尚硅谷大数据技术之企业SQL面试题

(作者：尚硅谷大数据研发部)

版本：V1.0

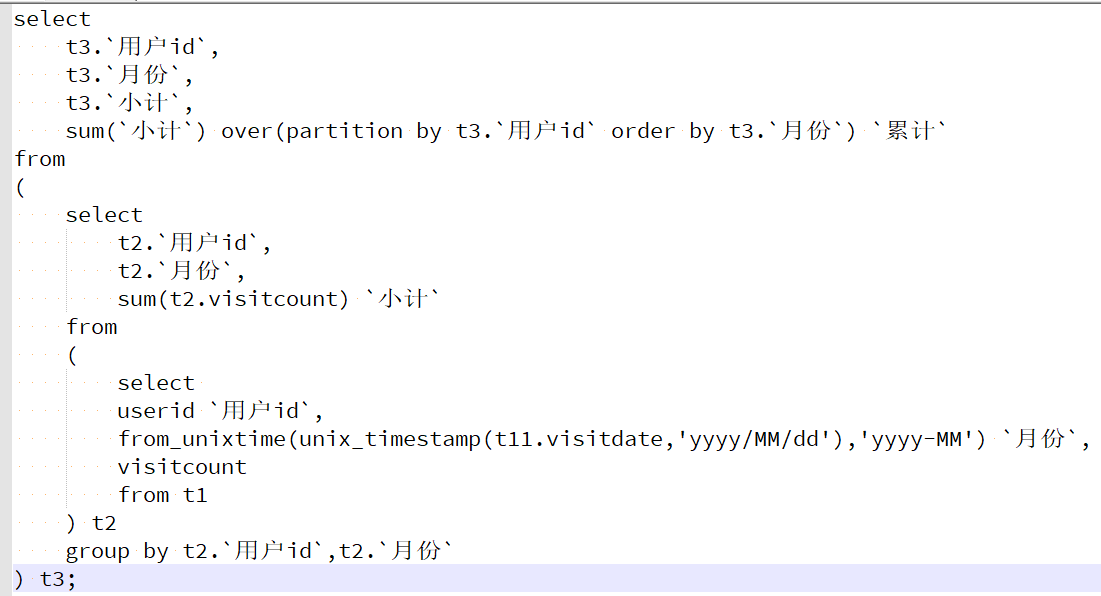
# 第1题

我们有如下的用户访问数据（t1）

|  |  |  |
| --- | --- | --- |
| userId | visitDate | visitCount |
| u01 | 2017/1/21 | 5 |
| u02 | 2017/1/23 | 6 |
| u03 | 2017/1/22 | 8 |
| u04 | 2017/1/20 | 3 |
| u01 | 2017/1/23 | 6 |
| u01 | 2017/2/21 | 8 |
| U02 | 2017/1/23 | 6 |
| U01 | 2017/2/22 | 4 |

要求使用SQL统计出每个用户的累积访问次数，如下表所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 用户id | 月份 | 小计 | 累积 |
| u01 | 2017-01 | 11 | 11 |
| u01 | 2017-02 | 12 | 23 |
| u02 | 2017-01 | 12 | 12 |
| u03 | 2017-01 | 8 | 8 |
| u04 | 2017-01 | 3 | 3 |



select

t3.`用户id`,

t3.`月份`,

t3.`小计`,

sum(`小计`) over(partition by t3.`用户id` order by t3.`月份`) `累计`

from

(

select

t2.`用户id`,

t2.`月份`,

sum(t2.visitcount) `小计`

from

(

select

userid `用户id`,

from\_unixtime(unix\_timestamp(t11.visitdate,'yyyy/MM/dd'),'yyyy-MM') `月份`,

visitcount

from t1

) t2

group by t2.`用户id`,t2.`月份`

) t3;

# 第2题 京东

有50W个京东店铺，每个顾客访客访问任何一个店铺的任何一个商品时都会产生一条访问日志，访问日志存储的表名为Visit，访客的用户id为user\_id，被访问的店铺名称为shop，请统计：

1. 每个店铺的UV（访客数）

PV：页面访问量，即PageView，用户每次对网站的访问均被记录，用户对同一页面的多次访问，访问量累计。 UV：独立访问用户数：即UniqueVisitor，访问网站的一台电脑客户端为一个访客。00:00-24:00内相同的客户端只被计算一次。

select

t1.shop,

count(t1.user\_id) `UV`

from

(

select

shop,

user\_id

from Visit

group by shop,user\_id

) t1

group by t1.shop;

1. 每个店铺访问次数top3的访客信息。输出店铺名称、访客id、访问次数

select

t2. shop,

t2. user\_id,

t2. visitcount,

t2.`rank`

from

(

select

t1. shop,

t1. user\_id,

t1. visitcount,

rank() over(partition by t1.shop order by t1.visitcount desc) `rank`

from

(

select

shop,

user\_id,

count(user\_id) visitcount

from Visit

group by shop,user\_id

) t1

) t2

where `rank`<=3;

# 第3题

已知一个表STG.ORDER，有如下字段:Date，Order\_id，User\_id，amount。请给出sql进行统计:数据样例:2017-01-01,10029028,1000003251,33.57。

1）给出 2017年每个月的订单数、用户数、总成交金额。

select

t1.`month`,

count(t1.User\_id) usercount,

sum(t1.ordercount) ordercount,

sum(t1.amount) amount

from

(

select

date\_format(Date,’yyyy-MM’) `month`,

User\_id,

count(Order\_id) ordercount,

sum(amount) amount

from STG.ORDER

group by date\_format(Date,’yyyy-MM’),User\_id

) t1

group by t1.`month`;

2）给出2017年11月的新客数(指在11月才有第一笔订单)

select

count(\*)

from

(

select

Date，Order\_id，User\_id，amount,

rank() over(partition by User\_id order by Date) `rank`

from STG.ORDER

) t1

where `rank`=1 and date\_format(t1.Date,’yyyy-MM’)=’2017-11’

group by user\_id;

# 第4题

有一个5000万的用户文件user\_tbl(user\_id，name，age)，一个2亿记录的用户看电影的记录文件movie\_tbl(user\_id，url)，根据年龄段观看电影的次数进行排序？

select

t1.age,

count(\*) count

from

(

select

m.user\_id,

m.url,

u.age

from movie\_tbl m

left join user\_tbl u on m.user\_id=u.user\_id

) t1

group by t1.age

order by count;

# 第5题

有日志如下，请写出代码求得所有用户和活跃用户的总数及平均年龄。（活跃用户指连续两天都有访问记录的用户）

日期 用户 年龄

11,test\_1,23

11,test\_2,19

11,test\_3,39

11,test\_1,23

11,test\_3,39

11,test\_1,23

12,test\_2,19

13,test\_1,23

with tmp\_uv as

(

select

count(t4.`用户`) uv\_count,

avg(t4.`年龄`) uv\_avg

from

(

select

t3.`用户`,

t3.`年龄`,

t3.diff,

count(\*) count

from

(

select

t2.`用户`,

t2.`年龄`,

t2.`日期`,

t2.`rank`,

t2.`日期`-t2.`rank` diff

from

(

select

t1.`用户`,

t1.`年龄`,

t1.`日期`,

rank() over(partition by t1. `用户` order by t1.`日期`) `rank`

from

(

select

t.`userid` `用户`,

t.`date` `日期`,

t.`age` `年龄`

from test\_log t

group by t.`userid`,t.`date`,t.`age`

) t1

) t2

) t3

group by t3.`用户`,t3.`年龄`,t3.diff

having count>=2

) t4

group by t4.`用户`,t4.`年龄`

),

tmp\_u as

(

select

count(l1.`用户`) u\_count,

avg(l1.`年龄`) u\_avg

from

(

select

l.`userid` `用户`,

l.`age` `年龄`

from

test\_log l

group by l.`userid`,l.`age`

) l1

)

select \* from tmp\_uv join tmp\_u;

# 第6题

请用sql写出所有用户中在今年10月份第一次购买商品的金额，表ordertable字段（购买用户：userid，金额：money，购买时间：paymenttime(格式：2017-10-01)，订单id：orderid）

select

t2.userid,

t2. money

from

(

select

t1. userid,

t1. money,

t1. paymenttime,

t1. orderid

rank() over(partition by userid order by paymenttime) `rank`

from

(

select

userid，

money，

paymenttime,

orderid

from ordertable

where date\_format(paymenttime,’yyyy-MM’)=’2017-10’

) t1

) t2

where t2.`rank`=1;

# 第7题

现有图书管理数据库的三个数据模型如下：

图书（数据表名：BOOK）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名称 | 字段描述 | 字段类型 |
| 1 | BOOK\_ID | 总编号 | 文本 |
| 2 | SORT | 分类号 | 文本 |
| 3 | BOOK\_NAME | 书名 | 文本 |
| 4 | WRITER | 作者 | 文本 |
| 5 | OUTPUT | 出版单位 | 文本 |
| 6 | PRICE | 单价 | 数值（保留小数点后2位） |

读者（数据表名：READER）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名称 | 字段描述 | 字段类型 |
| 1 | READER\_ID | 借书证号 | 文本 |
| 2 | COMPANY | 单位 | 文本 |
| 3 | NAME | 姓名 | 文本 |
| 4 | SEX | 性别 | 文本 |
| 5 | GRADE | 职称 | 文本 |
| 6 | ADDR | 地址 | 文本 |

借阅记录（数据表名：BORROW\_LOG）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 字段名称 | 字段描述 | 字段类型 |
| 1 | READER\_ID | 借书证号 | 文本 |
| 2 | BOOK\_ID | 总编号 | 文本 |
| 3 | BORROW\_ATE | 借书日期 | 日期 |

1. 创建图书管理库的图书、读者和借阅三个基本表的表结构。请写出建表语句。

create table Book(BOOK\_ID string, SORT string, BOOK\_NAME string, WRITER string, OUTPUT string, PRICE decimal(10,2)) row format delimited fields terminated by ‘\t’;

create table READER (READER\_ID string, COMPANY string, NAME string, SEX string, GRADE string, ADDR string) row format delimited fields terminated by ‘\t’;

create table BORROW LOG(READER\_ID string, BOOK\_D string, BORROW\_ATE string) row format delimited fields terminated by ‘\t’;

1. 找出姓李的读者姓名（NAME）和所在单位（COMPANY）。

select NAME, COMPANY from READER where NAME like ‘李%’;

1. 查找“高等教育出版社”的所有图书名称（BOOK\_NAME）及单价（PRICE），结果按单价降序排序。

select BOOK\_NAME, PRICE from BOOK where OUTPUT is ‘高等教育出版社’ order by PRICE desc;

1. 查找价格介于10元和20元之间的图书种类(SORT）出版单位（OUTPUT）和单价（PRICE），结果按出版单位（OUTPUT）和单价（PRICE）升序排序。

select SORT, OUTPUT, PRICE from BOOK where PRICE>=10 and PRICE<=20 order by OUTPUT, PRICE;

1. 查找所有借了书的读者的姓名（NAME）及所在单位（COMPANY）。

select NAME,COMPANY from BORROW LOG bl join READER r on bl. READER\_ID=r. READER\_ID group by NAME,COMPANY;

1. 求”科学出版社”图书的最高单价、最低单价、平均单价。

select MAX(PRICE),MIN(PRICE),AVG(PRICE) from BOOK;

1. 找出当前至少借阅了2本图书（大于等于2本）的读者姓名及其所在单位。

select

READER.NAME, READER.COMPANY

from

(

select READER\_ID from BORROW\_LOG group by READER\_ID having count(READER\_ID) >=2

) t1 join READER on t1. READER\_ID= READER. READER\_ID

1. 考虑到数据安全的需要，需定时将“借阅记录”中数据进行备份，请使用一条SQL语句，在备份用户bak下创建与“借阅记录”表结构完全一致的数据表BORROW\_LOG\_BAK.井且将“借阅记录”中现有数据全部复制到BORROW\_LOG\_ BAK中。

create table BORROW\_LOG\_BAK select \* from BORROW\_LOG;

1. 现在需要将原Oracle数据库中数据迁移至Hive仓库，请写出“图书”在Hive中的建表语句（Hive实现，提示：列分隔符|；数据表数据需要外部导入：分区分别以month＿part、day＿part 命名）

create external table (BOOK\_ID string,SORT string,BOOK\_NAME string,WRITER string,OUTPUT string,PRICE decimal(10,2))

row format delimited fields terminated by ‘|’

partitioned by (month\_part,day\_part);

1. Hive中有表A，现在需要将表A的月分区　201505　中　user＿id为20000的user＿dinner字段更新为bonc8920，其他用户user＿dinner字段数据不变，请列出更新的方法步骤。（Hive实现，提示：Hlive中无update语法，请通过其他办法进行数据更新）

create table tmp\_A as select \* from A where user\_id<>20000 and month\_part=201505;

insert into table tmp\_A partition(month\_part=’201505’) values(20000,其他字段,bonc8920);

insert overwrite table A partition(month\_part=’201505’) select \* from tmp\_A where month\_part=201505;

# 第8题

有一个线上服务器(server)访问日志格式如下（用sql答题）

时间 接口 ip地址

2016-11-09 11：22：05 /api/user/login 110.23.5.33

2016-11-09 11：23：10 /api/user/detail 57.3.2.16

.....

2016-11-09 23：59：40 /api/user/login 200.6.5.166

求11月9号下午14点（14-15点），访问api/user/login接口的top10的ip地址

select ip ,count(\*) count

from server

where (date\_format(`time`,'yyyy-MM-dd HH')='2016-11-09 14' or `time`='2016-11-09 15:00:00') and interface='/api/user/login'

group by ip

order by count desc

limit 2;

# 第9题

有一个充值日志表如下：

CREATE TABLE `credit log`

(

`dist\_id` int（11）DEFAULT NULL COMMENT '区组id',

`account` varchar（100）DEFAULT NULL COMMENT '账号',

`money` int(11) DEFAULT NULL COMMENT '充值金额',

`create\_time` datetime DEFAULT NULL COMMENT '订单时间'

)ENGINE=InnoDB DEFAUILT CHARSET-utf8

请写出SQL语句，查询充值日志表2015年7月9号每个区组下充值额最大的账号，要求结果：

区组id，账号，金额，充值时间

第一个思路是单组单笔充值金额最大的账号：

with tmp\_1 as(

select

`dist\_id`,

max(money) money

from `credit log`

where date\_format(`create\_time`,'yyyy-MM-dd')='2015-07-09'

group by `dist\_id`

)

select

cl.\*

from `credit log` cl join tmp\_1 on cl.money=tmp\_1.money and cl.`dist\_id`=tmp\_1.`dist\_id`;

用第10题的思路套用公式得出：

select

c1.\*,

(

select count(1)+1

from `credit log ` c2

where c2.`dist\_id`=c1.`dist\_id` and c2.`gold`>c1.`gold`

) rank

from `credit log `c1

having rank=1

order by rank;

#同样对于第二思路，可以先产生临时表将每个账户的总金额求出来.再套用上面的公式即可。

第二个思路是单组同一账户单日总充值金额最大的用户的信息：

with tmp\_1 as(

select

`dist\_id`,

`account`,

sum(money) money,

'2015-07-09'

from `credit log`

where date\_format(`create\_time`,'yyyy-MM-dd')='2015-07-09'

group by `dist\_id`,`account`

),

with tmp\_2 as(

select

`dist\_id`,

max(money) money

from tmp\_1

group by `dist\_id`

)

select

tmp\_1.\*

from tmp\_1 join tmp\_2 on tmp\_1.money=tmp\_2.money and tmp\_1.`dist\_id`=tmp\_2.`dist\_id`;

# 第10题

有一个账号表如下，请写出SQL语句，查询各自区组的money排名前十的账号（分组取前10）

CREATE TABIE `account`

(

`dist\_id` int（11）

DEFAULT NULL COMMENT '区组id'，

`account` varchar（100）DEFAULT NULL COMMENT '账号' ,

`gold` int（11）DEFAULT NULL COMMENT '金币'

PRIMARY KEY （`dist\_id`，`account\_id`），

）ENGINE=InnoDB DEFAULT CHARSET-utf8

重点：如何在mysql中实现类似oracle中rank() over()开窗函数的效果？

select

a1.\*,

(

select count(1)+1

from account a2

where a2.`dist\_id`=a1.`dist\_id` and a2.`gold`>a1.`gold`

) rank

from account a1

having rank<=10

order by rank;

# 第11题

1）有三张表分别为会员表（member）销售表（sale）退货表（regoods）

（1）会员表有字段memberid（会员id，主键）credits（积分）；

（2）销售表有字段memberid（会员id，外键）购买金额（MNAccount）；

（3）退货表中有字段memberid（会员id，外键）退货金额（RMNAccount）；

2）业务说明：

（1）销售表中的销售记录可以是会员购买，也可是非会员购买。（即销售表中的memberid可以为空）

（2）销售表中的一个会员可以有多条购买记录

（3）退货表中的退货记录可以是会员，也可是非会员4、一个会员可以有一条或多条退货记录

查询需求：分组查出销售表中所有会员购买金额，同时分组查出退货表中所有会员的退货金额，把会员id相同的购买金额-退款金额得到的结果更新到表会员表中对应会员的积分字段（credits）

with tmp\_t1 as

(

select

memberid,

sum(MNAccount) MNAccount

from sale

where memberid is not null

group by memberid

),

tmp\_t2 as(

select

memberid,

sum(RMNAccount) RMNAccount

from regoods

where memberid is not null

),

tmp\_t3 as(

select

tmp\_t1.memberid,

tmp\_t1.MNAccount,

tmp\_t2.RMNAccount,

tmp\_t1.MNAccount-tmp\_t2.RMNAccount credits

from

tmp\_t1 left join tmp\_t2 on tmp\_t1.memberid=tmp\_t2.memberid

)

update table member set member.credits=(member.credits+tmp\_t3.credits) where member.memberid=tmp\_t3.memberid;

#不确定update能不能这么用。

# 第12题 百度

现在有三个表student（学生表）、course(课程表)、score（成绩单），结构如下：

create table student

(

id bigint comment ‘学号’，

name string comment ‘姓名’,

age bigint comment ‘年龄’

);

create table course

(

cid string comment ‘课程号，001/002格式’,

cname string comment ‘课程名’

);

Create table score

(

Id bigint comment ‘学号’,

cid string comment ‘课程号’,

score bigint comment ‘成绩’

) partitioned by(event\_day string)

其中score中的id、cid，分别是student、course中对应的列请根据上面的表结构，回答下面的问题

1. 请将本地文件（/home/users/test/20190301.csv）文件，加载到分区表score的20190301分区中，并覆盖之前的数据

load data local inpath '/home/users/test/20190301.csv' overwrite into table score partition (event\_day='20190301');

1. 查出平均成绩大于60分的学生的姓名、年龄、平均成绩

select

student.\*,

t1.avg\_score

from

student join

(

select

sc.id,

avg(sc.score) avg\_score

from score sc

where every\_day= '20190301'

group by score.id

having avg\_score>60

) t1

on student.id=t1.id;

1. 查出没有‘001’课程成绩的学生的姓名、年龄

select

student.`name`,

student.`age`

from

student left join

(

select

sc.id

from

score sc

where sc.cid='001' and every\_day= '20190301'

) t1 on student.id=t1.id

where t1.id is null;

1. 查出有‘001’\’002’这两门课程下，成绩排名前3的学生的姓名、年龄

select

student.`name`,

student.`age`

from

student join

(

select

s1.id,

sum(s1.score) sum\_score

from

score s1 join

(

select

sc.id,

count(\*)

from

score sc

where event\_day='20190301' and (sc.cid='001' or sc.cid='002')

group by sc.id

having count(\*)=2

) t1 on s1.id=t1.id

group by s1.id

order by sum\_score

limit 3

) t2 on student.id=t2.id;

1. 创建新的表score\_20190317，并存入score表中20190317分区的数据

create table score\_20190317 as

select \* from score where event\_day='20190301';

1. 如果上面的score表中，uid存在数据倾斜，请进行优化，查出在20190101-20190317中，学生的姓名、年龄、课程、课程的平均成绩

set hive.map.aggr = true

set hive.groupby.mapaggr.checkinterval = 100000

set hive.groupby.skewindata = true

select

student.`name`,

student.`age`,

course.`cname`,

t1.avg\_score

from

(

select

uid,

cid,

avg(sc.score) avg\_score

from

score sc

where event\_day>='20190101' and event\_day<='20190317'

group by uid,cid

) t1 left join student on student.id=t1.uid

left join course on t1.cid=course.cid;

1. 描述一下union和union all的区别，以及在mysql和HQL中用法的不同之处？

union会对数据进行排序去重，union all不会排序去重。

HQL中要求union或union all操作时必须保证select 集合的结果相同个数的列，并且每个列的类型是一样的。

8）简单描述一下lateral view语法在HQL中的应用场景，并写一个HQL实例

一般用于udtf函数，可以实现一进多出的炸裂函数效果。

比如一个学生表为：

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 | 姓名 | 年龄 | 成绩（语文|数学|英语） |
| 001 | 张三 | 16 | 90，80，95 |

需要实现效果：

|  |  |
| --- | --- |
| 学号 | 成绩 |
| 001 | 90 |
| 001 | 80 |
| 001 | 95 |

create table student(

`id` string,

`name` string,

`age` int,

`scores` array<string>

)

row format delimited fields terminated by '\t'

collection items terminated by ',';

select

id,

score

from

student lateral view explode(scores) tmp\_score as score;