通俗易懂的学会：SQL窗口函数

猴子 猴子SQL 2019-11-19 21:00

一.窗口函数有什么用?

在日常工作中，经常会遇到需要**在每组内排名**，比如下面的业务需求：

> 排名问题：每个部门按业绩来排名
> topN问题：找出每个部门排名前N的员工进行奖励

面对这类需求，就需要使用sql的高级功能窗口函数了。

二.什么是窗口函数?

窗口函数，也叫OLAP函数（Online Anallytical Processing，联机分析处理），可以对数据库数据进行实时分析处理。

窗口函数的基本语法如下：

```
<窗口函数> over (partition by <用于分组的列名>
                 order by <用于排序的列名>)
```

那么语法中的<窗口函数>都有哪些呢?

<窗口函数>的位置，可以放以下两种函数：

1） 专用窗口函数，包括后面要讲到的rank, dense_rank, row_number等专用窗口函数。
2） 聚合函数，如sum. avg, count, max, min等

因为窗口函数是对where或者group by子句处理后的结果进行操作，所以**窗口函数原则上只能写在select子句中**。

三.如何使用?

接下来，就结合实例，给大家介绍几种窗口函数的用法。

1.专用窗口函数rank

例如下图，是班级表中的内容

班级表

学号	班级	成绩
0001	1	86
0002	1	95
0003	2	89
0004	1	83
0005	2	86
0006	3	92
0007	3	86
0008	1	88

猴子

如果我们想在每个班级内按成绩排名，得到下面的结果。

学号	班级	成绩	ranking
0002	1	95	1
0008	1	88	2
0001	1	86	3
0004	1	83	4
0003	2	89	1
0005	2	86	2
0006	3	92	1
0007	3	86	2

猴子

以班级“1”为例，这个班级的成绩“95”排在第1位，这个班级的“83”排在第4位。上面这个结果确实按我们的要求在每个班级内，按成绩排名了。

得到上面结果的sql语句代码如下：

```
select *,
   rank() over (partition by 班级
                order by 成绩 desc) as ranking
from 班级表
```

我们来解释下这个sql语句里的select子句。rank是排序的函数。要求是“每个班级内按成绩排名”，这句话可以分为两部分：

1）每个班级内：按班级分组

partition by用来对表分组。在这个例子中，所以我们指定了按“班级”分组（`partition by 班级`）

2）按成绩排名

order by子句的功能是对分组后的结果进行排序，默认是按照升序（asc）排列。在本例中（`order by 成绩 desc`）是按成绩这一列排序，加了desc关键词表示降序排列。

通过下图，我们就可以理解partiition by（分组）和order by（在组内排序）的作用了。

猴子

窗口函数具备了我们之前学过的group by子句分组的功能和order by子句排序的功能。那么，为什么还要用窗口函数呢?

这是因为，**group by分组汇总后改变了表的行数，一行只有一个类别。而partiition by和rank函数不会减少原表中的行数。**例如下面统计每个班级的人数。

group by分组汇总改变行数

```
select 班级, count(学号)
from 班级表
group by 班级
order by 班级
```

班级	count(学号)
1	4
2	2
3	2

partition by分组汇总行数不变

```
select 班级,
  count(学号) over (partition by 班级
          order by 班级) as
current_count
from 班级表
```

班级	current_count
1	4
1	4
1	4
1	4
2	2
2	2
3	2
3	2

猴子

相信通过这个例子，你已经明白了这个窗口函数的使用：

```
select *,
   rank() over (partition by 班级
                 order by 成绩 desc) as ranking
from 班级表
```

现在我们说回来，为什么叫“窗口”函数呢？这是因为partition by分组后的结果称为“窗口”，这里的窗口不是我们家里的门窗，而是表示“范围”的意思。

简单来说，窗口函数有以下功能：
1）同时具有分组和排序的功能
2）不减少原表的行数
3）语法如下：

```
<窗口函数> over (partition by <用于分组的列名>
```

```
                order by <用于排序的列名>)
```

2.其他专业窗口函数

专用窗口函数rank, dense_rank, row_number有什么区别呢?

它们的区别我举个例子，你们一下就能看懂:

```
select *,
   rank() over (order by 成绩 desc) as ranking,
   dense_rank() over (order by 成绩 desc) as dese_rank,
   row_number() over (order by 成绩 desc) as row_num
from 班级表
```

得到结果:

学号	班级	成绩	ranking	dese_rank	row_num
0002	1	95	1	1	1
0006	3	92	2	2	2
0003	2	89	3	3	3
0008	1	88	4	4	4
0001	1	86	5	5	5
0005	2	86	5	5	6
0007	3	86	5	5	7
0004	1	83	8	6	8

猴子

从上面的结果可以看出:

> rank函数：这个例子中是5位，5位，5位，8位，也就是如果有并列名次的行，会占用下一名次的位置。比如正常排名是1，2，3，4，但是现在前3名是并列的名次，结果是：**1，1，1**，4。
>
> dense_rank函数：这个例子中是5位，5位，5位，6位，也就是如果有并列名次的行，不占用下一名次的位置。比如正常排名是1，2，3，4，但是现在前3名是并列的名次，结果是：**1，1，1**，2。
>
> row_number函数：这个例子中是5位，6位，7位，8位，也就是不考虑并列名次的情况。比如前3名是并列的名次，排名是正常的**1，2，3**，4。
>
> 这三个函数的区别如下：

成绩	ranking	dese_rank	row_num
100	1	1	1
100	1	1	2
100	1	1	3
98	4	2	4

猴子

最后，需要强调的一点是：在上述的这三个专用窗口函数中，函数后面的括号不需要任何参数，保持()空着就可以。

现在，大家对窗口函数有一个基本了解了吗？

3.聚合函数作为窗口函数

聚和窗口函数和上面提到的专用窗口函数用法完全相同，只需要把聚合函数写在窗口函数的位置即可，但是函数后面括号里面不能为空，需要指定聚合的列名。

我们来看一下窗口函数是聚合函数时，会出来什么结果：

```
select *,
    sum(成绩) over (order by 学号) as current_sum,
    avg(成绩) over (order by 学号) as current_avg,
    count(成绩) over (order by 学号) as current_count,
    max(成绩) over (order by 学号) as current_max,
    min(成绩) over (order by 学号) as current_min
from 班级表
```

得到结果：

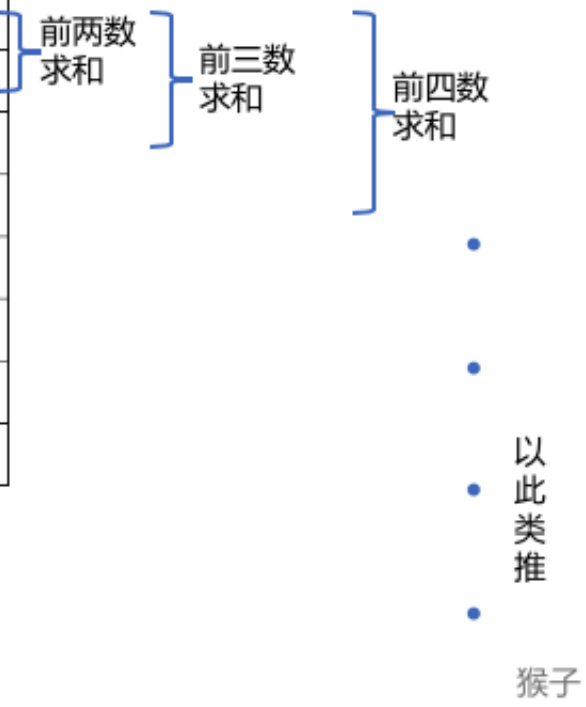
学号	班级	成绩	current_sum	current_avg	current_count	current_max	current_min
0001	1	86	86	86.0000	1	86	86
0002	1	95	181	90.5000	2	95	86
0003	2	89	270	90.0000	3	95	86
0004	1	83	353	88.2500	4	95	83
0005	2	86	439	87.8000	5	95	83
0006	3	92	531	88.5000	6	95	83
0007	3	86	617	88.1429	7	95	83
0008	1	88	705	88.1250	8	95	83

猴子

有发现什么吗？我单独用sum举个例子：

sum作为窗口函数

学号	班级	成绩	current_sum
0001	1	86	86
0002	1	95	181
0003	2	89	270
0004	1	83	353
0005	2	86	439
0006	3	92	531
0007	3	86	617
0008	1	88	705



如上图，聚合函数sum在窗口函数中，是对自身记录、及位于自身记录以上的数据进行求和的结果。比如0004号，在使用sum窗口函数后的结果，是对0001，0002，0003，0004号的成绩求和，若是0005号，则结果是0001号~0005号成绩的求和，以此类推。

不仅是sum求和，平均、计数、最大最小值，也是同理，都是针对自身记录、以及自身记录之上的所有数据进行计算，现在再结合刚才得到的结果（下图），是不是理解起来容易多了？

学号	班级	成绩	current_sum	current_avg	current_count	current_max	current_min
0001	1	86	86	86.0000	1	86	86
0002	1	95	181	90.5000	2	95	86
0003	2	89	270	90.0000	3	95	86
0004	1	83	353	88.2500	4	95	83
0005	2	86	439	87.8000	5	95	83
0006	3	92	531	88.5000	6	95	83
0007	3	86	617	88.1429	7	95	83
0008	1	88	705	88.1250	8	95	83

猴子

比如0005号后面的聚合窗口函数结果是：学号0001~0005五人成绩的总和、平均、计数及最大最小值。

如果想要知道所有人成绩的总和、平均等聚合结果，看最后一行即可。

这样使用窗口函数有什么用呢?

聚合函数作为窗口函数，可以在每一行的数据里直观的看到，截止到本行数据，统计数据是多少（最大值、最小值等）。同时可以看出每一行数据，对整体统计数据的影响。

四.注意事项

partition子句可是省略，省略就是不指定分组，结果如下，只是按成绩由高到低进行了排序:

```
select *,
   rank() over (order by 成绩 desc) as ranking
from 班级表
```

得到结果：

学号	班级	成绩	ranking
0002	1	95	1
0006	3	92	2
0003	2	89	3
0008	1	88	4
0001	1	86	5
0005	2	86	5
0007	3	86	5
0004	1	83	8

猴子

但是，这就失去了窗口函数的功能，所以一般不要这么使用。

四.总结

1.窗口函数语法

```
<窗口函数> over (partition by <用于分组的列名>
                 order by <用于排序的列名>)
```

<窗口函数>的位置，可以放以下两种函数：

1） 专用窗口函数，比如rank, dense_rank, row_number等
2） 聚合函数，如sum. avg, count, max, min等

2.窗口函数有以下功能：

1）同时具有分组（partition by）和排序（order by）的功能
2）不减少原表的行数，所以经常用来在每组内排名

3.注意事项

窗口函数原则上只能写在select子句中

4.窗口函数使用场景

1）业务需求“**在每组内排名**”，比如：

> 排名问题：每个部门按业绩来排名
> topN问题：找出每个部门排名前N的员工进行奖励

下一次会跟大家分享一些窗口函数的面试题，从而让各位在面试、工作中都能遇到这类问题，就想到哦，这用窗口函数就可以解决。