**hive是什么**  
 1.hive建立在Hadoop上的数据仓库基础架构。是一种可以存储，查询和分析存储在Hadoop中的大规模数据的机制。  
 2.hive是sql的解析引擎，他将sql语句转译成M/R job 在Hadoop上执行。  
**hive的系统结构和数据存储**  
 hive是数据仓库（不是数据库）。  
 hive定义了简单的SQL查询语言HQL(sql方言)，同时hive允许属性mapreduce的开发者使用mapper和reducer来处理分析工作。             
**系统结构：**  
**1.用户接口:**  
 CLI,JDBC/ODBC,WebGUI分别对应，shell命令，Java访问，浏览器访问  
**2.元数据存储**  
 hive将元数据存储在MySQL，derby(默认)，主要存储的是表的名字，表的列分区，表的属性（外部表还是受控表），表所在的目录等存储数据。  
**3.解释器，编译器，优化器，执行器**  
  完成hql查询语句，从词法分析，语法分析，编译，优化以及查询计划的生成，将生成的查询计划存储在hdfs中，由mapredce调用。  
**4.hadoop**  
 hive的数据存储在hdfs中，大部分查询由mapreduce完成                          
**hive的数据存储**  
  hive的数据存储基于Hadoop HDFS存储结构主要包括：数据库，文件，表视图，索引hive默认可以直接加载文本文件（TextFile）,还支持SequenceFile,RCFile创建表时可以指定数据的列和行的分割符。  
**hive的安装**  
  由于默认的derby数据库一次只能打开一个会话，不方便所有，使用MySQL。                  
**2、安装Hive**  
  前提是：hadoop必须已经启动了  
  **1°、解压hive的安装包**

tar -zxvf apache-hive-0.14.0-bin.tar.gz -C ../  
  **2°、备份配置文件**  
  cp hive-env.sh.template hive-env.sh  
  cp hive-default.xml.template hive-site.xml  
  **3°、配置hive的配置文件**  
   1）、修改**hive-env.sh**  
      加入三行内容  
     JAVA\_HOME=/usr/local/jdk1.7.0\_55  
     HADOOP\_HOME=/usr/local/hadoop-2.4.1  
     HIVE\_HOME=/usr/local/hive-0.14.0  
   2）、修改**hive-site.xml**  
     <property>  
        <name>javax.jdo.option.ConnectionURL</name>  
        <value>jdbc:mysql://ip:3306/hiveexample?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>root</value>  
     </property>  
     <property>  
        <name>javax.jdo.option.ConnectionPassword</name>  
        <value>root</value>  
     </property>  
     <property>  
        <name>hive.querylog.location</name>  
        <value>/usr/local/hive-0.14.0/tmp</value>  
        </property>  
     <property>  
        <name>hive.exec.local.scratchdir</name>  
        <value>/usr/local/hive-0.14.0/tmp</value>  
     </property>  
     <property>  
        <name>hive.downloaded.resources.dir</name>  
        <value>/usr/local/hive-0.14.0/tmp</value>  
     </property>  
**4°、拷贝mysql驱动到$HIVE\_HOME/lib目录下**  
  cp /usr/local/soft/mysql-connector-java-5.1.39.jar $HIVE\_HOME/lib/  
**5°、启动Hive(可以加入环境变量)**  
   脚本文件方式 hive   
   命令行方式cli：控制台模式 hive --service cli  
   JDBC方式：hiveserver  （hive --service hiveserver2）  
   web GUI接口 hwi方式  
**hive的数据类型**  
   **表中如何加载数据：**  
   （1）、从本地文件系统中导入数据到Hive表；  
   （2）、从HDFS上导入数据到Hive表；  
   （3）、从别的表中查询出相应的数据并导入到Hive表中；  
   （4）、在创建表的时候通过从别的表中查询出相应的记录并插入到所创建的表中。  
hive数据在Hadoop中是目录文件的形式。  
 load data local inpath 'linux-path' into table tblName;  
hive> load data local inpath 'hadoop路径下文件或者linux路径文件' into table t;  
也可以使用hadoop fs -put linux-path hive-table-dir  
列的默认分隔符：\001  
hive> insert into table test select id, name, tel from wyp;  
hive> create table test11 as select id, name, tel from wyp;  
**数据的加载模式**  
   **读模式**  
     就是说数据在被加载的时候，**不**对数据的合法性进行**校验**，当在查询的时候，将不合法的数据显示为NULL。  
     好处：数据加载的速度快，适合加载大数据。  
   **写模式**  
     就是说数据在被加载的时候，**需要**对其合法性进行**校验**，校验失败，则加载失败，在数据中的数据都是合法的。  
     好处：适合关系型数据的使用，对数据的准确性要求高。  
**Hive中MR的本地模式**  
     可以缩短mr的执行时间。但是有大小限制 128M  
     set hive.exec.mode.local.auto=true;  
 在进行测试进行使用。                  
**Hive数据加载和导出**[]表示可选，<>表示必须  
 **数据的加载**  
  load data [local] inpath 'path' [overwrite] into table tblName [partition partition\_sepc];  
  **local:有：**数据从**linux**本地path去加载数据  
     **无：**数据到**hdfs**的相关目录path下去加载数据，相当于hadoop fs -mv   
  **overwrite:有**：删除原来的数据，**新增**一份  
  **无**：在原来的基础上**追加**一份  
**从其他的表加载数据**  
   insert <overwrite|into> table t8\_bucket select id from t7\_partition where dt="2016-07-25";  
**在创建表的时候加载数据**  
   create table tblName as select cols from oldTbl;  
   另外的：可以通过like来快速的复制一张表的结构  
       eg. create table t10\_b like t1;  
**动态分区的数据的加载**  
   要使用动态分区，那必须要**开启动态分区模式**，将其设置为**nonstrict**                  
hive>set hive.exec.dynamic.partition=true;  
hive>set hive.exec.dynamic.partition.mode=nostrict;  
hive>set hive.exec.max.dynamic.partitions.pernode=1000;          
需要注意：前后两张表字段个数一定要对应。  
insert into table t10\_c\_d\_p partition(dt) select id from t7\_partition ;  
insert into table t10\_d\_d\_p partition(year=2015,school) select id, name, school from t7\_partition\_1 where year=2015;  
insert overwrite table t10\_d\_d\_p partition(school, year) select \* from t7\_partition\_1;          
**对同一份数据的多次处理：**  
hive> from sales\_log  
hive> insert into table sales select \* where dt="2015-07-08";  
hive> insert into table history select \* where dt="2015-07-08";  
**导出：**  
  1、hdfs目录级别的拷贝：hadoop fs -cp hdfs\_uri\_1 hdfs\_uri\_2;  
  2、使用insert overwrite directory命令  
  INSERT OVERWRITE [LOCAL] DIRECTORY '...' SELECT ...FROM...WHERE ...;  
  Export：export tblName to 'hdfs\_path';  
  export table t1 to '/export/';  
  Import：import table from 'hdfs\_path';   
  import table t1\_1 from "/export/";：  
**Hive中hql--->mr**  
**hql没有转化为mr**  
    如果说将hql能够转化为hadoop hdfs的shell就不会产生mr。eg select \* from tbl[ partition partition\_sepc] [limit x];  
**hql转化为了mr**  
   hadoop jar xxx.jar input output  
**Hive中如何控制MR**  
  **Map**  
   inputSplit/block.size  
   hadoop:10M 20M 30M-->3  
   hive:10M 20M 30M-->3 CombineHiveInputFormat  
     对小文件进行了合并，减少maptask的数量，提高了map的执行效率  
     10M 20M 100M==130M-->1个，**阈值 1.1倍也就是140.8M**   
    1000M/128 = 7  
  **Reduce**  
   主要由：hive.exec.reducers.bytes.per.reducer  
   hive.exec.reducers.max以及mapred.reduce.tasks  
   <property>  
      <name>hive.exec.reducers.bytes.per.reducer</name>  
      <value>256000000</value>  
     <description>size per reducer.The default is 256Mb, i.e if the input size is 1G, it will use 4 reducers.</description>  
   </property>  
   <property>  
      <name>hive.exec.reducers.max</name>  
      <value>1009</value>  
      <description>  
       max number of reducers will be used. If the one specified in the configuration parameter mapred.reduce.tasks is negative, Hive will use this one as the max number of reducers when automatically determine number of reducers.  
      </description>  
   </property>  
**hive.exec.parallel并行执行MR**  
**hive.exec.parallel.thread.number一次最多执行的job数量**  
                   
**只有一个Reducer的情况**  
  进行统计没有写group by  
  笛卡尔积(A join B on A.xx=B.yy)A:M,B:N M\*N  
  在我们进行排序的使用order by  
**Hive中的排序**  
  select \* from t1 order by id desc;  
  **order by**是一个全局的排序，会产生一个reducer  
  **sort by**是单节点的排序  
   distribute by <col> sort by <col>能够达到order的效果  
  **cluster by** <col>是对上面distribute by sort by的缩写 -->不支持asc和desc  
**Hive中文件的类型**  
    <property> 默认为textfile  
      通过 hive-site.xml 配置文件的配置项  
      <name>hive.default.fileformat</name>  
      <value>TextFile</value>  
      <description>  
          Expects one of [textfile, sequencefile, rcfile, orc].Default file format for CREATE TABLE statement. Users can explicitly override it by CREATE TABLE ... stored as [FORMAT]  
      </description>  
    </property>  
**开启压缩：**  
   set hive.exec.compress.output=true;  
   set mapred.output.compress=true;

set mapred.output.compression.codec=org.apache.hadoop.io.compress.GzipCodec;  
                set io.compression.codec=org.apache.hadoop.io.compress.GzipCodec;  
   set mapred.output.compression.type=BLOCK;  
**TextFile**  
   Hive 默认格式，数据不做压缩，磁盘开销大，数据解析开销大。  
   可结合 Gzip、Bzip2、Snappy 等使用（系统自动检查，执行查询时自动解压），但使用这种方式，  
    hive 不会对数据进行切分，从而无法对数据进行并行操作。  
**SequenceFile**   
   是 Hadoop API 提供的一种二进制文件支持，其具有使用方便、可分割、可压缩的特点。  
   SequenceFile 支持三种压缩选择：NONE，RECORD，BLOCK。Record 压缩率低，一般建议使用 BLOCK 压缩。  
**RCFILE**   
   是一种行列存储相结合的存储方式。首先，其将数据按行分块，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。其次，块数据列式存储，有利于数据压缩和快速的列存取  
**Orc**   
   文件格式是对 RCFile 的一个扩展和升级，拥有更高效的压缩比和执行效率。Parquet 是新起的一种列式存储文件类型  
  
**几种压缩方式笔记：**  
   **textfile** 存储空间消耗比较大，并且压缩的 text 无法分割和合并查询的效率最低,可以直接存储，加载数据的速度最高  
   **sequencefile** 存储空间消耗大,压缩的文件可以分割和合并 查询效率高，需要通过tex文件转化来加载  
   **rcfile** 存储空间最小，查询的效率最高 ，需要通过 text 文件转化来加载，加载的速度最低相比 TEXTFILE 和 SEQUENCEFILE，RCFILE 由于列式存储方式，数据加载时性能消耗较大，但是具有较好的压缩比和查询响应。数据仓库的特点是一次写入、多次读取，因此，  
**整体来看，RCFILE 相比其余两种格式具有较明显的优势。**  
   **通过比较：**使用压缩可以提高hdfs的存储能力，还有加快我们查询效率。  
   在工作中常用压缩方式rc和orc          
          
**Hive中的内嵌函数**   
   show functions查看函数  
   desc function functionName;  
**hive自定义函数**  
**udf步骤：**  
   1.写一个类继承(org.apache.hadoop.hive.ql.)UDF  
   2.覆盖方法eturn evaluate(param...)  
   3.打jar包，提交hive服务器  
   4.添加到hive资源里面(add jar jar的路径)--->临时的加入的hive的classpath  
  5.创建一个临时的函数create temparay function functionName as '自定义类的全类名';  
   6.执行相应的操作  
   7[可选]、drop 临时函数  
**hive的JDBC**  
增删改查  
**hive权限管理**：控制一些人有权限做一些事，一些人没权限做一些事  
用户、组、角色  
为了使用权限管理，需要开启权限认证：set hive.security.authorization.enabled=true;  
开启权限认证的时候，我们再创建表的时候，就会提示：No privilege 'Create'等相应权限，  
**授予权限：**  
 grant create on <database|table> <dbName|tblName> to user root;  
查看当前系统用户：set system:user.name;  
查看当前用户拥有的权限：show <user|group|role> <uname|gname|rname> on <database|table> <dbName|tblName>;  
**组级别的权限**  
授予权限：grant select on table t\_group to group root;  
**角色级别的权限**  
如果没有角色，则需要创建相应的角色：create role roleName;  
将相关的用户或者添加到角色里面：grant role roleName to <user|group> <uname|gname>;  
将相关的操作权限赋予给角色:grant <select|create...> on <database|table> <dbName|tblName> to role roleName;  
**Hive中有哪些常见的权限：**  
ALL------所有的操作  
Create---创建数据库或表  
DROP-----删除数据库或表或分区  
SELECT---查询表数据  
UPDATE---load  
ALTER----对表结构的修改  
SHOW\_DATABASE---查看数据库  
**剔除权限：**  
revoke <create|drop|select|update|...> on <database|table> <dbName|tblName> from <user|group|role> <uname|gname|rname>;  
**分区表权限的空**  
要做到更细粒度的权限控制，要开启如下属性：  
alter table t\_partition set tblproperties("PARTITION\_LEVEL\_PRIVILEGE"="TRUE");  
为了能够操作相关的分区，要对相应的分区进行权限的授予  
grant <create|drop|select|update|...> on table tblName partition(partition\_sepc) to <user|group|role> <uname|gname|rname>;  
**hive 调优**  
**1.EXPLAIN**   
 EXPLAIN EXTENDED   
 使用EXPLAIN

可以hive对sql的查询计划信息解析  
  EXPLAIN SELECT COUNT(1) FROM T1;  
 hive将查询转为mapreduce执行，所以优化hive可以转化为优化mr.  
 根据EXPLAIN展示的信息可以做出优化  
**2.限制调整，**  
使用order by;sort by ;distribute by ;cluster by ;limit返回部分结果  
**3.jvm重用**  
jvm执行map和reduce任务，启动jvm过程开销很大，  
开启jvm重用可以使得jvm实例在同一个job中重用n次，n在配置文件中设置。  
**4.其他**  
join优化；小表在左边  
本地模式；并行执行，动态分区，严格模式等  
hive的调优本质上是对hive-default.xml中参数的优化。  
  
**数据倾斜：**  
shuffle阶段优化过程中，会遇到数据倾斜。  
如何将map输出的数据均匀分配到reduce中是解决数据倾斜的根本。  
**原因:**1.key分布不均匀  
   2.业务数据本身的特性  
   3.建表时考虑不周全  
   4.sql本身就数据倾斜  
**表现：**任务长时间在99%不动。单一reduce的记录数与平均记录数差异过大。  
**解决方案：**参数调节,进行负载均衡  
hive.map.aggr=true  
map端聚合，相当于combiner  
hive.groupby.skewindata=true  
保证相同的group by key分布到同一个reduce  
sql语句调节  
join，map join  
**业务场景：**  
日志中信息丢失，空值产生数据倾斜  
**1.**为空的字段不参与关联  
**2.**给空值赋值,io和作业数都少了，空值key赋值字符串加上随机数  
**3.**日志信息中不同字段类型关联产生数据倾斜  
例如：当按照用户id进行两表join操作时，默认hash操作会按int类型的id进行分配，  
这样string类型的id的记录都会分配到一个reducer中，  
解决办法将数字类型转换为字符串类型  
cast(a.user\_id as string)  
  
**hive面试常见问题**  
1.Hive内外部表的区别  
2.Hive如何做到权限管理  
3.Hive的数据倾斜和调优  
4.Hive文件压缩格式有哪些？压缩效率如何  
5.说说对桶表的理解  
6.Hive解析hql转化为MR的执行过程  
7.Hive的分组排序(row\_number())，组内TopN  
8.hive有哪些保存元数据的方式，有什么特点  
9.hive能见多个库吗  
10.hive的底层怎么与数据库交互的。  
11.hive实现统计的查询语句以及与mr的区别  
12.对于hive你写过哪些UDF函数，作用是干什么的  
13.hive与hbase的区别  
14.hive中的压缩格式textfile, sequencefile, rcfile, orc的区别  
15.hive的工作原理  
16.编写hive自定义函数，实现oracle数据库中的addmonths函数功能，然后封装到hive函数库中。  
   Addmonths(data a,int b)求传入日期a经过b个月后的日期。  
17.另外看到一道很不错的面试题