**2. Hive 查询语法**

**2.1. SELECT**

SELECT [ALL | DISTINCT] select\_expr, select\_expr, ...

FROM table\_reference

[WHERE where\_condition]

[GROUP BY col\_list [HAVING condition]]

[CLUSTER BY col\_list | [DISTRIBUTE BY col\_list]

[SORT BY| ORDER BY col\_list] ]

[LIMIT number]

1. order by 会对输入做全局排序，因此只有一个reducer，会导致当输入规模较大时，需要较长的计算时间。
2. sort by不是全局排序，其在数据进入reducer前完成排序。因此，如果用sort by进行排序，并且设置 mapred.reduce.tasks>1，则sort by只保证每个reducer的输出有序，不保证全局有序。
3. distribute by(字段)根据指定的字段将数据分到不同的reducer，且分发算法是hash散列。
4. Cluster by(字段) 除了具有Distribute by的功能外，还会对该字段进行排序。 —> distribute by + sort by

因此，如果分桶和sort字段是同一个时，此时， cluster by = distribute by + sort by 分桶表的作用：最大的作用是用来提高join操作的效率；

思考这个问题： select a.id,a.name,b.addr from a join b on a.id = b.id; 如果a表和b表已经是分桶表，而且分桶的字段是id字段 做这个join操作时，还需要全表做笛卡尔积吗?

**2.2. 查询语法**

**全表查询**

select \* from score;

**选择特定列**

select s\_id ,c\_id from score;

**列别名** 1）重命名一个列。 2）便于计算。 3）紧跟列名，也可以在列名和别名之间加入关键字‘AS’

select s\_id as myid ,c\_id from score;

**2.3. 常用函数**

* 求总行数

（count） select count(1) from score;

* 求分数的最大值

（max） select max(s\_score) from score;

* 求分数的最小值

（min） select min(s\_score) from score;

* 求分数的总和

（sum） select sum(s\_score) from score;

* 求分数的平均值

（avg） select avg(s\_score) from score;

**2.4. LIMIT语句**

典型的查询会返回多行数据。LIMIT子句用于限制返回的行数。

select \* from score limit 3;

**2.5. WHERE语句**

1. 使用WHERE 子句，将不满足条件的行过滤掉。
2. WHERE 子句紧随 FROM 子句。
3. 案例实操 查询出分数大于60的数据

select \* from score where s\_score > 60;

**2.6 比较运算符**

* 查询分数等于80的所有的数据

select \* from score where s\_score = 80;

* 查询分数在80到100的所有数据

select \* from score where s\_score between 80 and 100;

* 查询成绩为空的所有数据

select \* from score where s\_score is null;

* 查询成绩是80和90的数据

select \* from score where s\_score in(80,90);

**2.7 LIKE 和 RLIKE**

1. 使用LIKE运算选择类似的值
2. 选择条件可以包含字符或数字:

% 代表零个或多个字符(任意个字符)。

\_ 代表一个字符。

1. RLIKE子句是Hive中这个功能的一个扩展，其可以通过Java的正则表达式这个更强大的语言来指定匹配条件。

**2.8 案例实操**

1. 查找以8开头的所有成绩

select \* from score where s\_score like '8%';

1. 查找第二个数值为9的所有成绩数据

select \* from score where s\_score like '\_9%';

1. 查找成绩中含9的所有成绩数据

select \* from score where s\_score rlike '[9]'; # like '%9%'

**2.9 逻辑运算符**

1. 查询成绩大于80，并且s\_id是01的数据

select \* from score where s\_score >80 and s\_id = '01';

1. 查询成绩大于80，或者s\_id 是01的数

select \* from score where s\_score > 80 or s\_id = '01';

1. 查询s\_id 不是 01和02的学生

select \* from score where s\_id not in ('01','02');

**2.10. 分组 GROUP BY 语句**

GROUP BY语句通常会和聚合函数一起使用，按照一个或者多个列队结果进行分组，然后对每个组执行聚合操作。

**案例实操**

1. 计算每个学生的平均分数

select s\_id ,avg(s\_score) from score group by s\_id;

1. 计算每个学生最高成绩

select s\_id ,max(s\_score) from score group by s\_id;

**HAVING 语句**

* having与where不同点
  + where针对表中的列发挥作用，查询数据；having针对查询结果中的列发挥作用，筛选数据。
  + where后面不能写分组函数，而having后面可以使用分组函数。
  + having只用于group by分组统计语句。

**案例实操：**

1. 求每个学生的平均分数

select s\_id ,avg(s\_score) from score group by s\_id;

1. 求每个学生平均分数大于85的人

select s\_id ,avg(s\_score) avgscore from score group by s\_id having avgscore > 85;