**大数据SQL面试题**

## SQL试题一

使用SQL查询a，b表中不相交的数据集。

a表：

|  |  |
| --- | --- |
| id | name |
| 1 | zhangsan |
| 2 | lisi |
| 3 | wangwu |
| 4 | maliu |
| 5 | tianqi |

b表：

|  |  |
| --- | --- |
| id | name |
| 1 | zhangsan |
| 2 | lisi |
| 3 | wangwu |
| 6 | zhaoqi |
| 7 | gaoba |

数据附件

|  |
| --- |
| create table a(id int ,name string) row format delimited fields terminated by '\t';  create table b(id int ,name string) row format delimited fields terminated by '\t';  select b.id,b.name from a full join b on a.id = b.id where a.id is null  union  select a.id,a.name from a full join b on a.id = b.id where b.id is null |

## SQL试题二

使用SQL根据表A，表B 计算出表C：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 表A | | | date | v1 | | 2019/1/1 | 74 | | 2019/1/2 | 54 | | 2019/1/3 | 19 | | 2019/1/6 | 84 | | 2019/1/7 | 2 | | 2019/1/8 | 78 | | 2019/1/9 | 19 | | 2019/1/10 | 95 | | |  |  | | --- | --- | | 表B | | | date | V2 | | 2019/1/1 | 13 | | 2019/1/2 | 3 | | 2019/1/3 | 29 | | 2019/1/4 | 81 | | 2019/1/5 | 37 | | 2019/1/8 | 56 | | 2019/1/9 | 53 | | 2019/1/10 | 50 | | |  |  |  | | --- | --- | --- | | 表C | | | | date | V1 | V2 | | 2019/1/1 | 74 | 13 | | 2019/1/2 | 54 | 3 | | 2019/1/3 | 19 | 29 | | 2019/1/4 | 0 | 81 | | 2019/1/5 | 0 | 37 | | 2019/1/6 | 84 | 0 | | 2019/1/7 | 2 | 0 | | 2019/1/8 | 78 | 56 | | 2019/1/9 | 19 | 53 | | 2019/1/10 | 95 | 50 | |

数据附件：

|  |
| --- |
| create table a(`date` string,v1 int) row format delimited fields terminated by '\t';  create table b(`date` string,v2 int) row format delimited fields terminated by '\t';  select nvl(a.date,b.date) as dt1,nvl(a.v1,0) as v1,nvl(b.v2,0) as v2 from a full join b on a.date = b.date |

## SQL试题三

有一个拉链表：存款利率表 bal\_rat\_table，数据如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| id | bal | rate | startDate | endDate |
| 1 | 100 | 0.1 | 2018-12-01 | 2019-02-06 |
| 2 | 100 | 0.1 | 2018-09-01 | 2019-03-05 |
| 1 | 150 | 0.1 | 2019-02-06 | 2019-04-30 |
| 3 | 200 | 0.1 | 2019-01-04 | 2019-03-25 |
| 4 | 300 | 0.1 | 2019-01-01 | 9999-12-31 |
| 1 | 200 | 0.1 | 2019-04-30 | 9999-12-31 |
| 2 | 200 | 0.1 | 2019-03-05 | 9999-12-31 |
| 3 | 250 | 0.1 | 2019-03-25 | 9999-12-31 |

id：用户id 、bal:用户存款、rate:利率、startDate:开始时间、endDate:结束日期，9999代表不知道何时结束。

需求:拉链表中存在多种状态的数据（存款），开始日期和结束日期不一致，求出2019年第一个季度：2019-01-01至2019-03-31的每个人的利息。

数据附件：

|  |
| --- |
| create table bal\_rat\_table(id int,bal int,rate double,startdate string,enddate string) row format delimited fields terminated by '\t';  SELECT  id,  sum(  bal \* rate \* datediff(  IF (  enddate > '2019-03-31',  '2019-03-31',  enddate  ),  IF (  startdate < '2019-01-01',  '2019-01-01',  startdate  )  )  ) AS total\_sum  FROM  bal\_rat\_table  WHERE  enddate > '2019-01-01'  AND startdate < '2019-03-31'  GROUP BY  id; |

## SQL试题四

已知：有如下table（名syc\_mianshi），含有三个字段，姓名Name(string)，消费时间DT（string），消费金额Money(Double)。记录条数有若干行。

|  |  |  |
| --- | --- | --- |
| NAME | DT | MONEY |
| 张三 | 2015/10/4 | 300 |
| 张三 | 2015/4/16 | 200 |
| 张三 | 2015/5/9 | 600 |
| 李四 | 2015/1/17 | 2500 |
| 李四 | 2016/2/9 | 3700 |
| 王五 | 2015/5/12 | 0.4 |
| 王五 | 2015/12/23 | 1.23 |
| 王五 | 2014/3/20 | 0.56 |

计算每个人在哪一天的消费金额最大，即应输出：

|  |  |
| --- | --- |
| 张三 | 2015/5/9 |
| 李四 | 2016/2/9 |
| 王五 | 2015/12/23 |

数据附件：

|  |
| --- |
| create table syc\_mianshi (name string,dt string,money double) row format delimited fields terminated by '\t';  注意：row\_number() over(partition by col1 order by col2 ) as rank  按照col1 分组，在每个组内按照col2排序从1打标签。  rank() over(partition by col1 order by col2 ):相同数据标号相同，不连续  dense\_rank() over(partition by col1 order by col2 ) :相同数据标号相同，连续  select name,dt from (select name,dt,money,row\_number() over(partition by name order by money desc ) as rank from syc\_mianshi) t where rank = 1; |

## SQL试题五

数据转换：

|  |  |  |
| --- | --- | --- |
| dt | cn | point |
| 1 | abc | 10 |
| 1 | def | 15 |
| 1 | xyz | 20 |
| 2 | abc | 13 |
| 2 | xyz | 40 |
| 3 | def | 50 |
| 3 | abc | 60 |
| ... | ... | ... |

查询转换成如下结构:

|  |  |  |
| --- | --- | --- |
| cn | dt | sumpoint |
| abc | 1 | 10 |
| def | 1 | 15 |
| xyz | 1 | 20 |
| abc | 2 | 23 |
| xyz | 2 | 60 |
| def | 3 | 65 |
| abc | 3 | 83 |
| ... | ... | ... |

数据附件：

|  |
| --- |
| create table t1(dt string,cn string,point int) row format delimited fields terminated by '\t';  select cn,dt,point,sum(point) over (partition by cn order by dt ) from t1; |

## SQL试题六

有用户每日游戏时长数据如下：

|  |  |  |
| --- | --- | --- |
| **uid** | **dt** | **duration** |
| uid1 | 2021-02-20 | 1 |
| uid1 | 2021-02-21 | 2 |
| uid1 | 2021-02-22 | 3 |
| uid2 | 2021-02-20 | 3 |
| uid2 | 2021-02-21 | 5 |
| uid2 | 2021-02-22 | 6 |
| uid3 | 2021-02-20 | 7 |
| uid3 | 2021-02-21 | 8 |
| uid3 | 2021-02-22 | 9 |
| uid4 | 2021-02-23 | 10 |

要求使用SparkSQL统计以下信息:

1. 统计每个用户每天游戏累计时长。(要求同一用户每天游戏时长累加之前所有天的游戏时长）
2. 统计每个用户每天游戏时长累加前一天游戏时长的累计时长。
3. 统计每个用户每天游戏时长累加后一天游戏时长的累计时长。

数据附件：

|  |
| --- |
| create table t2(uid string,dt string,**duration** int) row format delimited fields terminated by '\t';  select uid,dt,duration,sum(duration) over(partition by uid order by dt rows between 1 preceding and current row) as rt from t2;  select uid,dt,duration,sum(duration) over(partition by uid order by dt rows between current row and 1 following) as rt from t2; |

## SQL试题七

公司某设备监控数据如下:(device\_id:设备ID,state:设备状态，dt：监控时间)

|  |  |  |
| --- | --- | --- |
| **device\_id** | **state** | **dt** |
| d1 | 1 | 20210101 |
| d1 | 1 | 20210102 |
| d1 | 1 | 20210103 |
| d1 | 2 | 20210104 |
| d1 | 2 | 20210105 |
| d1 | 1 | 20210106 |
| d2 | 2 | 20210107 |
| d2 | 2 | 20210108 |
| d2 | 1 | 20210109 |
| d2 | 1 | 20210110 |
| d2 | 2 | 20210111 |
| d3 | 1 | 20210112 |
| d3 | 1 | 20210113 |
| d3 | 2 | 20210114 |
| d3 | 3 | 20210115 |
| d3 | 4 | 20210116 |

统计目标：统计出每个设备状态出现变化时前一条数据，统计结果如上图标颜色数据。

数据文件附加：

|  |
| --- |
| create table t3 (device\_id string,state int,dt string) row format delimited fields terminated by |

## SQL试题八

有如下数据，用户注册信息表和用户登录信息表：

用户注册信息表 regist\_infos：

|  |  |  |
| --- | --- | --- |
| **uid** | **regist\_dt** | **regist\_os** |
| u1 | 20210301 | android |
| u2 | 20210301 | android |
| u3 | 20210301 | iphone |
| u4 | 20210302 | android |
| u5 | 20210302 | iphone |
| u6 | 20210303 | android |
| u7 | 20210303 | android |
| u8 | 20210304 | iphone |
| u9 | 20210304 | iphone |

用户登录信息表 login\_infos：

|  |  |
| --- | --- |
| **uid** | **login\_dt** |
| u1 | 20210301  android |
| u1 | 20210301 |
| u2 | 20210301 |
| ... | ... |

SparkSQL统计注册日后1日、2日、3日、4日、5日、6日、7日用户留存数。期望得到的结果(此结果与上面结果无关)：

|  |  |  |
| --- | --- | --- |
| **regist\_dt** | **days** | **cnt** |
| 20210301 | 1  android | 2 |
| 20210301 | 2 | 2 |
| 20210302 | 5 | 3 |
| ... | ... | ... |

附件数据：

|  |
| --- |
|  |

## SQL试题九

行列变换操作，数据如下：

|  |  |  |
| --- | --- | --- |
| **username** | **item** | **price** |
| zhangsan | A | 1 |
| zhangsan | B | 2 |
| zhangsan | C | 3 |
| zhangsan | D | 4 |
| lisi | A | 5 |
| lisi | B | 6 |
| lisi | C | 7 |
| wangwu | A | 8 |

要求获取以下结果，同时再将数据转换回以上表格样式：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **username** | **A** | **B** | **C** | **D** |
| zhangsan | 1 | 2 | 3 | 4 |
| lisi | 5 | 6 | 7 | 0 |
| wangwu | 8 | 0 | 0 | 0 |

数据附件：

|  |
| --- |
|  |

## SQL试题十

行列变换操作，数据如下：

|  |  |  |
| --- | --- | --- |
| **username** | **item** | **price** |
| zhangsan | A | 1 |
| zhangsan | B | 2 |
| zhangsan | C | 3 |
| zhangsan | D | 4 |
| lisi | A | 5 |
| lisi | B | 6 |
| lisi | C | 7 |
| wangwu | A | 8 |

要求获取以下结果,同时将结果再次转换成以上表格（Price除外）：

|  |  |  |
| --- | --- | --- |
| **username** | **item**  **B** | **price**  **D** |
| zhangsan | A,B,C,D  2 | 10  4 |
| lisi | A,B,C  6 | 18  0 |
| wangwu | A  0 | 8  0 |

数据附件：

|  |
| --- |
|  |

## SQL试题十一

读取 vpnlog 日志文件，其中userName为用户名，ts为记录时间，当type为login时，为登入时间，type为logout为登出时间  
问题：  
1）这一天，每个小时在线用户数  
2）计算这一天，各个用户的在线总时长，在线次数，最大在线时长  
（如果用户一天开始是登出记录，则认为他零点登入，如果一天结束时登入日志，则认为他24点登出）  
要求使用SparkSQL实现，并给出计算逻辑说明

数据文件附件如下：

|  |
| --- |
|  |

## SQL试题十二

|  |  |  |  |
| --- | --- | --- | --- |
| app | channel | province | userid |
| 消消乐 | ios | 北京 | abc |
| 王者荣耀 | Android | 上海 | cde |

需求:求app,channel,province任意组合下的用户数，用一个sql实现，上面一条记录会产生多条记录。

数据附件：

|  |
| --- |
| create table test (app string,channel string,province string,userid string) row format delimited fields terminated by '\t'; |
| 答案：  select app,channel ,province ,count(userid) as cnt from test1 group by app,channel,province with cube;  select app,channel ,province ,count(userid) as cnt from test1 group by app,channel,province with roll up;  select app,channel ,province ,count(userid) as cnt from test1 group by app,channel,province grouping sets ((app,channel),(channel,province)  ); |

## SQL试题十三

统计连续5天登录的用户，login登录表：

|  |  |
| --- | --- |
| dt | userid |
| 2020-09-01 | a |
| 2020-09-01 | b |
| 2020-09-02 | a |
| 2020-09-03 | a |
| 2020-09-03 | b |
| 2020-09-04 | a |
| 2020-09-04 | b |
| 2020-09-05 | a |
| 2020-09-06 | a |
| 2020-09-07 | a |
| 2020-09-07 | b |
| 2020-09-08 | a |
| 2020-09-09 | a |
| 2020-09-10 | a |
| 2020-09-11 | a |
| 2020-09-11 | b |
| 2020-09-23 | b |
| 2020-09-24 | b |
| 2020-09-26 | b |
| 2020-09-27 | b |
| 2020-09-29 | b |
| 2020-09-30 | b |
| 2020-09-12 | b |
| 2020-09-17 | b |

数据附件：

|  |
| --- |
| create table login\_info(dt string,userid string) row format delimited fields terminated by '\t'  答案：  select distinct userid from (select dt,userid,lead(dt,4,-1) over (partition by userid order by dt) as new\_dt from login\_info) t1 where t1.new\_dt != "-1" and datediff(new\_dt,dt) = 4; |

## SQL试题十四

login登录表：

|  |  |
| --- | --- |
| dt | userid |
| 2020-09-01 | a1 |
| 2020-09-01 | a2 |
| 2020-09-02 | a2 |
| 2020-09-02 | a1 |
| 2020-09-03 | a1 |
| 2020-09-04 | a1 |
| 2020-09-05 | a1 |
| 2020-09-06 | a1 |
| 2020-09-07 | a1 |
| 2020-09-07 | a2 |
| 2020-09-08 | a1 |
| 2020-09-09 | a1 |
| ... | ... |

次日留存为：2020-09-01登录的用户在2020-09-02也登录。

需求：求2020年9月1号之后每日的次日留存数。

数据附件：

|  |
| --- |
| create table login (dt string,userid string) row format delimited fields terminated by '\t';  答案：  select new\_dt ,count(\*) as ct from (select dt ,userid, lag (dt,1,-1) over (partition by userid order by dt) as new\_dt from login ) t where  t.new\_dt != "-1" and datediff(dt,new\_dt) = 1 group by t.new\_dt; |

## SQL试题十五

Hive中有个一个info表，如下：

|  |  |
| --- | --- |
| date | result |
| 2020-05-09 | win |
| 2020-05-09 | win |
| 2020-05-09 | lose |
| 2020-05-09 | lose |
| 2020-05-10 | win |
| 2020-05-10 | lose |
| 2020-05-10 | lose |
| ... | ... |

如果要生成下列结果，该如何写sql?

|  |  |  |
| --- | --- | --- |
| data | win | lose |
| 2020-05-09 | 2 | 2 |
| 2020-05-10 | 1 | 2 |
| ... | ... | ... |

数据附件：

|  |
| --- |
| create table info (`date` string,result string) row format delimited fields terminated by '\t';  答案：  select `date` ,sum(case when result = "win" then cnt else 0 end) win,sum(case when result = "lose" then cnt else 0 end) lose from (select `  date` ,result ,count(\*) as cnt from info group by `date` ,result) t1 group by `date`; |

## SQL试题十六

表temp两个字段：

|  |  |
| --- | --- |
| 表temp | |
| user | profile |
| abc | key1:value,key2:value2 |
| def | key1:value,key2:value2,key3:value3,key4:value4 |
| xyz | Key1:value |

需要转换如下结构：

|  |  |  |
| --- | --- | --- |
| user | profile\_key | profile\_value |
| abc | key1 | value |
| abc | key2 | vlaue2 |
| def | key1 | value |
| def | key2 | value2 |
| def | key3 | value3 |
| def | key4 | value4 |
| xyz | key1 | value |

数据附件：

|  |
| --- |
| create table temp (usr string,profile string) row format delimited fields terminated by '\t';  答案：  select usr,k,v from temp lateral view explode(str\_to\_map(profile,",",":")) xx as k,v  select usr, split(str,":")[0] as key ,split(str,":")[1] as value from temp lateral view explode(split(profile,",")) xx as str |

## SQL试题十七

有如下用户登录明细表tb\_cuid\_1d，一个用户可能对应多条记录:

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 字段含义 | 字段类型 | 字段示例 |
| cuid | 用户的唯一标识  (不同用户cuid不同) | string | ed2s9w |
| os | 平台 | string | android |
| soft\_version | 版本 | string | 11.0.0.1 |
| every\_day | 日期 | string | 20190101 |
| ext | 扩展字段 | array | [{},{},{}] |

数据示例：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cuid | os | soft\_version | every\_day | ext |
| A1 | Android | 11.0.0.1 | 20200401 | [{“id”:10001,”type”:”show”,”from”:”home”,”source”:”his”},{“id”:1002,”type”:”click”,”from”:”swan”,”source”:”rcm”},{“id”:1003,”type”:”slide”,”from”:”tool”,”source”:”banner”},{“id”:1001,”type”:”del”,”from”:”wode”,”source”:”myswan”}] |

数据附件：

|  |
| --- |
| create table tb\_cuid\_1d (cuid string,os string,soft\_version string,every\_day string,ext string) row format delimited fields terminated by '\t'; |

1. 写出用户表 tb\_cuid\_1d的 20200401 的次日、次7日留存的具体HQL ：一条sql统计出以下指标 （4.1号uv，4.1号在4.2号的留存uv，4.1号在4.8号的留存uv）

|  |
| --- |
| select tt.every\_day,tt.diff,count(distinct tt.cuid) as cnt from (select t1.cuid,t1.os,t1.every\_day,datediff(t2.new\_tm,t1.new\_tm) as diff from (select cuid,os,every\_day,from\_unixtime(unix\_timestamp(every\_day,"yyyyMMdd"),"yyyy-MM-dd") as new\_tm from tb\_cuid\_1d) t1 join (select cuid,os,every\_day,from\_unixtime(unix\_timestamp(every\_day,"yyyyMMdd"),"yyyy-MM-dd") as new\_tm from tb\_cuid\_1d) t2 on t1.cuid = t2.cuid where t1.new\_tm <= t2.new\_tm and t1.every\_day = '20200401') tt group by tt.every\_day,tt.diff; |

1. 解析tb\_cuid\_1d表中ext中所有的"type"对应的值

|  |
| --- |
| select cuid,get\_json\_object(jstr,"$.id") as id ,get\_json\_object(jstr,"$.type") as tp from tb\_cuid\_1d lateral view explode(split(ext,"#")) xx as jstr; |

1. 统计tb\_cuid\_1d表中，20200401号不同平台 、 版本下的uv、 pv

|  |
| --- |
| select every\_day,os,soft\_version,count(cuid) as pv,count(distinct cuid) as uv from tb\_cuid\_1d where every\_day = "20200401" group by every\_day, os ,soft\_version |

1. 基于以上统计结果，如果查看当天总的uv，pv是否能直接加和，为什么？

pv可以 ，uv不可以，因为如果同一个用户使用不同的系统、不同的版本登录统计结果不对。

1. 一条sql统计当天不同平台、版本下的uv、pv ， 以及整体的uv， pv

|  |
| --- |
| select distinct every\_day,os,soft\_version,currentday\_pv,currentday\_uv,total\_pv,total\_uv from (select cuid,os,soft\_version,every\_day,count(c  uid) over (partition by every\_day,os,soft\_version) as currentday\_pv,size(collect\_set(cuid) over(partition by every\_day,os,soft\_version)) as currentday\_uv,count(cuid) over (partition by every\_day) as total\_pv,size(collect\_set(cuid) over (partition by every\_day)) as total\_uv from tb\_cuid\_1d) ttt ; |

## SQL试题十八

现在有以下一个数据表

|  |  |
| --- | --- |
| **字段名称** | **备注** |
| id | 唯一ID |
| date | 日期（分区字段） |
| reg\_time | 注册时间（时间戳） |
| leave\_time | 注销时间（时间戳） |

请用一个sql计算2020年1月-2020年2月期间每天注册用户次日留存率？

举个例子：

0101注册用户20人，0101注册用户在0102注销用户10人，次日留存率=10/20

0102注册用户30人，0102注册用户在0103注销用户15人，次日留存率=15/30

数据举例：

|  |
| --- |
| 1 20200101 1577885949 1577972349  2 20200101 1577885949 1578184749  3 20200101 1577885949 1577972349  4 20200101 1577885949 1578184749  5 20200101 1577885949 1577972349  6 20200101 1577885949 1578011949  7 20200102 1577925549 1578011949  8 20200102 1577925549 1578357549  9 20200102 1577925549 1578011949  10 20200102 1577925549 1578357549  11 20200101 1577885949 1578357549 |

## SQL试题十九

使用SQL选出下表中6个指标中至少4个指标大于50的城市。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| local | oc | sc | pc | hc | gc | ghc |
| 青岛 | 96 | 50 | 56 | 55 | 43 | 21 |
| 北京 | 74 | 16 | 96 | 29 | 54 | 4 |
| 南京 | 5 | 52 | 18 | 82 | 18 | 83 |

## SQL试题二十

目前爬虫拿到的数据形成两张表，一张是行业表，另一张是商品表，需要清洗出每个行业对应的月销售额（子行业的销售额需要加总到对应的父行业）。

行业表 category：（json格式数据）

|  |  |  |
| --- | --- | --- |
| 行业  id | 父行业  pid | 行业名称  cg\_name |
| 1 |  | 服装 |
| 2 | 1 | 女装 |
| 3 | 2 | 时尚女装 |
| 4 | 1 | 男装 |
| 5 | 4 | 商务男装 |

商品表item：（json格式数据）

|  |  |  |  |
| --- | --- | --- | --- |
| 商品  item\_id | 商品行业  item\_cg\_id | 销售日期  sale\_dt | 销售额  sale |
| 1 | 3 | 20200102 | 100 |
| 2 | 2 | 20200104 | 200 |
| 3 | 5 | 20200201 | 3000 |

请使用Spark进行数据处理，java,scala都可以。最终得到各行业的月销售表(注意行业有层级关系，子行业的数据需要汇总到父行业)

|  |  |  |  |
| --- | --- | --- | --- |
| 行业名称 | 行业id | 月份 | 销售额 |
| 女装 | 2 | 202001 | 300 |
| 服装 | 1 | 202001 | 300 |
| 时尚女装 | 3 | 202001 | 100 |
| 商务男装 | 5 | 202002 | 3000 |
| 男装 | 4 | 202002 | 3000 |
| 服装 | 1 | 202002 | 3000 |

数据附件：

|  |
| --- |
|  |