尚硅谷大数据技术之Hive SQL题库-高级

（作者：尚硅谷研究院）

版本：V1.0

**第1题 同时在线人数问题**

**1.1 题目需求**

现有各直播间的用户访问记录表（live\_events）如下，表中每行数据表达的信息为，一个用户何时进入了一个直播间，又在何时离开了该直播间。

|  |  |  |  |
| --- | --- | --- | --- |
| user\_id  (用户id) | live\_id  (直播间id) | in\_datetime  (进入直播间的时间) | out\_datetime  (离开直播间的时间) |
| 100 | 1 | 2021-12-1 19:30:00 | 2021-12-1 19:53:00 |
| 100 | 2 | 2021-12-1 21:01:00 | 2021-12-1 22:00:00 |
| 101 | 1 | 2021-12-1 19:05:00 | 2021-12-1 20:55:00 |

现要求统计各直播间最大同时在线人数，期望结果如下：

|  |  |
| --- | --- |
| live\_id | max\_user\_count |
| 1 | 4 |
| 2 | 3 |
| 3 | 2 |

**1.2 数据准备**

**1)建表语句**

drop table if exists live\_events;

create table if not exists live\_events

(

user\_id int comment '用户id',

live\_id int comment '直播id',

in\_datetime string comment '进入直播间时间',

out\_datetime string comment '离开直播间时间'

)

comment '直播间访问记录';

**2）数据装载**

INSERT overwrite table live\_events

VALUES (100, 1, '2021-12-01 19:00:00', '2021-12-01 19:28:00'),

(100, 1, '2021-12-01 19:30:00', '2021-12-01 19:53:00'),

(100, 2, '2021-12-01 21:01:00', '2021-12-01 22:00:00'),

(101, 1, '2021-12-01 19:05:00', '2021-12-01 20:55:00'),

(101, 2, '2021-12-01 21:05:00', '2021-12-01 21:58:00'),

(102, 1, '2021-12-01 19:10:00', '2021-12-01 19:25:00'),

(102, 2, '2021-12-01 19:55:00', '2021-12-01 21:00:00'),

(102, 3, '2021-12-01 21:05:00', '2021-12-01 22:05:00'),

(104, 1, '2021-12-01 19:00:00', '2021-12-01 20:59:00'),

(104, 2, '2021-12-01 21:57:00', '2021-12-01 22:56:00'),

(105, 2, '2021-12-01 19:10:00', '2021-12-01 19:18:00'),

(106, 3, '2021-12-01 19:01:00', '2021-12-01 21:10:00');

**1.3 代码实现**

select

live\_id,

max(user\_count) max\_user\_count

from

(

select

user\_id,

live\_id,

sum(user\_change) over(partition by live\_id order by event\_time) user\_count

from

(

select user\_id,

live\_id,

in\_datetime event\_time,

1 user\_change

from live\_events

union all

select user\_id,

live\_id,

out\_datetime,

-1

from live\_events

)t1

)t2

group by live\_id;

**第2题 会话划分问题**

**2.1 题目需求**

现有页面浏览记录表（page\_view\_events）如下，表中有每个用户的每次页面访问记录。

|  |  |  |
| --- | --- | --- |
| user\_id | page\_id | view\_timestamp |
| 100 | home | 1659950435 |
| 100 | good\_search | 1659950446 |
| 100 | good\_list | 1659950457 |
| 100 | home | 1659950541 |
| 100 | good\_detail | 1659950552 |
| 100 | cart | 1659950563 |
| 101 | home | 1659950435 |
| 101 | good\_search | 1659950446 |
| 101 | good\_list | 1659950457 |
| 101 | home | 1659950541 |
| 101 | good\_detail | 1659950552 |
| 101 | cart | 1659950563 |
| 102 | home | 1659950435 |
| 102 | good\_search | 1659950446 |
| 102 | good\_list | 1659950457 |
| 103 | home | 1659950541 |
| 103 | good\_detail | 1659950552 |
| 103 | cart | 1659950563 |

规定若同一用户的相邻两次访问记录时间间隔小于60s，则认为两次浏览记录属于同一会话。现有如下需求，为属于同一会话的访问记录增加一个相同的会话id字段，期望结果如下：

|  |  |  |  |
| --- | --- | --- | --- |
| user\_id | page\_id | view\_timestamp | session\_id |
| 100 | home | 1659950435 | 100-1 |
| 100 | good\_search | 1659950446 | 100-1 |
| 100 | good\_list | 1659950457 | 100-1 |
| 100 | home | 1659950541 | 100-2 |
| 100 | good\_detail | 1659950552 | 100-2 |
| 100 | cart | 1659950563 | 100-2 |
| 101 | home | 1659950435 | 101-1 |
| 101 | good\_search | 1659950446 | 101-1 |
| 101 | good\_list | 1659950457 | 101-1 |
| 101 | home | 1659950541 | 101-2 |
| 101 | good\_detail | 1659950552 | 101-2 |
| 101 | cart | 1659950563 | 101-2 |
| 102 | home | 1659950435 | 102-1 |
| 102 | good\_search | 1659950446 | 102-1 |
| 102 | good\_list | 1659950457 | 102-1 |
| 103 | home | 1659950541 | 103-1 |
| 103 | good\_detail | 1659950552 | 103-1 |

**2.2 数据准备**

**1)建表语句**

drop table if exists page\_view\_events;  
create table if not exists page\_view\_events  
(  
 user\_id int comment '用户id',  
 page\_id string comment '页面id',  
 view\_timestamp bigint comment '访问时间戳'  
)  
 comment '页面访问记录';

**2）数据装载**

insert overwrite table page\_view\_events  
values (100, 'home', 1659950435),  
 (100, 'good\_search', 1659950446),  
 (100, 'good\_list', 1659950457),  
 (100, 'home', 1659950541),  
 (100, 'good\_detail', 1659950552),  
 (100, 'cart', 1659950563),  
 (101, 'home', 1659950435),  
 (101, 'good\_search', 1659950446),  
 (101, 'good\_list', 1659950457),  
 (101, 'home', 1659950541),  
 (101, 'good\_detail', 1659950552),  
 (101, 'cart', 1659950563),  
 (102, 'home', 1659950435),  
 (102, 'good\_search', 1659950446),  
 (102, 'good\_list', 1659950457),  
 (103, 'home', 1659950541),  
 (103, 'good\_detail', 1659950552),  
 (103, 'cart', 1659950563);

**2.3 代码实现**

select user\_id,  
 page\_id,  
 view\_timestamp,  
 concat(user\_id, '-', sum(session\_start\_point) over (partition by user\_id order by view\_timestamp)) session\_id  
from (  
 select user\_id,  
 page\_id,  
 view\_timestamp,  
 if(view\_timestamp - lagts >= 60, 1, 0) session\_start\_point  
 from (  
 select user\_id,  
 page\_id,  
 view\_timestamp,  
 lag(view\_timestamp, 1, 0) over (partition by user\_id order by view\_timestamp) lagts  
 from page\_view\_events  
 ) t1  
 ) t2;

**第3题 间断连续登录用户问题**

**3.1 题目需求**

现有各用户的登录记录表（login\_events）如下，表中每行数据表达的信息是一个用户何时登录了平台。

|  |  |
| --- | --- |
| user\_id | login\_datetime |
| 100 | 2021-12-01 19:00:00 |
| 100 | 2021-12-01 19:30:00 |
| 100 | 2021-12-02 21:01:00 |

现要求统计各用户最长的连续登录天数，间断一天也算作连续，例如：一个用户在1,3,5,6登录，则视为连续6天登录。期望结果如下：

|  |  |
| --- | --- |
| user\_id | max\_day\_count |
| 100 | 3 |
| 101 | 6 |
| 102 | 3 |
| 104 | 3 |
| 105 | 1 |

**3.2 数据准备**

**1)建表语句**

drop table if exists login\_events;  
create table if not exists login\_events  
(  
 user\_id int comment '用户id',  
 login\_datetime string comment '登录时间'  
)  
 comment '直播间访问记录';

**2）数据装载**

INSERT overwrite table login\_events  
VALUES (100, '2021-12-01 19:00:00'),  
 (100, '2021-12-01 19:30:00'),  
 (100, '2021-12-02 21:01:00'),  
 (100, '2021-12-03 11:01:00'),  
 (101, '2021-12-01 19:05:00'),  
 (101, '2021-12-01 21:05:00'),  
 (101, '2021-12-03 21:05:00'),  
 (101, '2021-12-05 15:05:00'),  
 (101, '2021-12-06 19:05:00'),  
 (102, '2021-12-01 19:55:00'),  
 (102, '2021-12-01 21:05:00'),  
 (102, '2021-12-02 21:57:00'),  
 (102, '2021-12-03 19:10:00'),  
 (104, '2021-12-04 21:57:00'),  
 (104, '2021-12-02 22:57:00'),  
 (105, '2021-12-01 10:01:00');

**3.3 代码实现**

select

user\_id,

max(recent\_days) max\_recent\_days --求出每个用户最大的连续天数

from

(

select

user\_id,

user\_flag,

datediff(max(login\_date),min(login\_date)) + 1 recent\_days --按照分组求每个用户每次连续的天数(记得加1)

from

(

select

user\_id,

login\_date,

lag1\_date,

concat(user\_id,'\_',flag) user\_flag --拼接用户和标签分组

from

(

select

user\_id,

login\_date,

lag1\_date,

sum(if(datediff(login\_date,lag1\_date)>2,1,0)) over(partition by user\_id order by login\_date) flag --获取大于2的标签

from

(

select

user\_id,

login\_date,

lag(login\_date,1,'1970-01-01') over(partition by user\_id order by login\_date) lag1\_date --获取上一次登录日期

from

(

select

user\_id,

date\_format(login\_datetime,'yyyy-MM-dd') login\_date

from login\_events

group by user\_id,date\_format(login\_datetime,'yyyy-MM-dd') --按照用户和日期去重

)t1

)t2

)t3

)t4

group by user\_id,user\_flag

)t5

group by user\_id;

**第4题 日期交叉问题**

**4.1 题目需求**

现有各品牌优惠周期表（promotion\_info）如下，其记录了每个品牌的每个优惠活动的周期，其中同一品牌的不同优惠活动的周期可能会有交叉。

|  |  |  |  |
| --- | --- | --- | --- |
| promotion\_id | brand | start\_date | end\_date |
| 1 | oppo | 2021-06-05 | 2021-06-09 |
| 2 | oppo | 2021-06-11 | 2021-06-21 |
| 3 | vivo | 2021-06-05 | 2021-06-15 |

现要求统计每个品牌的优惠总天数，若某个品牌在同一天有多个优惠活动，则只按一天计算。期望结果如下：

|  |  |
| --- | --- |
| brand | promotion\_day\_count |
| vivo | 17 |
| oppo | 16 |
| redmi | 22 |
| huawei | 22 |

**4.2 数据准备**

**1)建表语句**

drop table if exists promotion\_info;

create table promotion\_info

(

promotion\_id string comment '优惠活动id',

brand string comment '优惠品牌',

start\_date string comment '优惠活动开始日期',

end\_date string comment '优惠活动结束日期'

) comment '各品牌活动周期表';

**2）数据装载**

insert overwrite table promotion\_info  
values (1, 'oppo', '2021-06-05', '2021-06-09'),  
 (2, 'oppo', '2021-06-11', '2021-06-21'),  
 (3, 'vivo', '2021-06-05', '2021-06-15'),  
 (4, 'vivo', '2021-06-09', '2021-06-21'),  
 (5, 'redmi', '2021-06-05', '2021-06-21'),  
 (6, 'redmi', '2021-06-09', '2021-06-15'),  
 (7, 'redmi', '2021-06-17', '2021-06-26'),  
 (8, 'huawei', '2021-06-05', '2021-06-26'),  
 (9, 'huawei', '2021-06-09', '2021-06-15'),  
 (10, 'huawei', '2021-06-17', '2021-06-21');

**4.3 代码实现**

select  
 brand,  
 sum(datediff(end\_date,start\_date)+1) promotion\_day\_count  
from  
(  
 select  
 brand,  
 max\_end\_date,  
 if(max\_end\_date is null or start\_date>max\_end\_date,start\_date,date\_add(max\_end\_date,1)) start\_date,  
 end\_date  
 from  
 (  
 select  
 brand,  
 start\_date,  
 end\_date,  
 max(end\_date) over(partition by brand order by start\_date rows between unbounded preceding and 1 preceding) max\_end\_date  
 from promotion\_info  
 )t1  
)t2  
where end\_date>start\_date  
group by brand;