武汉生物工程学院

MySQL数据库

主讲教师：朱　　华

窗口函数使用场景

医院看病，怎样知道上次就医距现在的时间？环比如何计算？怎么样得到各部门工资排名前N名员工列表？查找各部门每人工资占部门总工资的百分比？

对于这样的需求，使用传统的SQL实现起来比较困难。这类需求都有一个共同的特点，需要在单表中满足某些条件的记录集内部做一些函数操作，不是简单的表连接，也不是简单的聚合可以实现的，通常会让写SQL的同学焦头烂额、绞尽脑汁，费了大半天时间写出来一堆长长的晦涩难懂的自连接SQL，且性能低下，难以维护。

要解决此类问题，最方便的就是使用窗口函数。

什么叫窗口

窗口的概念非常重要，它可以理解为记录集合，窗口函数也就是在满足某种条件的记录集合上执行的特殊函数。

对于每条记录都要在此窗口内执行函数，有的函数虽然记录不同，窗口大小都是固定的，这种属于静态窗口；有的函数则相反，不同的记录对应着不同的窗口，这种动态变化的窗口叫滑动窗口。

窗口函数

从version 8.0开始，MySQL支持在查询中使用窗口函数。

row_number()

row_number()（分组）排序编号，按照表中某一字段分组，再按照某一字段排序，对已有的数据生成一个编号。语法格式为：

row_number() over (parttion by 字段名 order by 字段名) as 编号;

窗口函数

例6.42 查询学生选课表中每门课程的最高分，按照课程编号分组后按成绩降序排列，返回每一组数据的第一条记录的成绩，即每门课的最高分。

Sql语句如下：

```
select * from (select row_number() over
(partition by 课程号 order by 成绩 desc) as
row_num,学号,课程号,成绩 from choose) t
where row_num=1;
```

窗口函数

```
mysql> select * from (select row_number() over (partition by 课程号 order by 成绩 desc)
    -> as row_num,学号,课程号,成绩 from choose) t where row_num=1;
+---------+----------+-----------+--------+
| row_num | 学号     | 课程号    | 成绩   |
+---------+----------+-----------+--------+
|       1 | 01640403 |         1 |     80 |
|       1 | 01640403 |         2 |     90 |
|       1 | 01640401 |         3 |     60 |
+---------+----------+-----------+--------+
3 rows in set (0.00 sec)

mysql> select 学号,课程号,max(成绩) from choose group by 课程号 order by 成绩 desc;
+----------+-----------+-------------+
| 学号     | 课程号    | max(成绩)   |
+----------+-----------+-------------+
| 01640401 |         3 |          60 |
| 01640401 |         1 |          80 |
| 01640401 |         2 |          90 |
+----------+-----------+-------------+
3 rows in set (0.00 sec)
```

窗口函数

rank()

rank()类似于row_number()，也是排序功能，如果表中有两条完全一样的数据， row_number()编号的时候，这两条数据被编了两个不同的号；而rank()在排序条件一样的情况下，其编号也一样。如例6.42的sql语句修改为：

```
select * from (select rank() over
(partition by 课程号 order by 成绩 desc) as
row_num,学号,课程号,成绩 from choose) t
where row_num=1;
```

窗口函数

```
mysql> select * from (select rank() over (part
e) t where row_num=1;
+---------+----------+-----------+--------+
| row_num | 学号     | 课程号    | 成绩   |
+---------+----------+-----------+--------+
|       1 | 01640403 |         1 |     80 |
|       1 | 01640403 |         2 |     90 |
|       1 | 01640401 |         3 |     60 |
|       1 | 01640404 |         3 |     60 |
+---------+----------+-----------+--------+
4 rows in set (0.00 sec)

mysql>     select * from (select row_number()
from choose) t where row_num=1;
+---------+----------+-----------+--------+
| row_num | 学号     | 课程号    | 成绩   |
+---------+----------+-----------+--------+
|       1 | 01640403 |         1 |     80 |
|       1 | 01640403 |         2 |     90 |
|       1 | 01640401 |         3 |     60 |
+---------+----------+-----------+--------+
3 rows in set (0.00 sec)
```

窗口函数

avg、sum等聚合函数在窗口函数中的增强

可以在聚合函数中使用窗口功能，比如sum(amount)over(parttion by user_no order by create_date) as sum_amont，实现一个累积计算的功能。

窗口函数

```
mysql> select * from 销售表;
+-----------+--------+-----------+
| 销售员    | 年度   | 销售额    |
+-----------+--------+-----------+
| 张三      |   2016 |    100.00 |
| 张三      |   2017 |    150.00 |
| 张三      |   2018 |    200.00 |
| 李四      |   2016 |    150.00 |
| 李四      |   2017 |    100.00 |
| 李四      |   2018 |    200.00 |
| 王五      |   2016 |    200.00 |
| 王五      |   2017 |    150.00 |
| 王五      |   2018 |    250.00 |
+-----------+--------+-----------+
9 rows in set (0.00 sec)

mysql> SELECT 年度,销售员,销售额,SUM(销售额) OVER (PARTITION BY 年度) 总销售额 FROM 销售表;
+--------+-----------+-----------+--------------+
| 年度   | 销售员    | 销售额    | 总销售额     |
+--------+-----------+-----------+--------------+
|   2016 | 张三      |    100.00 |       450.00 |
|   2016 | 李四      |    150.00 |       450.00 |
|   2016 | 王五      |    200.00 |       450.00 |
|   2017 | 张三      |    150.00 |       400.00 |
|   2017 | 李四      |    100.00 |       400.00 |
|   2017 | 王五      |    150.00 |       400.00 |
|   2018 | 张三      |    200.00 |       650.00 |
|   2018 | 李四      |    200.00 |       650.00 |
|   2018 | 王五      |    250.00 |       650.00 |
+--------+-----------+-----------+--------------+
9 rows in set (0.12 sec)
```

窗口函数

这里，sum()函数充当了窗口函数，得到了根据年度计算出的销售额的总和列，但是又不像它作为聚合函数使用时一样，这里的结果保留了每一行的信息。

原因就在于窗口函数的执行顺序（逻辑上的）是在FROM，JOIN，WHERE，GROUP BY，HAVING之后，在ORDER BY，LIMIT，SELECT DISTINCT之前。它执行时GROUP BY的聚合过程已经完成了，所以不会再产生数据聚合。

窗口函数

ntile()

ntile(n)将数据按照某些排序分成n组，在n组数据中获取其中一部分数据。

lag、lead函数

lag(column,n)和lead(column,n) 函数将数据按照某种排序规则的上（下）n行数据的某个字段。

窗口函数

cte公用表表达式

cte公用表表达式有两种用法，非递归的cte和递归的cte。非递归的cte可以用来增加代码的可读性，增加逻辑的结构化表达。对于一句sql有几十行甚至上百行，使用cte分段解决。

窗口函数

例6.45 查询选课人数最多的课程号和选课人数，sql语句如下：

```
with cte as(
select count(学号) a,课程号
from choose group by 课程号)
select max(a),课程号 from cte;
```

窗口函数

```
mysql>      with cte as(
    ->      select count(学号) a,课程号
    ->      from choose group by 课程号)
    ->      select max(a),课程号 from cte;
+---------+-------------+
| max(a)  | 课程号      |
+---------+-------------+
|       3 |           1 |
+---------+-------------+
1 row in set (0.05 sec)
```

窗口函数小结

窗口函数和普通聚合函数也很容易混淆，二者区别是聚合函数是将多条记录聚合为一条；而窗口函数是每条记录都会执行，有几条记录执行完还是几条。

窗口函数非常有意思，对于一些使用常规思维无法实现的SQL需求，大家尝试一下窗口函数吧，相信会有意想不到的收获。