题目1

用SQL分解留存计算，每步请用视图建立（90分）

1.0 建立用户活跃日期表(5分)

```
CREATE TABLE temp_user_act(
  user_id int,
  dates date)
```

1.1 用用户活跃日期表与用户活跃日期表做用户ID的左连接，保留两表的用户id与两表的日期（20分）

```
CREATE VIEW table_join
AS SELECT t1.user_id,
  t1.dates,
  t2.user_id user_id_2,
  t2.dates dates_2
FROM temp_user_act t1
LEFT JOIN temp_user_act t2 ON t1.user_id = t2.user_id
```

注意两个表相同，因此在连接后，要给和t1表格相同的列重新命名。该视图的如下所示：

user_id	dates	user_id_2	dates_2
98047837	2019-12-06	98047837	2019-12-06
97726136	2019-12-09	97726136	2019-12-09
98607707	2019-12-18	98607707	2019-12-18
98662432	2019-12-06	98662432	2019-12-06
98145908	2019-12-16	98145908	2019-12-16
93784494	2019-12-03	93784494	2019-12-03
94832743	2019-12-13	94832743	2019-12-13
95290487	2019-11-27	95290487	2019-11-27
96610296	2019-12-11	96610296	2019-12-11
100684618	2019-12-05	100684618	2019-12-05
100684618	2019-12-05	100684618	2019-11-26
100684618	2019-12-05	100684618	2019-12-11
100684618	2019-12-05	100684618	2019-12-12
100684618	2019-12-05	100684618	2019-12-06
100684618	2019-12-05	100684618	2019-12-14
100684618	2019-12-05	100684618	2019-12-10
100684618	2019-12-05	100684618	2019-12-16
100684618	2019-12-05	100684618	2019-11-23
100684618	2019-12-05	100684618	2019-12-03
100684618	2019-12-05	100684618	2019-12-15
100684618	2019-12-05	100684618	2019-11-22
100684618	2019-12-05	100684618	2019-12-18
100684618	2019-12-05	100684618	2019-11-24

图1-1 用户活跃日期表连接图（部分）

1.2 筛选出右表日期大于等于左表日期的内容（20分）

```
CREATE VIEW table_interval
AS SELECT *
FROM table_join
WHERE DATEDIFF(dates_2,dates) >= 0
```

dates	dates_2	user_id	user_id_2	diff
2019-12-06	2019-12-06	98047837	98047837	0
2019-12-09	2019-12-09	97726136	97726136	0
2019-12-18	2019-12-18	98607707	98607707	0
2019-12-06	2019-12-06	98662432	98662432	0
2019-12-16	2019-12-16	98145908	98145908	0
2019-12-03	2019-12-03	93784494	93784494	0
2019-12-13	2019-12-13	94832743	94832743	0
2019-11-27	2019-11-27	95290487	95290487	0
2019-12-11	2019-12-11	96610296	96610296	0
2019-12-05	2019-12-05	100684618	100684618	0
2019-11-26	2019-12-05	100684618	100684618	9
2019-11-23	2019-12-05	100684618	100684618	12
2019-12-03	2019-12-05	100684618	100684618	2
2019-11-22	2019-12-05	100684618	100684618	13
2019-11-24	2019-12-05	100684618	100684618	11

图1-2 右表日期大于左表的内容(部分)

1.3 计算以左表日期为首日的首日用户数，第二日用户数，第三日用户数，第四日用户数，第八日用户数；（20分）

```
CREATE VIEW table_amount
AS SELECT dates,
      COUNT(DISTINCT if(diff=0,user_id,null)) user_day1,
      COUNT(DISTINCT if(diff=1,user_id,null)) user_day2,
      COUNT(DISTINCT if(diff=2,user_id,null)) user_day3,
      COUNT(DISTINCT if(diff=3,user_id,null)) user_day4,
      COUNT(DISTINCT if(diff=7,user_id,null)) user_day8
FROM table_interval
GROUP BY dates
```

dates	user_day1	user_day2	user_day3	user_day4	user_day8
2019-11-18	226	147	146	136	136
2019-11-19	222	154	141	137	127
2019-11-20	231	146	137	143	132
2019-11-21	232	144	146	151	145
2019-11-22	226	157	145	148	143
2019-11-23	241	161	160	141	153
2019-11-24	236	163	151	157	154
2019-11-25	243	157	154	161	150
2019-11-26	221	150	149	144	125
2019-11-27	232	164	154	152	144
2019-11-28	240	163	152	154	138
2019-11-29	243	169	164	155	163
2019-11-30	244	155	166	158	158
2019-12-01	245	162	148	162	155
2019-12-02	241	162	161	162	156
2019-12-03	235	164	158	158	156
2019-12-04	247	168	169	163	165
2019-12-05	242	165	155	159	196
2019-12-06	250	163	161	161	154
2019-12-07	241	167	165	168	150
2019-12-08	247	171	173	181	153
2019-12-09	251	173	177	202	149
2019-12-10	254	190	207	166	163
2019-12-11	272	224	174	166	150
2019-12-12	324	203	193	195	0
2019-12-13	245	155	162	146	0
2019-12-14	235	161	152	151	0
2019-12-15	243	168	158	140	0
2019-12-16	245	166	143	0	0
2019-12-17	238	154	0	0	0
2019-12-18	224	0	0	0	0

图1-3 首日及第2、3、4、8日用户数

1.4 利用上述数据计算出每日的用户数以及次日留存率，二日留存，三日留存率，7日留存率(率需要使用百分比表示结果)；（20分）

第N天留存率 = 在第N天仍然使用产品的新增用户数/新增的总用户数

```
CREATE VIEW stay_rate
AS SELECT dates,
      user_day1 total,
      CONCAT(ROUND(100*user_day2/user_day1,2),"%") day2_stay,
      CONCAT(ROUND(100*user_day3/user_day1,2),"%") day3_stay,
      CONCAT(ROUND(100*user_day4/user_day1,2),"%") day4_stay,
      CONCAT(ROUND(100*user_day8/user_day1,2),"%") day8_stay
FROM table_amount
GROUP BY dates
```

运行结果：

dates	total	day2_stay	day3_stay	day4_stay	day8_stay
2019-11-18	226	65.04%	64.60%	60.18%	60.18%
2019-11-19	222	69.37%	63.51%	61.71%	57.21%
2019-11-20	231	63.20%	59.31%	61.90%	57.14%
2019-11-21	232	62.07%	62.93%	65.09%	62.50%
2019-11-22	226	69.47%	64.16%	65.49%	63.27%
2019-11-23	241	66.80%	66.39%	58.51%	63.49%
2019-11-24	236	69.07%	63.98%	66.53%	65.25%
2019-11-25	243	64.61%	63.37%	66.26%	61.73%
2019-11-26	221	67.87%	67.42%	65.16%	56.56%
2019-11-27	232	70.69%	66.38%	65.52%	62.07%
2019-11-28	240	67.92%	63.33%	64.17%	57.50%
2019-11-29	243	69.55%	67.49%	63.79%	67.08%
2019-11-30	244	63.52%	68.03%	64.75%	64.75%
2019-12-01	245	66.12%	60.41%	66.12%	63.27%
2019-12-02	241	67.22%	66.80%	67.22%	64.73%
2019-12-03	235	69.79%	67.23%	67.23%	66.38%
2019-12-04	247	68.02%	68.42%	65.99%	66.80%
2019-12-05	242	68.18%	64.05%	65.70%	80.99%
2019-12-06	250	65.20%	64.40%	64.40%	61.60%
2019-12-07	241	69.29%	68.46%	69.71%	62.24%
2019-12-08	247	69.23%	70.04%	73.28%	61.94%
2019-12-09	251	68.92%	70.52%	80.48%	59.36%
2019-12-10	254	74.80%	81.50%	65.35%	64.17%
2019-12-11	272	82.35%	63.97%	61.03%	55.15%
2019-12-12	324	62.65%	59.57%	60.19%	0.00%
2019-12-13	245	63.27%	66.12%	59.59%	0.00%
2019-12-14	235	68.51%	64.68%	64.26%	0.00%
2019-12-15	243	69.14%	65.02%	57.61%	0.00%
2019-12-16	245	67.76%	58.37%	0.00%	0.00%
2019-12-17	238	64.71%	0.00%	0.00%	0.00%
2019-12-18	224	0.00%	0.00%	0.00%	0.00%

图1-4 次日、二日、三日、七日留存率

1.5*.求出每日的次留与次留的周环比；（5分）

```sql
CREATE VIEW week_stay
AS SELECT dates,
    day2_stay,
    LEAD(day2_stay,7,null) OVER(order by dates) week2_day2_stay
FROM stay_rate
GROUP BY dates
```

结果：

dates	day2_stay	week2_day2_stay
2019-11-18	65.04%	64.61%
2019-11-19	69.37%	67.87%
2019-11-20	63.20%	70.69%
2019-11-21	62.07%	67.92%
2019-11-22	69.47%	69.55%
2019-11-23	66.80%	63.52%
2019-11-24	69.07%	66.12%
2019-11-25	64.61%	67.22%
2019-11-26	67.87%	69.79%
2019-11-27	70.69%	68.02%
2019-11-28	67.92%	68.18%
2019-11-29	69.55%	65.20%
2019-11-30	63.52%	69.29%
2019-12-01	66.12%	69.23%
2019-12-02	67.22%	68.92%
2019-12-03	69.79%	74.80%
2019-12-04	68.02%	82.35%
2019-12-05	68.18%	62.65%
2019-12-06	65.20%	63.27%
2019-12-07	69.29%	68.51%
2019-12-08	69.23%	69.14%
2019-12-09	68.92%	67.76%
2019-12-10	74.80%	64.71%
2019-12-11	82.35%	0.00%
2019-12-12	62.65%	(NULL)
2019-12-13	63.27%	(NULL)
2019-12-14	68.51%	(NULL)
2019-12-15	69.14%	(NULL)
2019-12-16	67.76%	(NULL)
2019-12-17	64.71%	(NULL)
2019-12-18	0.00%	(NULL)

图1-5 每日次留及次留周环比

题目二

某短视频公司有作者发布视频统计表如下：（10分）

表名：temp_author_act

字段：

> 字段名 字段类型 字段说明
>
> dates date 发布日期
>
> author_id varchar(5) 作者id

2.1 请用SQL求出作者的最近三个月内的最大断更天数、平均断更天数和最大持续更新天数；（5分）

2.1.1 总代码及结果图

总代码如下,思路会在后面详细解释：

```
## 1 创建表格
CREATE TABLE temp_author_act(
  dates date,
  author_id varchar(5))
## 2导入数据
## 3用lean计算相邻两次更新的时间差值
CREATE VIEW update_3m
AS SELECT *,
    DATEDIFF(lead_dates,dates) diff
FROM
    (SELECT author_id,
        dates,
        LEAD(dates) OVER(PARTITION BY author_id ORDER BY dates) lead_dates
    FROM temp_author_act a
    WHERE DATEDIFF(NOW(),dates) <=90) a
## 4 计算近三个月最大断更天数与平均断更天数
SELECT author_id,
      MAX(diff)-1 最大断更天数,
      CEIL(AVG(diff)-1) 平均断更天数
FROM update_3m
GROUP BY author_id
##5 计算最大持续更新天数
CREATE VIEW update_group
AS SELECT *,
    DATEDIFF(dates,"2020-05-28")-ranks group_num
FROM
  (SELECT *,
        ROW_NUMBER() OVER (PARTITION BY author_id ORDER BY dates) ranks
    FROM update_3m
    WHERE diff=1
    ) b

CREATE VIEW continua_update
AS SELECT author_id,
```

```
      MAX(update_num) 最大持续更新天数
FROM
  (SELECT author_id,
          group_num,
          COUNT(*) update_num
    FROM update_group
    GROUP BY author_id,group_num) c
GROUP BY author_id;
```

最大断更天数、平均断更天数结果：

author_id	最大断更天数	平均断更天数
a001	4	1
b001	11	7
c001	7	1

图2-1 不同作者最大断更天数、平均断更天数

最大持续更新天数结果：

author_id	最大持续更新天数
a001	5
b001	1
c001	14

图2-2 不同作者最大持续更新天数

2.1.2 创建表格

```
CREATE TABLE temp_author_act(
  dates date,
  author_id varchar(5))
```

2.2.2 导入数据

> 右键表格导入向导中设置

2.2.3 建立视图计算近三月最大断更天数及平均断更天数

根据题目，关键词：近三个月、最大断更天数、平均断更天数，最大持续断更天数。第一个关键词近三个月，所以想到要用where进行数据筛选。最大断更天数、平均断更天数、最大持续断更天数，这三个关键词让人想到了窗口函数中的练习8、练习9（“查询支付时间间隔查过100天的用户数”、“查询出每年支付时间间隔最长的用户”），因此想到要用lean函数新建一列，两列相减就是最近两次更新天数的差值。因此，先建立一个视图，包含需要的四列：author_id,dates, lead_dates,diff（即前面两个天数的差值）。代码如下：

```
CREATE VIEW update_3m
AS SELECT *,
    DATEDIFF(lead_dates,dates) diff
FROM
    (SELECT author_id,
        dates,
        LEAD(dates) OVER(PARTITION BY author_id ORDER BY dates) lead_dates
    FROM temp_author_act a
    WHERE DATEDIFF(NOW(),dates) <=90) a
```

视图结果如下：

author_id	dates	lead_dates	diff
a001	2020-05-29	2020-06-02	4
a001	2020-06-02	2020-06-03	1
a001	2020-06-03	2020-06-04	1
a001	2020-06-04	2020-06-07	3
a001	2020-06-07	2020-06-08	1
a001	2020-06-08	2020-06-09	1
a001	2020-06-09	2020-06-12	3
a001	2020-06-12	2020-06-13	1
a001	2020-06-13	2020-06-14	1
a001	2020-06-14	2020-06-15	1
a001	2020-06-15	2020-06-18	3
a001	2020-06-18	2020-06-19	1
a001	2020-06-19	2020-06-20	1
a001	2020-06-20	2020-06-21	1
a001	2020-06-21	2020-06-26	5
a001	2020-06-26	2020-06-27	1
a001	2020-06-27	2020-06-28	1

图2-3 近三个月作者临近更新天数(diff这列) (部分数据)

根据上面这个视图，最大断更天数、平均断更天数其实很好计算。diff-1就是断更的天数。

> 例如，假如更新时间是06-07， 06-09，那么中间06-08没更新，断更一天。此时diff = 06-08 - 06-06 = 2天。所以断更天数 = diff-1

因此，最大断更天数就是 max(diff) -1, 平均断更天数 = avg（diff）-1。代码如下：

```
SELECT author_id,
      MAX(diff)-1 最大断更天数,
      CEIL(AVG(diff)-1) 平均断更天数
FROM update_3m
GROUP BY author_id
```

结果如下：

author_id	最大断更天数	平均断更天数
a001	4	1
b001	11	7
c001	7	1

图2-4 不同作者最大断更天数及平均断更天数

2.1.4 计算最大持续更新天数

接下里看“最大持续更新天数的计算”。在求这个之前，我们需要先求每个作者的持续更新天数，然后max(持续更新天数)就是我们要的结果。那么怎么求持续更新天数？ 既然涉及到天数的计算，我们肯定要用到count()来计算。首先想到用diff这列进行计算，我们只要把每个作者diff=1的数据筛选出来，然后分别计算行数就行。我们想要的一个结果为：

间隔段数	包含的记录数目
第一组(dates=06-02,06-03这两行)	2
第二组(dates=06-07,06-08这两行)	2
第三组(dates=06-12,06-13，06-14这三行)	3
……	……

首先，找出连续更新的天数，这些天数的共同点是diff=1, 所以我们应该先用WHERE 把diff=1的记录1全部筛选出来。 此时数据如下图所示：

author_id	dates	lead_dates	diff
a001	2020-06-02	2020-06-03	1
a001	2020-06-03	2020-06-04	1
a001	2020-06-07	2020-06-08	1
a001	2020-06-08	2020-06-09	1
a001	2020-06-12	2020-06-13	1
a001	2020-06-13	2020-06-14	1
a001	2020-06-14	2020-06-15	1
a001	2020-06-18	2020-06-19	1
a001	2020-06-19	2020-06-20	1
a001	2020-06-20	2020-06-21	1
a001	2020-06-26	2020-06-27	1
a001	2020-06-27	2020-06-28	1

图2-5

此时，我们遇到的问题变为：如何把不同连续区间的记录区分开来。 如果我们把每次连续更新的日期看作一个组，那么所有diff=1的数据可以被分为很多组，如下图所示。

author_id	dates	lead_dates	diff	
a001	2020-06-02	2020-06-03	1	1
a001	2020-06-03	2020-06-04		1
a001	2020-06-07	2020-06-08	2	1
a001	2020-06-08	2020-06-09		1
a001	2020-06-12	2020-06-13	3	1
a001	2020-06-13	2020-06-14		1
a001	2020-06-14	2020-06-15		1
a001	2020-06-18	2020-06-19	4	1
a001	2020-06-19	2020-06-20		1
a001	2020-06-20	2020-06-21		1
a001	2020-06-26	2020-06-27		1
a001	2020-06-27	2020-06-28		1

图2-6

可以看到，这些数据的特点是：对于同一个作者，在组内，dates是连续变化的，每增加一行，dates+1；但是组间dates是跳跃变化的。如果我们在边上加一个排序结果，从第一行开始向下排序，序号分别为1，2，3...那么，用dates减去排序，就会使得每组组内的值是稳定的，但是各组组间的值均不同（因为组间dates是跳跃的）。最后，我们根据每组的值进行count，就可以得出每组持续更新多少天。所以，此时的思路是：

> 1. 建立新的一列,名称为"rank"，该列放置连续排序的数值 （可以用窗口函数row_number实现)
> 2. 另外新建一列Group_number，放置(dates -rank)的值。注意dates是日期数据，我们需要进行转换。首先考虑到用day(dates) - rank。这样对于同一个月是可行的，但是对不同的月份就会有问题。比如6月30号如果更新了，07-01号也更新了，两者计算出来的排序不同。
>
>   因此，需要考虑到月份。由于每个月份的日期会从30、31之间变化，用month*30/31 这种算法会比较麻烦。可以随便指定三个月前的一个日子，让dates-这个日期。所以用datediff函数。
> 3. 计算每个作者的持续更新天数：根据Group_number进行分组,count每组有多少条记录，结果即为每组的持续更新天数
> 4. MAX(每组持续更新天数)即为该作者最大持续更新天数。

步骤一代码(筛选出diff=1的数据，并建立代码）：

```
SELECT *,
        ROW_NUMBER() OVER (PARTITION BY author_id ORDER BY dates) ranks
FROM update_3m
WHERE diff=1
```

步骤二(在步骤一的基础上计算出Group_number)

```
CREATE VIEW update_group
AS SELECT *,
    DATEDIFF(dates,"2020-05-28")-ranks group_num
FROM
  (SELECT *,
        ROW_NUMBER() OVER (PARTITION BY author_id ORDER BY dates) ranks
    FROM update_3m
    WHERE diff=1
    ) b
```

结果如图所示:

author_id	dates	lead_dates	diff	ranks	groupnum
a001	2020-06-02	2020-06-03	1	1	4
a001	2020-06-03	2020-06-04	1	2	4
a001	2020-06-07	2020-06-08	1	3	7
a001	2020-06-08	2020-06-09	1	4	7
a001	2020-06-12	2020-06-13	1	5	10
a001	2020-06-13	2020-06-14	1	6	10
a001	2020-06-14	2020-06-15	1	7	10
a001	2020-06-18	2020-06-19	1	8	13
a001	2020-06-19	2020-06-20	1	9	13
a001	2020-06-20	2020-06-21	1	10	13
a001	2020-06-26	2020-06-27	1	11	18

图2-7 不同作者持续更新天数的不同分组

步骤三：计算每个作者的持续更新天数

```
SELECT author_id,
          group_num,
          COUNT(*) update_num
    FROM update_group  #步骤二生成的表格
    GROUP BY author_id,group_num
```

步骤四: 计算出每个作者持续更新的天数

```
CREATE VIEW continua_update
AS SELECT author_id,
      MAX(update_num)
FROM
  (SELECT author_id,
          group_num,
          COUNT(*) update_num
    FROM update_group
    GROUP BY author_id,group_num) c
GROUP BY author_id;
```

结果如下图:

author_id	最大持续更新天数
a001	5
b001	1
c001	14

图2-8

2.2 运营人员需要对作者做电话访问，需要你用SQL求出每位作者在最大断更天数时对应的日期范围。用于访问该日期内的断更原因。（5分）

2.2.1 方法一

之前2.1 已经知道作者a、b、c最大断更天数分别为4、11、7。即他们最大的diff值分别为5、12、8。因此，只要用这些值在update_3m （包含diff数据的视图）中做筛选就行。

```
SELECT *
FROM update_3m
WHERE (author_id = 'a001' AND diff = 5)
OR (author_id = 'b001' AND diff = 12)
OR (author_id = 'c001' AND diff = 8)
```

这个方法的查询时间为0.001秒，比方法二(0.004秒)要快一些。

2.2.2 方法二

一个更简便的方法是用窗口函数降序排序，每组去排序为1的数据出来。具体代码如下:

```
SELECT *
FROM
  (SELECT *,
        DENSE_RANK() OVER (PARTITION BY author_id ORDER BY diff DESC) ranks
  FROM update_3m) rank_table
WHERE ranks = 1;
```

该方法的相比方法一的好处是不需要实现知道最大断更天数的值为多少，可以直接使用。结果如下图:

author_id	dates	lead_dates	diff	ranks
a001	2020-06-21	2020-06-26	5	1
a001	2020-07-05	2020-07-10	5	1
a001	2020-08-09	2020-08-14	5	1
b001	2020-06-11	2020-06-23	12	1
c001	2020-08-09	2020-08-17	8	1

图2-9