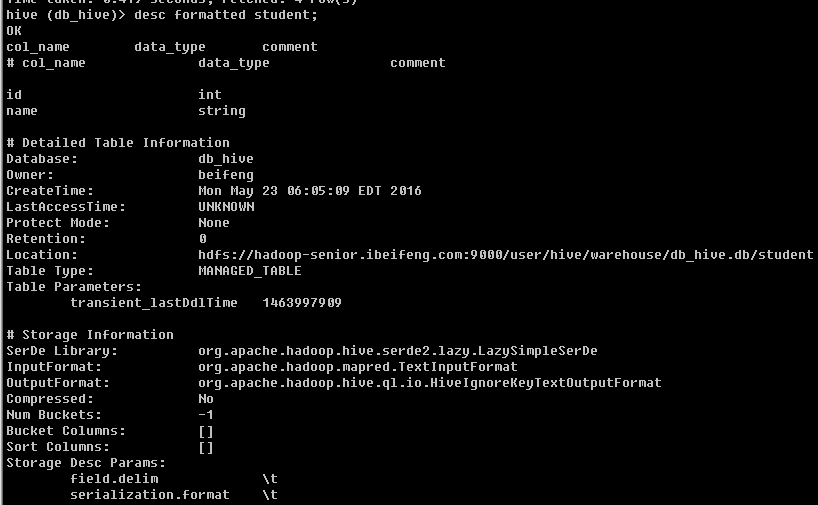
create table student(id int, name string) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

show tables ;

desc student ;

desc extended student ;

desc formatted student ;



use db\_hive ;

load data local inpath '/opt/datas/student.txt'into table db\_hive.student ;

show functions ;

desc function upper ;

desc function extended upper ;

select id ,upper(name) uname from db\_hive.student ;

### 日志配置

$HIVE\_HOME/conf/hive-log4j.properties

hive.log.dir=/opt/modules/hive-0.13.1/logs

hive.log.file=hive.log命令行显示日志信息：

[beifeng@hadoop-senior hive]$ bin/hive --hiveconf hive.root.logger=INFO,console

### Hive 命令帮助

[beifeng@hadoop-senior hive-0.13.1]$ bin/hive -help

usage: hive

-d,--define <key=value> Variable subsitution to apply to hive

commands. e.g. -d A=B or --define A=B

--database <databasename> Specify the database to use

-e <quoted-query-string> SQL from command line

-f <filename> SQL from files

-H,--help Print help information

-h <hostname> connecting to Hive Server on remote host

--hiveconf <property=value> Use value for given property

--hivevar <key=value> Variable subsitution to apply to hive

commands. e.g. --hivevar A=B

-i <filename> Initialization SQL file

-p <port> connecting to Hive Server on port number

-S,--silent Silent mode in interactive shell

-v,--verbose Verbose mode (echo executed SQL to the

console)

\* bin/hive -e <quoted-query-string>

eg:

bin/hive -e "select \* from db\_hive.student ;"

\* bin/hive -f <filename>

eg:

$ touch hivef.sql

select \* from db\_hive.student ;

$ bin/hive -f /opt/datas/hivef.sql

$ bin/hive -f /opt/datas/hivef.sql > /opt/datas/hivef-res.txt

\* bin/hive -i <filename>

与用户udf相互使用

在hive cli命令窗口中如何查看hdfs文件系统

hive (default)> dfs -ls / ;

在hive cli命令窗口中如何查看本地文件系统

hive (default)> !ls /opt/datas ；

# Hive深入使用

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual>

## 建库建表

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL>

create table IF NOT EXISTS default.bf\_log\_20150913(

ip string COMMENT 'remote ip address' ,

user string ,

req\_url string COMMENT 'user request url')

COMMENT 'BeiFeng Web Access Logs'

ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '

STORED AS TEXTFILE ;

load data local inpath '/opt/datas/bf-log.txt' into table default.bf\_log\_20150913;

create table IF NOT EXISTS default.bf\_log\_20150913\_sa

AS select ip,req\_url from default.bf\_log\_20150913 ;

create table IF NOT EXISTS default.bf\_log\_20150914

like default.bf\_log\_20150913 ;

create database if not exists db\_hive\_01;

create database if not exists db\_hive\_02 location '/user/ibeifeng/hive/warehouse/db\_hive\_02.db';

use db\_hive ;

desc database db\_hive\_03 ;

desc database extended db\_hive\_03 ;

drop database db\_hive\_03 ;

drop database db\_hive\_03 cascade;

drop database if exists db\_hive\_03 ;

## 示例

员工表

create table IF NOT EXISTS default.emp(

empno int,

ename string,

job string,

mgr int,

hiredate string,

sal double,

comm double,

deptno int

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

部门表

create table IF NOT EXISTS default.dept(

deptno int,

dname string,

loc string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

导入数据

load data local inpath '/opt/datas/emp.txt' overwrite into table emp ;

load data local inpath '/opt/datas/dept.txt' overwrite into table dept ;

create table if not exists default.dept\_cats

as

select \* from dept ;

truncate table dept\_cats ;

create table if not exists default.dept\_like

like

default.dept ;

alter table dept\_like rename to dept\_like\_rename ;

drop table if exists dept\_like\_rename ;

在Hive中表的类型

\* 管理表

\* 托管表（外部表）

## 外部表



create EXTERNAL table IF NOT EXISTS default.emp\_ext2(

empno int,

ename string,

job string,

mgr int,

hiredate string,

sal double,

comm double,

deptno int

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

location '/user/beifeng/hive/warehouse/emp\_ext2';

## 分区表



bf\_log

/user/hive/warehouse/bf\_log/

/20150911/

20150911.log

/20150912/

20150912.log

create EXTERNAL table IF NOT EXISTS default.emp\_partition(

empno int,

ename string,

job string,

mgr int,

hiredate string,

sal double,

comm double,

deptno int

)

partitioned by (month string,day string)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' ;

load data local inpath '/opt/datas/emp.txt' into table default.emp\_partition partition (month='201509',day='13') ;

### 注意事项非分区表

create table IF NOT EXISTS default.dept\_nopart(

deptno int,

dname string,

loc string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

dfs -put /opt/datas/dept.txt /user/hive/warehouse/dept\_nopart ;

select \* from dept\_nopart ;

### -----------分区表---------------

create table IF NOT EXISTS default.dept\_part(

deptno int,

dname string,

loc string

)

partitioned by (day string)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

第一种方式

dfs -mkdir -p /user/hive/warehouse/dept\_part/day=20150913 ;

dfs -put /opt/datas/dept.txt /user/hive/warehouse/dept\_part/day=20150913 ;

hive (default)> msck repair table dept\_part ;

第二种方式

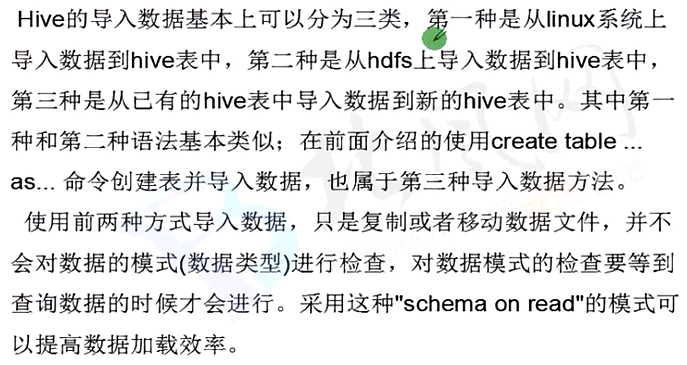
dfs -mkdir -p /user/hive/warehouse/dept\_part/day=20150914 ;

dfs -put /opt/datas/dept.txt /user/hive/warehouse/dept\_part/day=20150914 ;

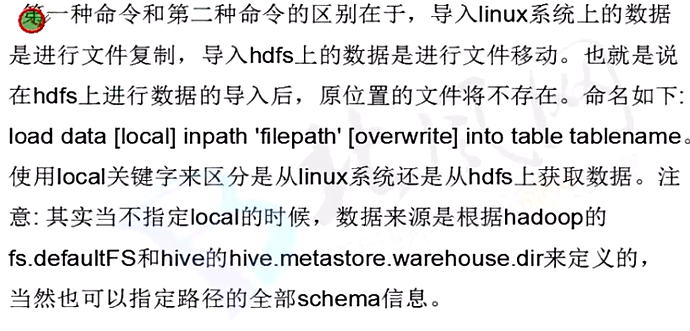
alter table dept\_part add partition(day='20150914');

show partitions dept\_part ;

## Hive 导入导出数据



### 导入



LOAD DATA [LOCAL] INPATH 'filepath'

[OVERWRITE] INTO TABLE tablename

[PARTITION (partcol1=val1, partcol2=val2 ...)]

load data [local] inpath 'filepath' [overwrite] into table tablename [partition (partcol1=val1,...)];

\* 原始文件存储的位置

\* 本地 local

\* hdfs

\* 对表的数据是否覆盖

\* 覆盖 overwrite

\* 追加

\* 分区表加载，特殊性

partition (partcol1=val1,...)

1）加载本地文件到hive表

load data local inpath '/opt/datas/emp.txt' into table default.emp ;

2）加载hdfs文件到hive中

load data inpath '/user/beifeng/hive/datas/emp.txt' overwrite into table default.emp ;

3）加载数据覆盖表中已有的数据

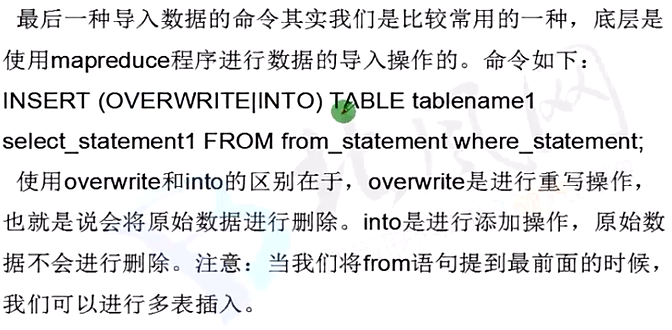
load data inpath '/user/beifeng/hive/datas/emp.txt' into table default.emp ;

4）创建表是通过insert加载

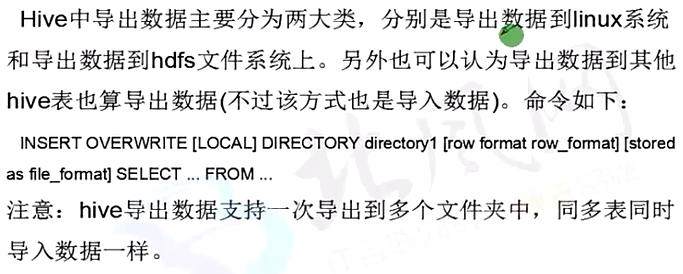
create table default.emp\_ci like emp ;

insert into table default.emp\_ci select \* from default.emp ;

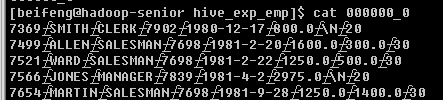
5）创建表的时候通过location指定加载 见前文



### 导出



insert overwrite local directory '/opt/datas/hive\_exp\_emp' select \* from default.emp ;



insert overwrite directory '/user/beifeng/hive/hive\_exp\_emp' select \* from default.emp ;

**同时导出到本地和HDFS**

hive> from (select students.\* from students left semi join classes on students.classid = classes.classid) as tmp insert overwrite local directory '/home/hadoop/hive/student' select \* insert overwrite directory '/tmp/students' select \*;

### Import/Export

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+ImportExport>

Export

导出，将Hive表中的数据，导出到外部

Import

导入，将外部数据导入Hive表中

EXPORT TABLE default.emp TO '/user/beifeng/hive/export/emp\_exp' ;

export\_target\_path：

指的是HDFS上路径

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db\_name.]table\_name

LIKE existing\_table\_or\_view\_name

[LOCATION hdfs\_path];

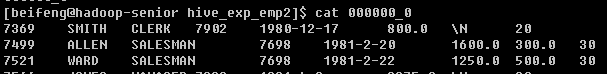
create table db\_hive.emp like default.emp ;

import table db\_hive.emp from '/user/beifeng/hive/export/emp\_exp';

#### 导出指定分隔符

insert overwrite local directory '/opt/datas/hive\_exp\_emp2'

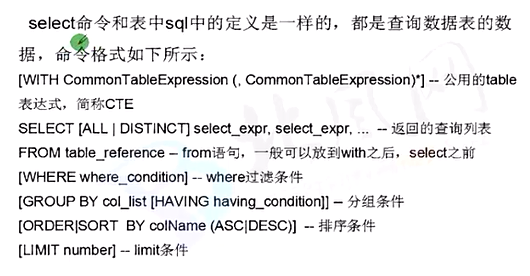
ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t' COLLECTION ITEMS TERMINATED BY '\n' select \* from default.emp ;

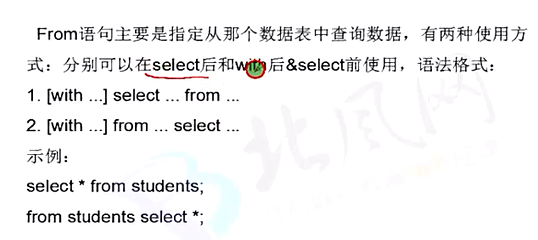


bin/hive -e "select \* from default.emp ;" > /opt/datas/exp\_res.txt

## 查询

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Select>





### CTE



cte语法：

1. 获取班级号为1的学生信息:

with tmp as (select studentid as sid,classid as cid,studentname as name from students where classid=1) from tmp select \*;

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]

group by /having

分组

emp表

\* 每个部门的平均工资

select t.deptno, avg(t.sal) avg\_sal from emp t group by t.deptno ;

\* 每个部门中每个岗位的做高薪水

select t.deptno, t.job, max(t.sal) avg\_sal from emp t group by t.deptno, job ;

>>>having

\* where 是针对单条记录进行筛选

\* having 是针对分组结果进行筛选

求每个部门的平均薪水大于2000的部门

select deptno, avg(sal) from emp group by deptno ;

select deptno, avg(sal) avg\_sal from emp group by deptno having avg\_sal > 2000;

#### >> order by

对全局数据的一个排序，仅仅只有1个reduce

select \* from emp order by empno desc ;

#### >> sort by

对每一个reduce内部数据进行排序的，全局结果集来说不是排序 局部排序

set mapreduce.job.reduces= 3;

select \* from emp sort by empno asc ;

insert overwrite local directory '/opt/datas/sortby-res' select \* from emp sort by empno asc ;

#### >> distribute by

分区partition

类似于MapReduce中分区partition,对数据进行分区，结合sort by进行使用

insert overwrite local directory '/opt/datas/distby-res' select \* from emp distribute by deptno sort by empno asc ;

注意事项：

distribute by 必须要在sort by 前面。

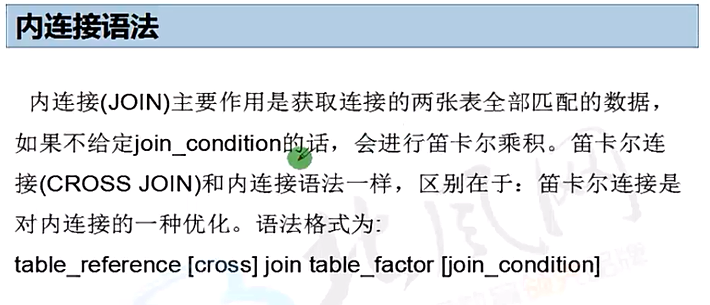
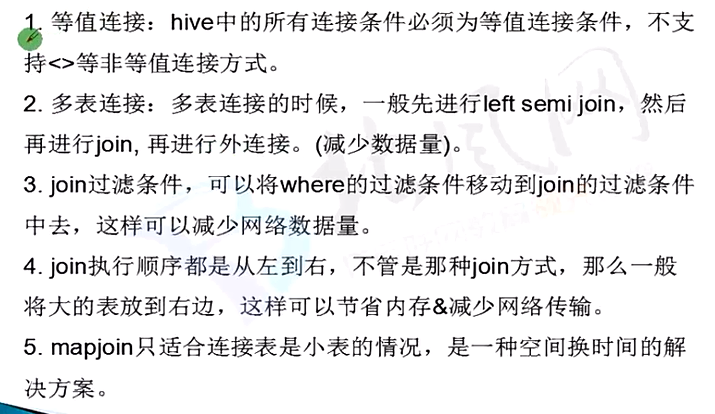
#### >> cluster by

当distribute by和sort by 字段相同时，可以使用cluster by ;

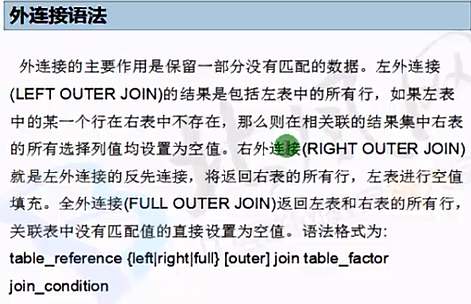
insert overwrite local directory '/opt/datas/cluster-res' select \* from emp cluster by empno ;

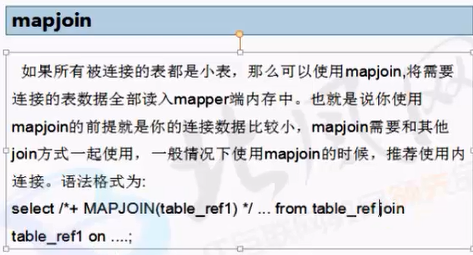
### JOIN





hive> select students.\*, classes.\* from classes cross join students on classes.classid=students.classid;





## Hive UDF (User Defined Function)

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF>







pom.xml

<!-- Hive Client -->

<dependency>

<groupId>org.apache.hive</groupId>

<artifactId>hive-jdbc</artifactId>

<version>${hive.version}</version>

</dependency>

<dependency>

<groupId>org.apache.hive</groupId>

<artifactId>hive-exec</artifactId>

<version>${hive.version}</version>

</dependency>

Creating Custom UDFs

1)First, you need to create a new class that extends UDF, with one or more methods named evaluate.

2)Usage

add jar /opt/datas/hiveudf.jar ;

create temporary function my\_lower as "com.beifeng.senior.hive.udf.LowerUDF" ;

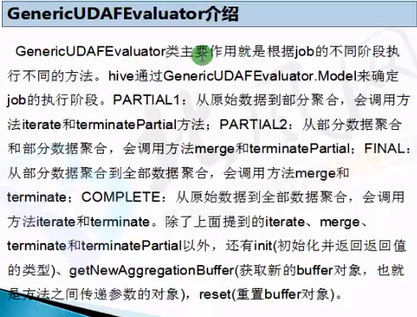
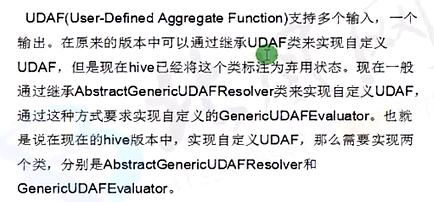
select ename, my\_lower(ename) lowername from emp limit 5 ;



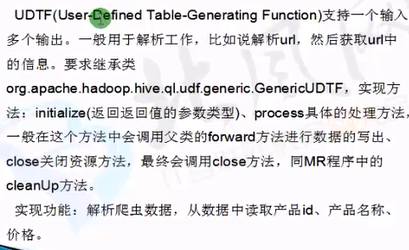
hive (default)> create function myfunc as 'com.vip.hive.udf.LowerUDF' USING JAR 'hdfs:///user/beifeng/hive/jars/hiveudf.'/hive/jars/hiveudf.jar';

select ename, self\_lower(ename) lowername from emp limit 5 ;

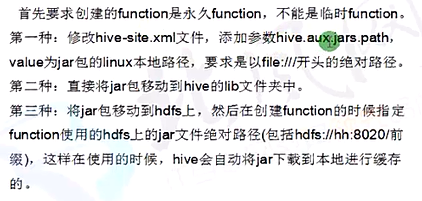
## UDAF



## UDTF



## hive集成自定义函数的各种不同的方式



1. 修改hive-site.xml

<property>

<name>hive.aux.jars.path</name>

<value>file:///home/hadoop/jobs/beifeng14-0.0.1.jar</value>

</property>

**最常用的：使用将jar上传到hdfs中，然后再进行操作的方式。**

1. 上传文件到hdfs: dfs -put /home/hadoop/jobs/beifeng14-0.0.1.jar /beifeng/beifeng14-0.0.1.jar

2. 创建函数:create function f2 as 'com.beifeng.hive.ql.UDTFCase' using jar 'hdfs://hh:8020/beifeng/beifeng14-0.0.1.jar';

3. 正常使用



# Hive高级使用

## HiveSever2

<https://cwiki.apache.org/confluence/display/Hive/HiveServer2+Clients>

bin/hiveserver2

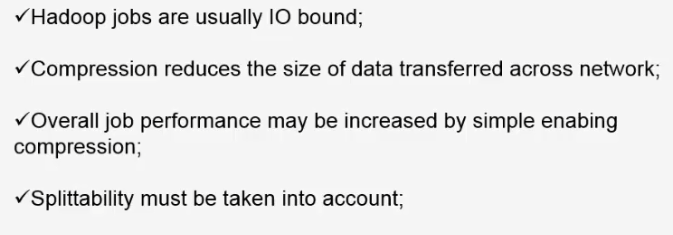
bin/beeline

!connect jdbc:hive2://hadoop-senior.ibeifeng.com:10000 beifeng beifeng org.apache.hive.jdbc.HiveDriver



bin/beeline -u jdbc:hive2://hadoop-senior.ibeifeng.com:10000/default

## 数据压缩



mapreduce 默认不支持压缩，需要自己编译hadoop 源码进行安装

1. 安装snappy

下载源码 到官网

<http://code.google.com/p/snappy/>

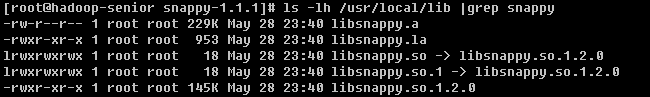
或者到 [https://github.com/google/snappy](https://github.com/google/snappy%20)下载源码，目前版本为 1.1.1。 2.2、编译安装 解压 tar -zxvf  snappy-1.1.1.tar.gz ,然后以 root 用户 执行标准的三步进行编译安装

[root@hadoop-senior snappy-1.1.1]# ./configure

[root@hadoop-senior snappy-1.1.1]# make

[root@hadoop-senior snappy-1.1.1]# make install

[root@hadoop-senior snappy-1.1.1]# ls -lh /usr/local/lib |grep snappy



<http://chaozi204.github.io/blog/hadoop-v2-snappy-install/>

1. 编译haodop 2.x源码

编译前安装上 yum install -y bzip2-devel命令

[beifeng@hadoop-senior native]$ mvn clean package -Pdist,native -DskipTests -Dtar -Drequire.snappy -Dbundle.snappy -Dsnappy.lib=/usr/local/lib

/opt/modules/hadoop-2.5.0-src/hadoop-dist/target/hadoop-2.5.0/lib/native

[beifeng@hadoop-senior lib]$ pwd

/opt/modules/hadoop2/lib

[beifeng@hadoop-senior lib64]$ cp -a /opt/modules/hadoop-2.5.0-src/hadoop-dist/target/hadoop-2.5.0/lib/native .

[beifeng@hadoop-senior hadoop-2.5.0]$ bin/hadoop checknative

15/08/31 23:10:16 INFO bzip2.Bzip2Factory: Successfully loaded & initialized native-bzip2 library system-native

15/08/31 23:10:16 INFO zlib.ZlibFactory: Successfully loaded & initialized native-zlib library

Native library checking:

hadoop: true /opt/modules/hadoop-2.5.0/lib/native/libhadoop.so

zlib: true /lib64/libz.so.1

snappy: true /opt/modules/hadoop-2.5.0/lib/native/libsnappy.so.1

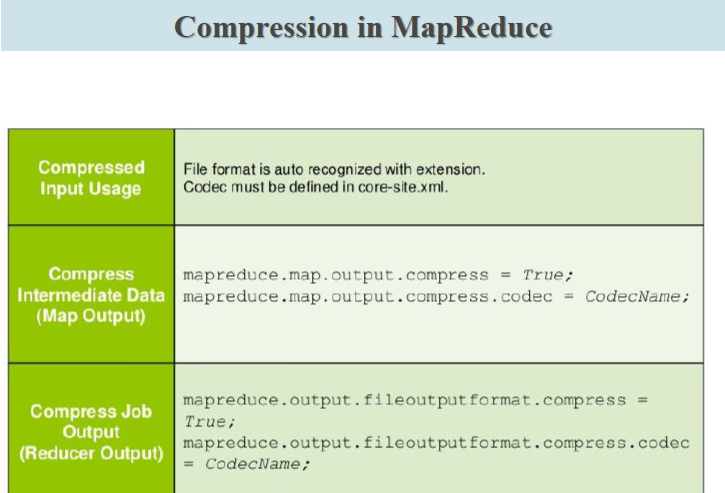
lz4: true revision:99

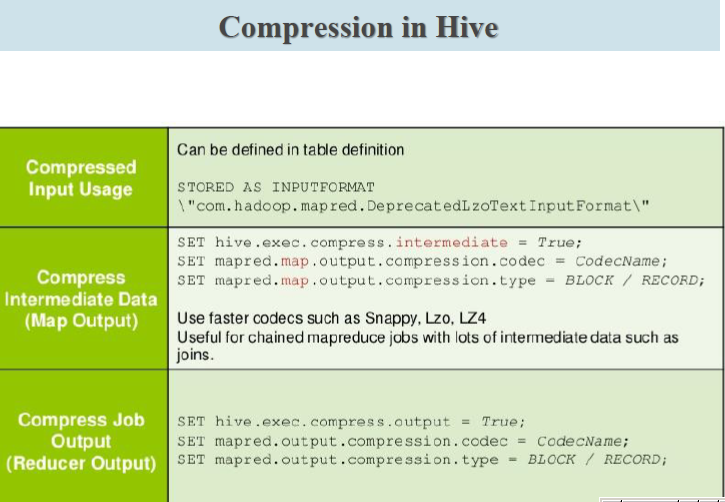
bzip2: true /lib64/libbz2.so.1

>>>>>>>>>>>>>>

bin/yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount /user/beifeng/mapreduce/wordcount/input /user/beifeng/mapreduce/wordcount/output

bin/yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.0.jar wordcount -Dmapreduce.map.output.compress=true -Dmapreduce.map.output.compress.codec=org.apache.hadoop.io.compress.SnappyCodec /user/beifeng/mapreduce/wordcount/input /user/beifeng/mapreduce/wordcount/output22





## 数据存储

<https://cwiki.apache.org/confluence/display/Hive/SerDe>

file\_format:

  :

| SEQUENCEFILE

  | TEXTFILE    -- (Default, depending on hive.default.fileformat configuration)

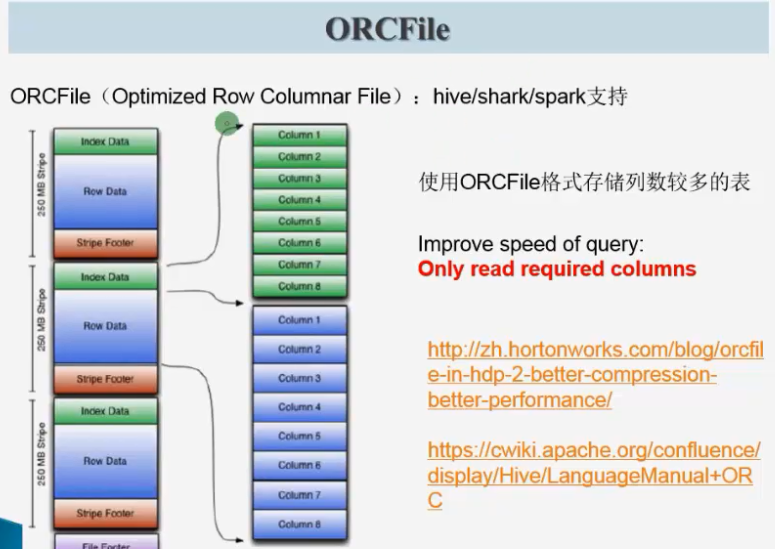
  | RCFILE      -- (Note: Available in Hive 0.6.0 and later)

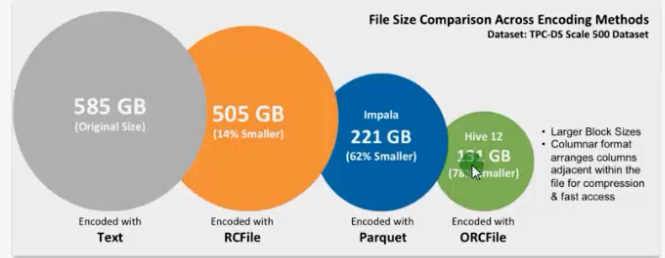
  | ORC         -- (Note: Available in Hive 0.11.0 and later)

  | PARQUET     -- (Note: Available in Hive 0.13.0 and later)

  | AVRO        -- (Note: Available in Hive 0.14.0 and later)

  | INPUTFORMAT input\_format\_classname OUTPUTFORMAT output\_format\_classname





数据存储

\* 按行存储数据

\* 按列存储数据

### 实例

create table page\_views(

track\_time string,

url string,

session\_id string,

referer string,

ip string,

end\_user\_id string,

city\_id string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS TEXTFILE ;

load data local inpath '/opt/datas/page\_views.data' into table page\_views ;

dfs -du -h /user/hive/warehouse/page\_views/ ;

>>>>>>>orc

create table page\_views\_orc(

track\_time string,

url string,

session\_id string,

referer string,

ip string,

end\_user\_id string,

city\_id string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS orc ;

insert into table page\_views\_orc select \* from page\_views ;

dfs -du -h /user/hive/warehouse/page\_views\_orc/ ;

2.6 M /user/hive/warehouse/page\_views\_orc/000000\_0

>>>>>>>> parquet

create table page\_views\_parquet(

track\_time string,

url string,

session\_id string,

referer string,

ip string,

end\_user\_id string,

city\_id string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS PARQUET ;

insert into table page\_views\_parquet select \* from page\_views ;

dfs -du -h /user/hive/warehouse/page\_views\_parquet/ ;

13.1 M /user/hive/warehouse/page\_views\_parquet/000000\_0

select session\_id,count(\*) cnt from page\_views group by session\_id order by cnt desc limit 30 ;

select session\_id,count(\*) cnt from page\_views\_orc group by session\_id order by cnt desc limit 30 ;

select session\_id from page\_views limit 30 ;

select session\_id from page\_views\_orc limit 30 ;

select session\_id from page\_views\_parquet limit 30 ;

========================================================

create table page\_views\_orc\_snappy(

track\_time string,

url string,

session\_id string,

referer string,

ip string,

end\_user\_id string,

city\_id string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS orc tblproperties ("orc.compress"="SNAPPY");

insert into table page\_views\_orc\_snappy select \* from page\_views ;

dfs -du -h /user/hive/warehouse/page\_views\_orc\_snappy/ ;

--------------

create table page\_views\_orc\_none(

track\_time string,

url string,

session\_id string,

referer string,

ip string,

end\_user\_id string,

city\_id string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS orc tblproperties ("orc.compress"="NONE");

insert into table page\_views\_orc\_none select \* from page\_views ;

--------------------

set parquet.compression=SNAPPY ;

create table page\_views\_parquet\_snappy(

track\_time string,

url string,

session\_id string,

referer string,

ip string,

end\_user\_id string,

city\_id string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS parquet;

insert into table page\_views\_parquet\_snappy select \* from page\_views ;

dfs -du -h /user/hive/warehouse/page\_views\_parquet\_snappy/ ;

总结：

在实际的项目开发当中，hive表的数据

\* 存储格式

orcfile / qarquet

\* 数据压缩

snappy

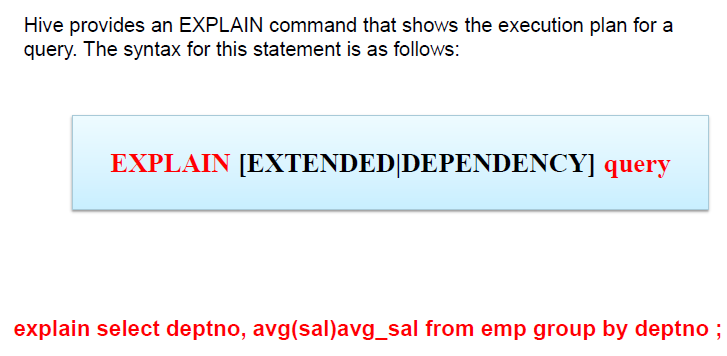
## 企业优化

http://shiyanjun.cn/archives/588.html





### Explain Execution Plan

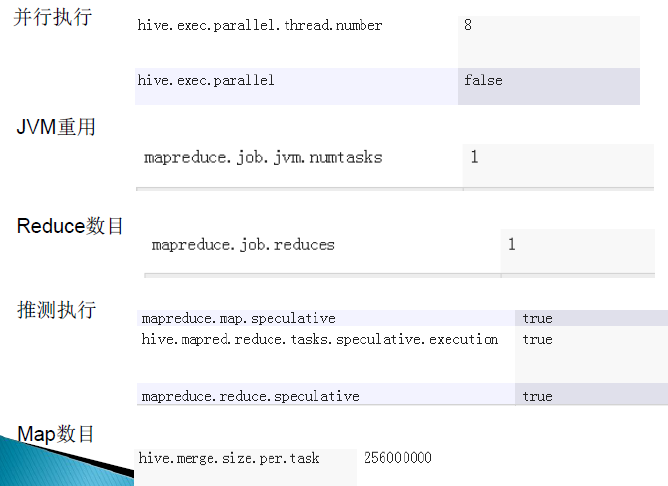


EXPLAIN select \* from emp ;

EXPLAIN select deptno,avg(sal) avg\_sal from emp group by deptno ;

EXPLAIN EXTENDED select deptno,avg(sal) avg\_sal from emp group by deptno ;

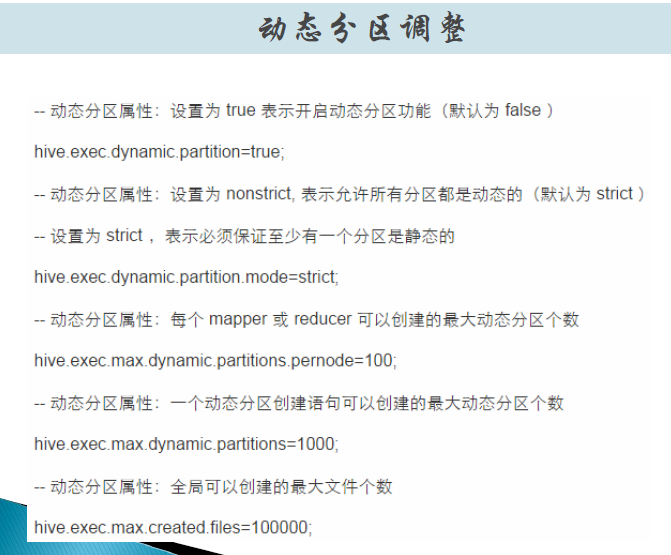
一个stage 就是一个mapreduce 任务



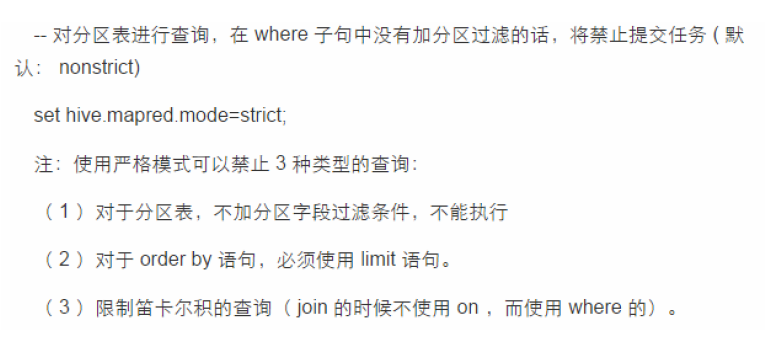
并行度10-20之间

JVM 重用不要超过9个

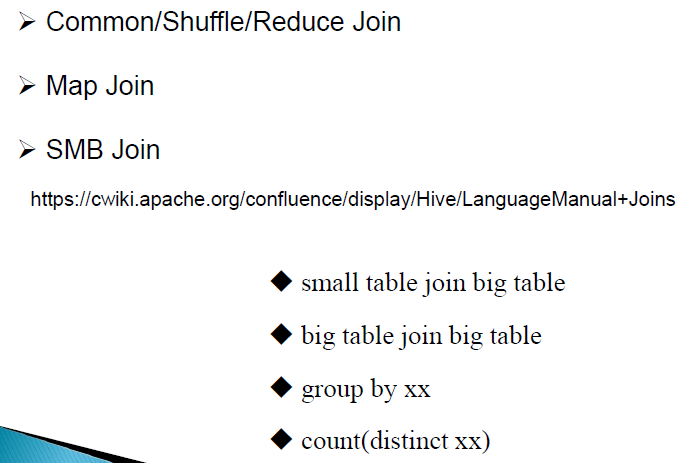
### 动态分区调整



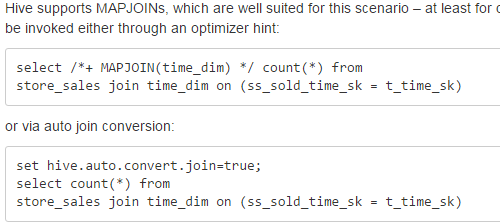
### StrictMode



<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Joins>



### Prior Support for MAPJOIN



### Auto Conversion to SMB Map Join

set hive.auto.convert.sortmerge.join=true;

set hive.optimize.bucketmapjoin = true;

set hive.optimize.bucketmapjoin.sortedmerge = true;

## 实战案例

依据业务数据表

方式一、原始表bf\_log\_src，加载数据（预先处理）

方式二、创建正则表RegexSerDe

https://cwiki.apache.org/confluence/display/Hive/GettingStarted

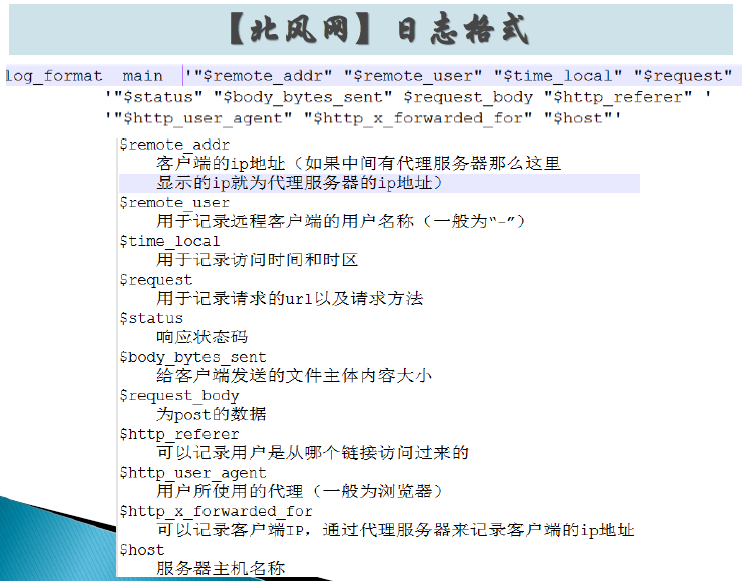
http://wpjam.qiniudn.com/tool/regexpal/

数据ETL

拆分表（子表）、数据存储格式

数据预处理ETL（udf、python）

数据分析HQL



### 思路

\* 原表

\* 针对不同的业务创建不同的子表

\* 数据存储格式orcfile/parquet

\* 数据压缩 snappy

\* map output 数据压缩 snappy

\* 外部表

\* 分区表（演示）

### 创建源表

create table IF NOT EXISTS default.bf\_log\_src (

remote\_addr string,

remote\_user string,

time\_local string,

request string,

status string,

body\_bytes\_sent string,

request\_body string,

http\_referer string,

http\_user\_agent string,

http\_x\_forwarded\_for string,

host string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '

stored as textfile ;

load data local inpath '/opt/datas/moodle.ibeifeng.access.log' into table default.bf\_log\_src ;

select count(\*) from bf\_log\_src ;

select \* from bf\_log\_src limit 5 ;

>>>>>>>>>>>>>>>

drop table if exists default.bf\_log\_src ;

create table IF NOT EXISTS default.bf\_log\_src (

remote\_addr string,

remote\_user string,

time\_local string,

request string,

status string,

body\_bytes\_sent string,

request\_body string,

http\_referer string,

http\_user\_agent string,

http\_x\_forwarded\_for string,

host string

)

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'

WITH SERDEPROPERTIES (

"input.regex" = "(\"[^ ]\*\") (\"-|[^ ]\*\") (\"[^\]]\*\") (\"[^\"]\*\") (\"[0-9]\*\") (\"[0-9]\*\") (-|[^ ]\*) (\"[^ ]\*\") (\"[^\"]\*\") (-|[^ ]\*) (\"[^ ]\*\")"

)

STORED AS TEXTFILE;

load data local inpath '/opt/datas/moodle.ibeifeng.access.log' into table default.bf\_log\_src ;

### 业务需求表1

>>>>>>>>>>>>>>>>>>>>>>>>>>>

drop table if exists default.bf\_log\_comm ;

create table IF NOT EXISTS default.bf\_log\_comm (

remote\_addr string,

time\_local string,

request string,

http\_referer string

)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS orc tblproperties ("orc.compress"="SNAPPY");

insert into table default.bf\_log\_comm select remote\_addr, time\_local, request,http\_referer from default.bf\_log\_src ;

select \* from bf\_log\_comm limit 5 ;

### 数据清洗

============

定义UDF，对原表数据进行清洗

第一个udf

去除引号

public class RemoveQuotoUDF extends UDF {

public Text evaluate(final Text s) {

if (s == null) { return new Text(); }

return new Text(s.toString().replaceAll("\"", ""));

}

public static void main(String[] args) {

System.out.println(new RemoveQuotoUDF().evaluate(new Text("HIVE UDF")));

}

}

hive (default)> add jar /opt/datas/removeQuotoUDF.jar;

hive (default)> create temporary function my\_removequotes as "com.vip.hive.udf.RemoveQuotoUDF";

insert overwrite table default.bf\_log\_comm select my\_removequotes(remote\_addr), my\_removequotes(time\_local), my\_removequotes(request), my\_removequotes(http\_referer) from default.bf\_log\_src ;

select \* from bf\_log\_comm limit 5 ;

============

第二个 UDF

处理日期时间字段

31/Aug/2015:00:04:37 +0800

20150831000437

hive (default)> add jar /opt/datas/DateUDF.jar;

hive (default)> create temporary function my\_datetransform as "com.vip.hive.udf.DateProcessUDF";

insert overwrite table default.bf\_log\_comm select my\_removequotes(remote\_addr), my\_datetransform(my\_removequotes(time\_local)), my\_removequotes(request), my\_removequotes(http\_referer) from default.bf\_log\_src ;

select \* from bf\_log\_comm limit 5 ;

select substring('20150831230437',9,2) hour from bf\_log\_comm limit 1 ;

select t.hour, count(\*) cnt from

(select substring(time\_local,9,2) hour from bf\_log\_comm ) t

group by t.hour order by cnt desc ;

select t.prex\_ip, count(\*) cnt from

(

select substring(remote\_addr,1,7) prex\_ip from bf\_log\_comm

) t

group by t.prex\_ip order by cnt desc limit 20 ;

### Python

<https://cwiki.apache.org/confluence/display/Hive/GettingStarted>

CREATE TABLE u\_data (

userid INT,

movieid INT,

rating INT,

unixtime STRING)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

STORED AS TEXTFILE;

LOAD DATA LOCAL INPATH '/opt/datas/ml-100k/u.data' OVERWRITE INTO TABLE u\_data;

-------

CREATE TABLE u\_data\_new (

userid INT,

movieid INT,

rating INT,

weekday INT)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';

add FILE /opt/datas/ml-100k/weekday\_mapper.py;

INSERT OVERWRITE TABLE u\_data\_new

SELECT

TRANSFORM (userid, movieid, rating, unixtime) -- input from source table

USING 'python weekday\_mapper.py' -- script

AS (userid, movieid, rating, weekday) --output from python

FROM u\_data;

INSERT OVERWRITE TABLE u\_data\_new

SELECT

TRANSFORM (userid, movieid, rating, unixtime)

USING 'python weekday\_mapper.py'

AS (userid, movieid, rating, weekday)

FROM u\_data;

SELECT weekday, COUNT(\*) FROM u\_data\_new GROUP BY weekday;

SELECT weekday, COUNT(1) cnt FROM u\_data\_new GROUP BY weekday order by cnt desc;

----

import sys

import datetime

for line in sys.stdin:

line = line.strip()

userid, movieid, rating, unixtime = line.split('\t')

weekday = datetime.datetime.fromtimestamp(float(unixtime)).isoweekday()

print '\t'.join([userid, movieid, rating, str(weekday)])