一场pandas与SQL的巅峰大战（五）

超哥要努力
公众号：超哥的杂货铺

已关注

4 人赞同了该文章

本文目录：

在之前的四篇系列文章中，我们对比了pandas和SQL在数据方面的多项操作。

具体来讲，第一篇文章一场pandas与SQL的巅峰大战涉及到**数据查看**，**去重计数**，**条件选择**，**合并连接**，**分组排序**等操作。

第二篇文章一场pandas与SQL的巅峰大战（二）涉及**字符串处理**，**窗口函数**，**行列转换**，**类型转换**等操作。

第三篇文章一场pandas与SQL的巅峰大战（三）围绕日期操作展开，主要讨论了**日期获取**，**日期转换**，**日期计算**等内容。

第四篇文章一场pandas与SQL的巅峰大战（四）学习了在MySQL，Hive SQL和pandas中用多种方式计算**日环比，周同比**的方法。

本篇文章一起来探讨如何在SQL和pandas中计算累计百分比。仍然分别在MySQL，Hive SQL和pandas中用多种方案来实现。

◆ ◆ ◆ ◆ ◆

数据准备

我们仍然使用前一篇的orderamt数据，数据导入方式可以参考之前的内容。需要分别在MySQL，Hive，pandas中进行数据导入，在此不作赘述。公众号后台回复“**对比五**”，可以获取本文全部代码和数据。数据的样例为：

id	dt	amt
1	2019-11-01	1,043
2	2019-11-02	1,119
3	2019-11-03	878
4	2019-11-04	1,197
5	2019-11-05	1,156
6	2019-11-06	1,142
7	2019-11-07	826
8	2019-11-08	948
9	2019-11-09	1,153
10	2019-11-10	826
11	2019-11-11	841
12	2019-11-12	1,017
13	2019-11-13	1,022
14	2019-11-14	945
15	2019-11-15	979
16	2019-11-16	925
17	2019-11-17	885
18	2019-11-18	1,041

知乎 @超哥要努力

我们的目标是，计算累计到当天的销售额占总销售额的比例。在实现时，首先分别计算出累计到当天的销售金额和总计的金额，然后就可以很方便的求出比例了。

MySQL计算累计百分比

1.不分组情况

最直观的思路是，对每一行的金额，都累加从第一行到当前行的金额。在MySQL中，可以考虑自连接的方式，但需要使用不等值连接。代码和结果如下：

```
select a.id, a.dt, a.orderamt,
sum(b.orderamt) as cum--对b表的金额进行求和
from t_orderamt a
join t_orderamt b
on a.dt >= b.dt--使用不等值连接
group by a.id, a.dt, a.orderamt
```

信息 结果1 概况 状态

id	dt	orderamt	cum
1	2019-11-01	1043	1043
2	2019-11-02	1119	2162
3	2019-11-03	878	3040
4	2019-11-04	1197	4237
5	2019-11-05	1156	5393
6	2019-11-06	1142	6535
7	2019-11-07	826	7361
8	2019-11-08	948	8309
9	2019-11-09	1153	9462
10	2019-11-10	826	10288
11	2019-11-11	841	11129
12	2019-11-12	1017	12146
13	2019-11-13	1022	13168
14	2019-11-14	945	14113
15	2019-11-15	979	15092

知乎 @超哥要努力

图中的cum列即是我们想要求的累加值。而所有销售金额的总计值，我们可以直接使用sum求出。

```
select sum(orderamt) as total
from t_orderamt
```

结合上面的两段SQL，就可以求得累计的百分比，注意连接条件我们使用了 1=1 这种恒成立的方式。代码和结果如下：

```
select c.id, c.orderamt, c.cum,
concat(round((c.cum / d.total) * 100, 2), '%') as cum_pct
from
(
select a.id, a.dt, a.orderamt, sum(b.orderamt) as cum
from t_orderamt a
```

```
join t_orderamt b
on a.dt >= b.dt
group by a.id, a.dt, a.orderamt
) c
left join
(
select sum(orderamt) as total
from t_orderamt
) d on 1 = 1
```

id	orderamt	cum	cum_pct
1	1043	1043	1.71%
2	1119	2162	3.54%
3	878	3040	4.98%
4	1197	4237	6.94%
5	1156	5393	8.84%
6	1142	6535	10.71%
7	826	7361	12.06%
8	948	8309	13.62%
9	1153	9462	15.51%
10	826	10288	16.86%
11	841	11129	18.24%
12	1017	12146	19.91%
13	1022	13168	21.58%
14	945	14113	23.13%
15	979	15092	24.73%

知乎@程序要努力

2.分组情况

需要思考：我们的原始数据是两个月的数据，目前我们的算法是把两个月的销售额累计到一起算的。但在实际中可能更多会关心每天的累计销售额分别占当月的百分比。如何能按照月份分组求每组的累计百分比呢?

首先仍然是求累计金额，但要分月累计。在上面的基础上加上月份相等条件即可，从结果中可以看到，在11月和12月cum列是分别累计的。

```
select substr(a.dt, 1, 7) as mon, a.dt, a.orderamt, sum(b.orderamt) as cum
from t_orderamt a
join t_orderamt b
on a.dt >= b.dt and
```

```
substr(a.dt, 1, 7) = substr(b.dt, 1, 7)--增加了这个条件
group by substr(a.dt, 1, 7), a.dt, a.orderamt
```

mon	dt	orderamt	cum
2019-11	2019-11-25	817	24867
2019-11	2019-11-26	1171	26038
2019-11	2019-11-27	1200	27238
2019-11	2019-11-28	917	28155
2019-11	2019-11-29	1194	29349
2019-11	2019-11-30	1196	30545
2019-12	2019-12-01	1182	1182
2019-12	2019-12-02	1186	2368
2019-12	2019-12-03	874	3242
2019-12	2019-12-04	904	4146
2019-12	2019-12-05	932	5078
2019-12	2019-12-06	820	5898
2019-12	2019-12-07	813	6711
2019-12	2019-12-08	1100	7811
2019-12	2019-12-09	[illegible]	[illegible]

求每月总计金额的代码比较简单：

```
select substr(a.dt, 1, 7) as mon, sum(orderamt) as total
from t_orderamt a
group by substr(a.dt, 1, 7)
```

同样的，我们把两段代码进行合并，就得到每月的累计百分比情况：

```
select c.mon, c.dt, c.orderamt, c.cum, d.total,
concat(round((c.cum / d.total) * 100, 2), '%') as cum_pct
from
(
select substr(a.dt, 1, 7) as mon, a.dt, a.orderamt, sum(b.orderamt) as cum
from t_orderamt a
join t_orderamt b
on a.dt >= b.dt and substr(a.dt, 1, 7) = substr(b.dt, 1, 7)
group by substr(a.dt, 1, 7), a.dt, a.orderamt
) c
left join
(
```

```
select substr(a.dt, 1, 7) as mon, sum(orderamt) as total
from t_orderamt a
group by substr(a.dt, 1, 7)
) d on c.mon = d.mon
```

mon	dt	orderamt	cum	total	cum_pct
2019-11	2019-11-25	817	24867	30545	81.41%
2019-11	2019-11-26	1171	26038	30545	85.24%
2019-11	2019-11-27	1200	27238	30545	89.17%
2019-11	2019-11-28	917	28155	30545	92.18%
2019-11	2019-11-29	1194	29349	30545	96.08%
2019-11	2019-11-30	1196	30545	30545	100.00%
2019-12	2019-12-01	1182	1182	30471	3.88%
2019-12	2019-12-02	1186	2368	30471	7.77%
2019-12	2019-12-03	874	3242	30471	10.64%
2019-12	2019-12-04	904	4146	30471	13.61%
2019-12	2019-12-05	932	5078	30471	16.67%
2019-12	2019-12-06	820	5898	30471	19.36%
2019-12	2019-12-07	813	6711	30471	22.02%
2019-12	2019-12-08	1100	7811	30471	25.63%
2019-12	2019-12-09	818	8629	30471	28.32%

知乎 @超哥要努力

Hive 计算累计百分比

1.不分组情况

Hive SQL中我们可以沿用MySQL中的思路，但需要注意，Hive 不支持在on中写不等号的连接条件，虽然可以采用where的方式改造一下，代码如下所示。但这并不是最优的方案。我们可以使用Hive中的窗口函数，很方便的计算累计值。

```
--where方法
select a.id, a.dt, a.orderamt,
sum(b.orderamt) as cum--对b表的金额进行求和
from t_orderamt a
join t_orderamt b
on 1=1
where a.dt >= b.dt--使用不等值连接
group by a.id, a.dt, a.orderamt
```

```
--窗口函数
select *, sum(orderamt) over(order by dt) as cum from t_orderamt;
```

两段代码的执行结果都如下图所示：

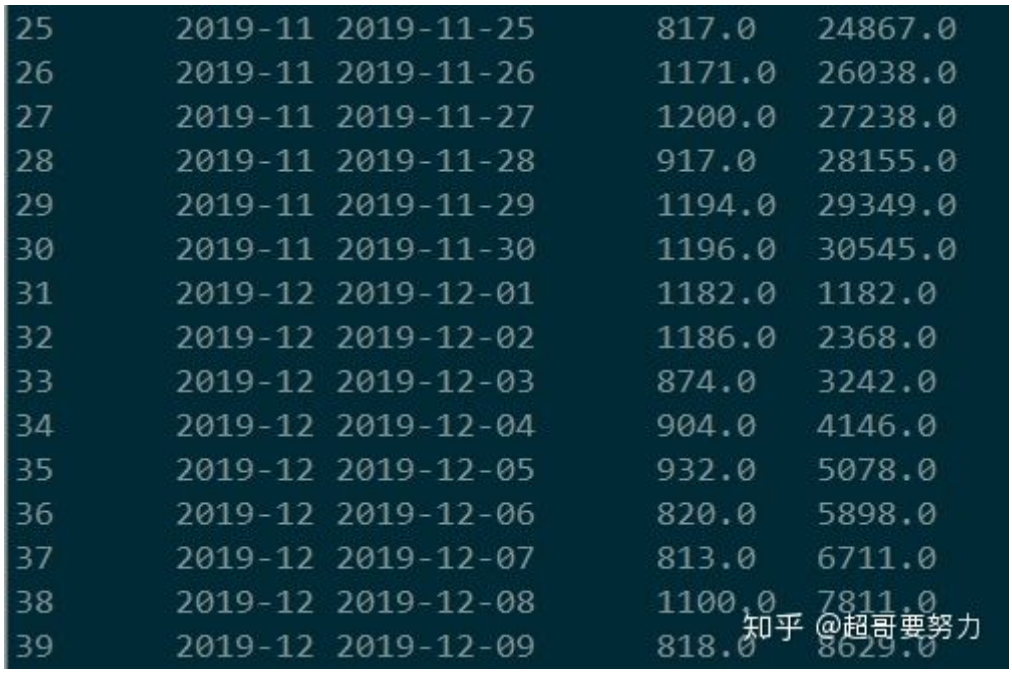
接下来我们重点看窗口函数的方式。在计算总计值的时候和前面MySQL的方式类似，累计百分比的计算也是需要把两部分代码结合在一起。

```
select c.id, c.dt, c.orderamt, c.cum,
concat(round((c.cum / d.total) * 100, 2), '%') as cum_pct
from
(
select *, sum(orderamt) over(order by dt) as cum
from t_orderamt
) c
left join
(
select sum(orderamt) as total
from t_orderamt
) d
on 1 = 1--在Hive中这个条件可以不写
```

2.分组情况

分组的情况，在窗口函数里是可以用 partition by 直接指定分组的，见如下代码

```
select id, substr(dt, 1, 7) as mon,
dt, orderamt,
sum(orderamt) over(partition by substr(dt, 1, 7) order by dt) as cum
from t_orderamt;
```



可以看到，同前面的分组情况一样，在11月和12月cum列是分别累计的。

接下来也很容易就写出分组计算累计百分比的代码，结果和上面也是一致的。

```
select c.mon, c.dt, c.orderamt, c.cum, d.total,
concat(round((c.cum / d.total) * 100, 2), '%') as cum_pct
from
(
select id, substr(dt, 1, 7) as mon, dt, orderamt, sum(orderamt) over(partition
from t_orderamt
) c
left join
(
select substr(dt, 1, 7) as mon, sum(orderamt) as total
from t_orderamt
group by substr(dt, 1, 7)
) d on c.mon = d.mon
```

```
2019-11 2019-11-25      817.0   24867.0 30545.0 81.41%
2019-11 2019-11-26      1171.0  26038.0 30545.0 85.24%
2019-11 2019-11-27      1200.0  27238.0 30545.0 89.17%
2019-11 2019-11-28      917.0   28155.0 30545.0 92.18%
2019-11 2019-11-29      1194.0  29349.0 30545.0 96.08%
2019-11 2019-11-30      1196.0  30545.0 30545.0 100.0%
2019-12 2019-12-01      1182.0  1182.0  30471.0 3.88%
2019-12 2019-12-02      1186.0  2368.0  30471.0 7.77%
2019-12 2019-12-03      874.0   3242.0  30471.0 10.64%
2019-12 2019-12-04      904.0   4146.0  30471.0 13.61%
2019-12 2019-12-05      932.0   5078.0  30471.0 16.67%
2019-12 2019-12-06      820.0   5898.0  30471.0 19.36%
2019-12 2019-12-07      813.0   6711.0  30471.0 22.02%
2019-12 2019-12-08      1100.0  7811.0  30471.0 25.63%
2019-12 2019-12-09      818.0   8629.0  30471.0 28.32%
```

知乎 @超哥要努力

pandas计算累计百分比

在pandas中，提供了专门的函数来计算累计值，分别是 `cumsum` 函数， `expanding` 函数， `rolling` 函数。我们一起来看一下使用三种函数计算分组和不分组累计百分比的方法。

1.不分组情况

cumsum函数

cumsum是pandas中专门用于计算累计和的函数。类似的函数还有cumprod计算累计积，cummax计算前n个值的最大值，cummin计算前n个值的最小值。

```
import pandas as pd
orderamt = pd.read_excel('orderamt.xlsx')
orderamt['cum_amt'] = orderamt['amt'].cumsum()
orderamt.head(15)
```

	id	dt	amt	cum_amt
0	1	2019-11-01	1043	1043
1	2	2019-11-02	1119	2162
2	3	2019-11-03	878	3040
3	4	2019-11-04	1197	4237
4	5	2019-11-05	1156	5393
5	6	2019-11-06	1142	6535
6	7	2019-11-07	826	7361
7	8	2019-11-08	948	8309
8	9	2019-11-09	1153	9462
9	10	2019-11-10	826	10288
10	11	2019-11-11	841	11129
11	12	2019-11-12	1017	12146
12	13	2019-11-13	1022	13168
13	14	2019-11-14	945	14113
14	15	2019-11-15	979	15092

知乎 @超能要努力

直接对amt列使用cumsum函数即可计算累计值，结果和用SQL计算得到的一致。

计算累计的百分比也很容易。

```
orderamt['cum_amt_pct'] = orderamt['cum_amt'] / orderamt['amt'].sum()
orderamt.head(15)
```

	id	dt	amt	cum_amt	cum_amt_pct
0	1	2019-11-01	1043	1043	0.017094
1	2	2019-11-02	1119	2162	0.035433
2	3	2019-11-03	878	3040	0.049823
3	4	2019-11-04	1197	4237	0.069441
4	5	2019-11-05	1156	5393	0.088387
5	6	2019-11-06	1142	6535	0.107103
6	7	2019-11-07	826	7361	0.120640
7	8	2019-11-08	948	8309	0.136177
8	9	2019-11-09	1153	9462	0.155074
9	10	2019-11-10	826	10288	0.168612
10	11	2019-11-11	841	11129	0.182395
11	12	2019-11-12	1017	12146	0.199063
12	13	2019-11-13	1022	13168	0.215812
13	14	2019-11-14	945	14113	0.231300
14	15	2019-11-15	979	15092	0.247344

知乎 @超哥努力

关于结果如何显示成百分比的形式，可以参考上一篇文章，此处略 。

expanding函数

pandas中的expanding函数是窗口函数的一种，它不固定窗口的大小，而是进行累计的计算。类似于cumsum()，但更强大。

```
orderamt = pd.read_excel('orderamt.xlsx')
orderamt['mon'] = orderamt['dt'].dt.strftime('%Y-%m')#得到字符串形式的月份
orderamt['cum_expand'] = orderamt.expanding(min_periods=1)['amt'].sum()
orderamt.head(15)
```

	id	dt	amt	mon	cum_expand
0	1	2019-11-01	1043	2019-11	1043.0
1	2	2019-11-02	1119	2019-11	2162.0
2	3	2019-11-03	878	2019-11	3040.0
3	4	2019-11-04	1197	2019-11	4237.0
4	5	2019-11-05	1156	2019-11	5393.0
5	6	2019-11-06	1142	2019-11	6535.0
6	7	2019-11-07	826	2019-11	7361.0
7	8	2019-11-08	948	2019-11	8309.0
8	9	2019-11-09	1153	2019-11	9462.0
9	10	2019-11-10	826	2019-11	10288.0
10	11	2019-11-11	841	2019-11	11129.0
11	12	2019-11-12	1017	2019-11	12146.0
12	13	2019-11-13	1022	2019-11	13168.0
13	14	2019-11-14	945	2019-11	14113.0
14	15	2019-11-15	979	2019-11	15092.0

知乎 @超凡的努力

参数 min_periods 表示最小的观测窗口，默认为1，可以设置为其他值，但如果窗口内记录数不足该值，则会显示NA。

有了累计值，计算累计的百分比，可以按照cumsum中的方法进行，此处省略。

rolling函数

rolling函数与expanding相比，主要是固定了窗口大小。当窗口超过dataframe的长度时，可以实现与expanding同样的效果。上面的代码使用rolling函数的方式可以改写如下，注意指定了window参数为len(orderamt)：

```
orderamt = pd.read_excel('orderamt.xlsx')
orderamt['mon'] = orderamt['dt'].dt.strftime('%Y-%m')#得到字符串形式的月份
orderamt['cum_roll'] = orderamt.rolling(window=len(orderamt), min_periods=1)['a
orderamt.head(15)
```

	id	dt	amt	mon	cum_roll
0	1	2019-11-01	1043	2019-11	1043.0
1	2	2019-11-02	1119	2019-11	2162.0
2	3	2019-11-03	878	2019-11	3040.0
3	4	2019-11-04	1197	2019-11	4237.0
4	5	2019-11-05	1156	2019-11	5393.0
5	6	2019-11-06	1142	2019-11	6535.0
6	7	2019-11-07	826	2019-11	7361.0
7	8	2019-11-08	948	2019-11	8309.0
8	9	2019-11-09	1153	2019-11	9462.0
9	10	2019-11-10	826	2019-11	10288.0
10	11	2019-11-11	841	2019-11	11129.0
11	12	2019-11-12	1017	2019-11	12146.0
12	13	2019-11-13	1022	2019-11	13168.0
13	14	2019-11-14	945	2019-11	14113.0
14	15	2019-11-15	979	2019-11	15092.0

知乎 @越努力越努力

此处同样省略计算累计百分比的代码。

2.分组情况

cumsum函数

```
#添加pandas显示设置，显示所有行
pd.set_option('display.max_rows', None)

orderamt = pd.read_excel('orderamt.xlsx')
orderamt['mon'] = orderamt['dt'].dt.strftime('%Y-%m')

#分组后对amt求累计和
orderamt['cum_mon'] = orderamt.groupby('mon')['amt'].cumsum()
orderamt
```

25	26	2019-11-26	1171	2019-11	26038
26	27	2019-11-27	1200	2019-11	27238
27	28	2019-11-28	917	2019-11	28155
28	29	2019-11-29	1194	2019-11	29349
29	30	2019-11-30	1196	2019-11	30545
30	31	2019-12-01	1182	2019-12	1182
31	32	2019-12-02	1186	2019-12	2368
32	33	2019-12-03	874	2019-12	3242
33	34	2019-12-04	904	2019-12	4146
34	35	2019-12-05	932	2019-12	5078
35	36	2019-12-06	820	2019-12	5898
36	37	2019-12-07	813	2019-12	6711
37	38	2019-12-08	1100	2019-12	7811
38	39	2019-12-09	818	2019-12	8629

知乎 @超哥要努力

接下来计算分组的总计值，这里用到了pandas中的transform函数，可以把分组后计算的总计值写入原dataframe。如果你不是很理解，可以参考下面这篇文章，讲的很清楚。

jianshu.com/p/509d7b970

```
orderamt['mon_total'] = orderamt.groupby('mon')["amt"].transform('sum')
orderamt['grp_cum_pct'] = orderamt['cum_mon'] / orderamt['mon_total']
orderamt
```

24	25	2019-11-25	817	2019-11	24867	30545	0.814110
25	26	2019-11-26	1171	2019-11	26038	30545	0.852447
26	27	2019-11-27	1200	2019-11	27238	30545	0.891734
27	28	2019-11-28	917	2019-11	28155	30545	0.921755
28	29	2019-11-29	1194	2019-11	29349	30545	0.960845
29	30	2019-11-30	1196	2019-11	30545	30545	1.000000
30	31	2019-12-01	1182	2019-12	1182	30471	0.038791
31	32	2019-12-02	1186	2019-12	2368	30471	0.077713
32	33	2019-12-03	874	2019-12	3242	30471	0.106396
33	34	2019-12-04	904	2019-12	4146	30471	0.136064
34	35	2019-12-05	932	2019-12	5078	30471	0.166650
35	36	2019-12-06	820	2019-12	5898	30471	0.193561
36	37	2019-12-07	813	2019-12	6711	30471	0.220242
37	38	2019-12-08	1100	2019-12	7811	30471	0.256342
38	39	2019-12-09	818	2019-12	8629	30471	0.283187

知乎 @超哥要努力

结果和前面SQL计算的是一致的。此处同样省略了转换百分比格式的代码，可参考前一篇文章。

expanding函数

分组情况下使用expanding函数需要和groupby结合，注意得到的结果是多重索引，需要取values才能赋值给原dataframe。

```
orderamt = pd.read_excel('orderamt.xlsx')
orderamt['mon'] = orderamt['dt'].dt.strftime('%Y-%m')
orderamt_mon_group = orderamt.groupby('mon').expanding(min_periods=1)['amt'].su
#这里的orderamt_mon_group索引会有两重，我们直接取values的值就可以和原dataframe拼接在一起
orderamt['orderamt_mon_group'] = orderamt_mon_group.values
orderamt
```

24	25	2019-11-25	817	2019-11	24867.0
25	26	2019-11-26	1171	2019-11	26038.0
26	27	2019-11-27	1200	2019-11	27238.0
27	28	2019-11-28	917	2019-11	28155.0
28	29	2019-11-29	1194	2019-11	29349.0
29	30	2019-11-30	1196	2019-11	30545.0
30	31	2019-12-01	1182	2019-12	1182.0
31	32	2019-12-02	1186	2019-12	2368.0
32	33	2019-12-03	874	2019-12	3242.0
33	34	2019-12-04	904	2019-12	4146.0
34	35	2019-12-05	932	2019-12	5078.0
35	36	2019-12-06	820	2019-12	5898.0
36	37	2019-12-07	813	2019-12	6711.0
37	38	2019-12-08	1100	2019-12	7811.0
38	39	2019-12-09	818	2019-12	8629.0

知乎 @超哥要努力

接下来就可以用前面同样的方法，计算分组的总计值，然后求得分组累计百分比了。

rolling函数

通过上文我们知道，rolling函数与expanding函数的代码几乎一样，需要加上window参数。如下所示：

```
orderamt = pd.read_excel('orderamt.xlsx')
orderamt['mon'] = orderamt['dt'].dt.strftime('%Y-%m')
orderamt_mon_group_roll = orderamt.groupby('mon').rolling(len(orderamt),min_pe
#这里的orderamt_mon_group_roll索引会有两重，我们直接取values的值就可以和原dataframe拼接
```

```
orderamt['orderamt_mon_group_roll'] = orderamt_mon_group_roll.values
orderamt
```

24	25	2019-11-25	817	2019-11	24867.0
25	26	2019-11-26	1171	2019-11	26038.0
26	27	2019-11-27	1200	2019-11	27238.0
27	28	2019-11-28	917	2019-11	28155.0
28	29	2019-11-29	1194	2019-11	29349.0
29	30	2019-11-30	1196	2019-11	30545.0
30	31	2019-12-01	1182	2019-12	1182.0
31	32	2019-12-02	1186	2019-12	2368.0
32	33	2019-12-03	874	2019-12	3242.0
33	34	2019-12-04	904	2019-12	4146.0
34	35	2019-12-05	932	2019-12	5078.0
35	36	2019-12-06	820	2019-12	5898.0
36	37	2019-12-07	813	2019-12	6711.0
37	38	2019-12-08	1100	2019-12	7811.0
38	39	2019-12-09	818	2019-12	8629.0

知乎 @[illegible]努力

结果和上面的是一致的。

至此，我们用多种方法实现了对于累计百分比的计算。

小结

本篇我们计算了分组和不分组情况的累计百分比。在MySQL中用了不等值连接的方法，在Hive SQL中使用了sum窗口函数。在pandas中学习了cumsum，expanding，rolling函数，最终都需要将累加值除以总计值得出累计百分比。本文代码较多，您可以在我的公众号后台回复“**对比五**”可以获取本文pdf版本，代码，数据等进行实战，希望对你有所帮助。

◆ ◆ ◆ ◆ ◆

推荐阅读：

1.一场pandas与SQL的巅峰大战

2.一场pandas与SQL的巅峰大战（二）

3.一场pandas与SQL的巅峰大战（三）

4.一场pandas与SQL的巅峰大战（四）