08丨什么是SQL的聚集函数，如何利用它们汇总表的数据？

2019-06-28 陈旸

我们上节课讲到了SQL函数，包括算术函数、字符串函数、日期函数和转换函数。实际上SQL函数还有一种，叫做聚集函数，它是对一组数据进行汇总的函数，输入的是一组数据的集合，输出的是单个值。通常我们可以利用聚集函数汇总表的数据，如果稍微复杂一些，我们还需要先对数据做筛选，然后再进行聚集，比如先按照某个条件进行分组，对分组条件进行筛选，然后得到筛选后的分组的汇总信息。

有关今天的内容，你重点需要掌握以下几个方面：

1. 聚集函数都有哪些，能否在一条SELECT语句中使用多个聚集函数；
2. 如何对数据进行分组，并进行聚集统计；
3. 如何使用HAVING过滤分组，HAVING和WHERE的区别是什么。

聚集函数都有哪些

SQL中的聚集函数一共包括5个，可以帮我们求某列的最大值、最小值和平均值等，它们分别是：

函数	说明
COUNT()	总行数
MAX()	最大值
MIN()	最小值
SUM()	求和
AVG()	平均值

这些函数你可能已经接触过，我们再来简单复习一遍。我们继续使用heros数据表，对王者荣耀的英雄数据进行聚合。

如果我们想要查询最大生命值大于6000的英雄数量。

```
SQL：SELECT COUNT(*) FROM heros WHERE hp_max > 6000
```

运行结果为41。

如果想要查询最大生命值大于6000，且有次要定位的英雄数量，需要使用COUNT函数。

```
SQL：SELECT COUNT(role_assist) FROM heros WHERE hp_max > 6000
```

运行结果是 23。

需要说明的是，有些英雄没有次要定位，即role_assist为NULL，这时COUNT(role_assist)会忽略值为NULL的数据行，而COUNT(*)只是统计数据行数，不管某个字段是否为NULL。

如果我们想要查询射手（主要定位或者次要定位是射手）的最大生命值的最大值是多少，需要使用MAX函数。

```
SQL：SELECT MAX(hp_max) FROM heros WHERE role_main = '射手' or role_assist = '射手'
```

运行结果为6014。

你能看到，上面的例子里，都是在一条SELECT语句中使用了一次聚集函数，实际上我们也可以在一条SELECT语句中进行多项聚集函数的查询，比如我们想知道射手（主要定位或者次要定位

是射手）的英雄数、平均最大生命值、法力最大值的最大值、攻击最大值的最小值，以及这些英雄总的防御最大值等汇总数据。

如果想要知道英雄的数量，我们使用的是COUNT(*)函数，求平均值、最大值、最小值，以及总的防御最大值，我们分别使用的是AVG、MAX、MIN和SUM函数。另外我们还需要对英雄的主要定位和次要定位进行筛选，使用的是WHERE role_main = '射手' or role_assist = '射手'。

```
SQL: SELECT COUNT(*), AVG(hp_max), MAX(mp_max), MIN(attack_max), SUM(defense_max) FROM heros WH
```

运行结果：

COUNT(*)	AVG(hp_max)	MAX(mp_max)	MIN(attack_max)	SUM(defense_max)
10	5798.5	1784	362	3333

需要说明的是AVG、MAX、MIN等聚集函数会自动忽略值为NULL的数据行，MAX和MIN函数也可以用于字符串类型数据的统计，如果是英文字母，则按照A--Z的顺序排列，越往后，数值越大。如果是汉字则按照全拼拼音进行排列。比如：

```
SQL：SELECT MIN(CONVERT(name USING gbk)), MAX(CONVERT(name USING gbk)) FROM heros
```

运行结果：

MIN(CONVERT(name USING gbk))	MAX(CONVERT(name USING gbk))
阿轲	庄周

需要说明的是，我们需要先把name字段统一转化为gbk类型，使用CONVERT(name USING gbk)，然后再使用MIN和MAX取最小值和最大值。

我们也可以对数据行中不同的取值进行聚集，先用DISTINCT函数取不同的数据，然后再使用聚集函数。比如我们想要查询不同的生命最大值的英雄数量是多少。

```
SQL: SELECT COUNT(DISTINCT hp_max) FROM heros
```

运行结果为61。

实际上在heros这个数据表中，一共有69个英雄数量，生命最大值不一样的英雄数量是61个。

假如我们想要统计不同生命最大值英雄的平均生命最大值，保留小数点后两位。首先需要取不同生命最大值，即DISTINCT hp_max，然后针对它们取平均值，即AVG(DISTINCT hp_max)，最后再针对这个值保留小数点两位，也就是ROUND(AVG(DISTINCT hp_max), 2)。

```
SQL: SELECT ROUND(AVG(DISTINCT hp_max), 2) FROM heros
```

运行结果为6653.84。

你能看到，如果我们不使用DISTINCT函数，就是对全部数据进行聚集统计。如果使用了DISTINCT函数，就可以对数值不同的数据进行聚集。一般我们使用MAX和MIN函数统计数据行的时候，不需要再额外使用DISTINCT，因为使用DISTINCT和全部数据行进行最大值、最小值的统计结果是相等的。

如何对数据进行分组，并进行聚集统计

我们在做统计的时候，可能需要先对数据按照不同的数值进行分组，然后对这些分好的组进行聚集统计。对数据进行分组，需要使用GROUP BY子句。

比如我们想按照英雄的主要定位进行分组，并统计每组的英雄数量。

```
SQL: SELECT COUNT(*), role_main FROM heros GROUP BY role_main
```

运行结果（6条记录）：

COUNT(*)	role_main
10	坦克
18	战士
19	法师
6	辅助
10	射手
6	刺客

如果我们想要对英雄按照次要定位进行分组，并统计每组英雄的数量。

```
SELECT COUNT(*), role_assist FROM heros GROUP BY role_assist
```

运行结果：（6条记录）

COUNT(*)	role_assist
6	战士
10	坦克
5	辅助
40	NULL
2	法师
6	刺客

你能看出如果字段为NULL，也会被列为一个分组。在这个查询统计中，次要定位为NULL，即只有一个主要定位的英雄是40个。

我们也可以使用多个字段进行分组，这就相当于把这些字段可能出现的所有的取值情况都进行分组。比如，我们想要按照英雄的主要定位、次要定位进行分组，查看这些英雄的数量，并按照这些分组的英雄数量从高到低进行排序。

```
SELECT COUNT(*) as num, role_main, role_assist FROM heros GROUP BY role_main, role_assist ORDER BY r
```

运行结果：（19条记录）

num	role_main	role_assist
12	法师	
9	射手	
8	战士	
6	战士	坦克
5	坦克	
3	刺客	
3	坦克	辅助
3	辅助	
3	战士	刺客
3	刺客	战士
2	法师	辅助
2	法师	刺客
2	法师	坦克
2	辅助	坦克
2	坦克	战士
1	射手	刺客
1	辅助	法师
1	法师	战士
1	战士	法师

如何使用HAVING过滤分组，它与WHERE的区别是什么？

当我们创建出很多分组的时候，有时候就需要对分组进行过滤。你可能首先会想到WHERE子句，实际上过滤分组我们使用的是HAVING。HAVING的作用和WHERE一样，都是起到过滤的作用，只不过WHERE是用于数据行，而HAVING则作用于分组。

比如我们想要按照英雄的主要定位、次要定位进行分组，并且筛选分组中英雄数量大于5的组，最后按照分组中的英雄数量从高到低进行排序。

首先我们需要获取的是英雄的数量、主要定位和次要定位，即SELECT COUNT(*) as num,

role_main, role_assist。然后按照英雄的主要定位和次要定位进行分组，即GROUP BY role_main, role_assist，同时我们要对分组中的英雄数量进行筛选，选择大于5的分组，即HAVING num > 5，然后按照英雄数量从高到低进行排序，即ORDER BY num DESC。

```
SQL: SELECT COUNT(*) as num, role_main, role_assist FROM heros GROUP BY role_main, role_assist HAVIN
```

运行结果：（4条记录）

num	role_main	role_assist
12	法师	
9	射手	
8	战士	
6	战士	坦克

你能看到还是上面这个分组，只不过我们按照数量进行了过滤，筛选了数量大于5的分组进行输出。如果把HAVING替换成了WHERE，SQL则会报错。对于分组的筛选，我们一定要用HAVING，而不是WHERE。另外你需要知道的是，HAVING支持所有WHERE的操作，因此所有需要WHERE子句实现的功能，你都可以使用HAVING对分组进行筛选。

我们再来看个例子，通过这个例子查看一下WHERE和HAVING进行条件过滤的区别。筛选最大生命值大于6000的英雄，按照主要定位、次要定位进行分组，并且显示分组中英雄数量大于5的分组，按照数量从高到低进行排序。

```
SQL: SELECT COUNT(*) as num, role_main, role_assist FROM heros WHERE hp_max > 6000 GROUP BY role_
```

运行结果：（2条记录）

num	role_main	role_assist
8	战士	
6	战士	坦克

你能看到，还是针对上一个例子的查询，只是我们先增加了一个过滤条件，即筛选最大生命值大于6000的英雄。这里我们就需要先使用WHERE子句对最大生命值大于6000的英雄进行条件过滤，然后再使用GROUP BY进行分组，使用HAVING进行分组的条件判断，然后使用ORDER BY进行排序。

总结

今天我对SQL的聚集函数进行了讲解。通常我们还会对数据先进行分组，然后再使用聚集函数统计不同组的数据概况，比如数据行数、平均值、最大值、最小值以及求和等。我们也可以使用HAVING对分组进行过滤，然后通过ORDER BY按照某个字段的顺序进行排序输出。有时候你能看到在一条SELECT语句中，可能会包括多个子句，用WHERE进行数据量的过滤，用GROUP BY进行分组，用HAVING进行分组过滤，用ORDER BY进行排序.....

你要记住，在SELECT查询中，关键字的顺序是不能颠倒的，它们的顺序是：

```
SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ...
```

另外需要注意的是，使用GROUP BY进行分组，如果想让输出的结果有序，可以在GROUP BY后使用ORDER BY。因为GROUP BY只起到了分组的作用，排序还是需要通过ORDER BY来完成。

我今天对SQL的聚集函数以及SQL查询中的关键字顺序进行了讲解，但你还是需要通过训练加深理解，基于heros数据表，请你写出下面2个SQL查询语句：

1. 筛选最大生命值大于6000的英雄，按照主要定位进行分组，选择分组英雄数量大于5的分组，按照分组英雄数从高到低进行排序，并显示每个分组的英雄数量、主要定位和平均最大生命值。

2. 筛选最大生命值与最大法力值之和大于7000的英雄，按照攻击范围来进行分组，显示分组的英雄数量，以及分组英雄的最大生命值与法力值之和的平均值、最大值和最小值，并按照分组英雄数从高到低进行排序，其中聚集函数的结果包括小数点后两位。

欢迎你在评论区与我分享你的答案，如果你觉得这篇文章有帮助，欢迎把它分享给你的朋友或者同事，一起切磋交流一下。

精选留言

ack

5

练习题

1.SELECT COUNT(*) AS num,role_main,AVG(hp_max) FROM heros WHERE hp_max > 6000 GROUP BY role_main HAVING num>5 ORDER BY num DESC;

2.SELECT COUNT(*) AS num,ROUND(MAX(hp_max+mp_max),2),ROUND(AVG(hp_max+mp_max),2),ROUND(MIN(hp_max+mp_max),2) FROM heros WHERE hp_max+mp_max > 7000 GROUP BY attack_range ORDER BY num DESC;

2019-06-28

作者回复

正确

2019-06-28

mickey

2

/*

1.筛选最大生命值大于6000的英雄，按照主要定位进行分组，选择分组英雄数量大于5的分组，按照分组英雄数从高到低进行排序，并显示每个分组的英雄数量、主要定位和平均最大生命值。

*/

```
SELECT count(*) as num, role_main, AVG(hp_max)
FROM heros
WHERE hp_max > 6000
GROUP BY role_main
HAVING num > 5
ORDER BY num DESC
```

```
num role_main AVG(hp_max)
---------------------------------
17 战士 7028
10 坦克 8312.4
6 法师 6417
```

/*

2.筛选最大生命值与最大法力值之和大于7000的英雄，按照攻击范围来进行分组，
显示分组的英雄数量，以及分组英雄的最大生命值与法力值之和的平均值、最大值和最小值，
并按照分组英雄数从高到低进行排序，其中聚集函数的结果包括小数点后两位。

*/

```
SELECT count(*) as num, ROUND(AVG(hp_max + mp_max), 2), MAX(hp_max + mp_max), MIN(hp_max + mp_max)
FROM heros
WHERE hp_max + mp_max > 7000
GROUP BY attack_range
HAVING num > 5
ORDER BY num DESC
```

```
num, ROUND(AVG(hp_max + mp_max), 2), MAX(hp_max + mp_max), MIN(hp_max + mp_max)
-----------------------------------------------------------------
62 8272.53 11036 7025
```

2019-06-28

作者回复

SQL正确，最后结果贴的不太完整

2019-06-28

Taozi

2

练习2中反复出现的hp_max+mp_max可以绑定到一个变量吗？

2019-06-28

Amanda

2

一个发现：虽然 SELECT 的执行顺序在 GROUP BY 和 HAVING 后面，但对于SELECT中列的别名都可以使用。

MySQL中

1. > SELECT COUNT(*) as num, role_main, AVG(hp_max) FROM heros
-> WHERE hp_max>6000
-> GROUP BY role_main
-> HAVING COUNT(*)>5
-> ORDER BY COUNT(*) DESC;
+-----+-----------+-------------+
| num | role_main | AVG(hp_max) |
+-----+-----------+-------------+
17	战士	7028
10	坦克	8312.4
6	法师	6417
+-----+-----------+-------------+

2. > SELECT COUNT(*) num, ROUND(AVG(hp_max+mp_max), 2) avg, ROUND(MAX(hp_max+mp_max), 2) max, ROUND(MIN(hp_max+mp_max), 2) min FROM heros
-> WHERE (hp_max+mp_max)>7000
-> GROUP BY attack_range
-> ORDER BY num DESC;
+-----+---------+----------+---------+
| num | avg | max | min |
+-----+---------+----------+---------+
| 36 | 8654.42 | 11036.00 | 7117.00 |
| 26 | 7743.77 | 8737.00 | 7025.00 |
+-----+---------+----------+---------+

2019-06-28

作者回复

在执行顺序上，SELECT字段在GROUP BY和HAVING之后，不过在SELECT字段之前，已经计算了聚集函数，也就是COUNT(*) as num。聚集函数的计算在GROUP BY之后，HAVING之前

2019-06-28

圆子蛋

2

1.SELECT COUNT(*) as num,role_main,AVG(hp_max) FROM heros WHERE hp_max > 6000 GROUP BY role_main HAVING num>5 ORDER BY num DESC;

2.SELECT COUNT(*) as num,ROUND(MAX(hp_max+mp_max),2),ROUND(AVG(hp_max+mp_max),2),ROUND(MIN(hp_max+mp_max),2) FROM heros WHERE (hp_max+mp_max) > 7000 GROUP BY attack_range ORDER BY num DESC;

老师在“如何对数据进行分组，并进行聚集统计”的第三个例子里，COUNT(*) 后面没有加 as num，但是 ORDER BY 里直接出现了 num？

2019-06-28

作者回复

COUNT(*)后面应该有 as num

2019-06-28

mickey 2

有个错误：

文中“比如，我们想要按照英雄的主要定位、次要定位进行分组，查看这些英雄的数量，并按照这些分组的英雄数量从高到低进行排序。”的SQL语句：SQL: SELECT COUNT(*), role_main, role_assist FROM heros GROUP BY role_main, role_assist ORDER BY num DESC

在MySQL里会报错：[Err] 1054 - Unknown column 'num' in 'order clause'

要改为：SELECT COUNT(*) as num, role_main, role_assist FROM heros GROUP BY role_main, role_assist ORDER BY num DESC;

2019-06-28

编辑回复

您好，文章已进行更正，谢谢您的反馈。

2019-06-28

安静的boy 2

where先对数据进行排序，group by再进行分组。让我对数据筛选和分组恍然大悟！

2019-06-28

作者回复

理解了HAVING和WHERE的区别，就了解了分组过滤和条件过滤。还有SELECT语句种的关键字的顺序：SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ...

2019-06-28

leol 0

虽然晚一节，作业还是要交的

select count(*) as num,role_main ,avg(hp_max) from heros where hp_max>6000
group by role_main having num>5 order by num desc ;

select count(*) as num ,round(avg(hp_max+mp_max),2),round(max(hp_max+mp_max),2),round(min(hp_max+mp_max),2) from heros where hp_max+mp_max>7000
group by attack_range order by num desc;

2019-07-01

华夏 0

SELECT COUNT(*) as num, role_main, AVG(hp_max) FROM heros WHERE hp_max > 6000 GROUP BY role_main HAVING num > 5 ORDER BY num DESC;

```
+-----+-----------+------------+
| num | role_main | AVG(hp_max) |
+-----+-----------+------------+
| 17 | 战士 | 7028 |
| 10 | 坦克 | 8312.4 |
```

| 6 | 法师 | 6417 |
+-----+----------+------------+
3 rows in set (0.00 sec)

SELECT COUNT(*) as num, attack_range, ROUND(AVG(hp_max+mp_max),2) as avg_, ROUND(MAX(hp_max+mp_max),2) as max_, ROUND(MIN(hp_max+mp_max),2) as min_ FROM heros WHERE hp_max + mp_max > 7000 GROUP BY attack_range ORDER BY num DESC;
+-----+--------------+---------+----------+---------+
| num | attack_range | avg_ | max_ | min_ |
+-----+--------------+---------+----------+---------+
| 36 | 近战 | 8654.42 | 11036.00 | 7117.00 |
| 26 | 远程 | 7743.77 | 8737.00 | 7025.00 |
+-----+--------------+---------+----------+---------+
2 rows in set (0.00 sec)

2019-06-30

莫莫

0

1、
SELECT COUNT(*) AS `sum`,role_main AS '主要定位',AVG(hp_max) AS '最大生命值' FROM heros
WHERE hp_max>6000 GROUP BY role_main HAVING `sum`>5 ORDER BY `sum` DESC

2、
SELECT COUNT(*) AS '英雄数量',ROUND(AVG(hp_max+mp_max),2) AS `avg`,ROUND(MAX(hp_max+mp_max),2) AS `max`,ROUND(MIN(hp_max+mp_max),2) AS `min`
FROM heros
WHERE (hp_max+mp_max)>7000 GROUP BY attack_range ORDER BY COUNT(*) DESC

2019-06-30

莫莫

0

1、
SELECT COUNT(*) AS `sum`,role_main AS '主要定位',AVG(hp_max) AS '最大生命值' FROM heros
WHERE hp_max>6000 GROUP BY role_main HAVING `sum`>5 ORDER BY `sum` DESC

2、
SELECT COUNT(*) AS '英雄数量',ROUND(AVG(hp_max+mp_max),2) AS `avg`,ROUND(MAX(hp_max+mp_max),2) AS `max`,ROUND(MIN(hp_max+mp_max),2) AS `min`
FROM heros
WHERE (hp_max+mp_max)>7000 GROUP BY attack_range

2019-06-30

未来的胡先森

0

交作业了

```
SELECT COUNT(*),role_main ,AVG(hp_max)
FROM heros
WHERE hp_max>6000
GROUP BY role_main
HAVING COUNT(*)>5
ORDER BY COUNT(*) DESC;

SELECT COUNT(*),ROUND( AVG(hp_max+mp_max),2),ROUND(MAX(hp_max+mp_max),2),
ROUND(MIN(hp_max+mp_max),2)
FROM heros
WHERE hp_max+mp_max>7000
GROUP BY attack_range
ORDER BY COUNT(*) DESC;
```

2019-06-30

Neo 0

1. SELECT COUNT(*) AS hero_num, role_main, AVG(hp_max) FROM heros where hp_max > 6000 GROUP BY role_main HAVING hero_num > 5;
2. SELECT COUNT(*) AS hero_num, ROUND(AVG(hp_max+mp_max), 2), MAX(hp_max+mp_max), MIN(hp_max+mp_max) FROM heros WHERE (hp_max+mp_max)>7000 GROUP BY attack_max ORDER BY hero_num ASC;

2019-06-30

Hoo-Ah 0

SELECT ... FROM ... WHERE ... GROUP BY ... HAVING ... ORDER BY ...

2019-06-30

田佳伟 0

老师，看了前几节课，讲述的都是sql的基本用法，没有涉及到性能问题，其实很多方式是不能用在大流量的生产环境的，以后的章节会涉及到高效查询的内容吗？

2019-06-29

psnail 0

如果将 select ... from ... where ...group by ... having..改为select ... (select. from. where. group by.)...where...是不是一样？不知道那种效率高？优化器处理逻辑是否一样？

2019-06-29

微凉 0

```
1.SELECT
COUNT(*) AS num,
role_main,
AVG(hp_max)
```

```
FROM
heros
WHERE
hp_max > 6000
GROUP BY
role_main
HAVING
num > 5
ORDER BY
num DESC

2.SELECT
COUNT(*) AS num,
ROUND(MAX(hp_max + mp_max), 2),
ROUND(AVG(hp_max + mp_max), 2),
ROUND(MIN(hp_max + mp_max), 2)
FROM
heros
WHERE
hp_max + mp_max > 7000
GROUP BY
attack_range
ORDER BY
num DESC
```

2019-06-29

遇见阳光 0

老师，您好，我有个问题问下: 按照sql执行顺序 having在select之前，按照理论来讲having应该不能使用select中的别名 但是实际是可以的。请问这是什么原因

2019-06-29

Geek_669849 0

```
SELECT
COUNT(*) num,
role_main,
AVG( hp_max ) hp_avg
FROM
heros
WHERE
hp_max > 6000
GROUP BY
role_main
HAVING
```

```
COUNT(*) > 5
ORDER BY
COUNT(num) DESC ;

SELECT
COUNT(*) num,
ROUND(AVG(hp_max + mp_max), 2),
ROUND(MAX(hp_max + mp_max), 2),
ROUND(MIN(hp_max + mp_max), 2)
FROM
heros
WHERE
(hp_max + mp_max) > 7000
GROUP BY
attack_range
ORDER BY
COUNT(*) desc
```

2019-06-29

作者回复

正确

2019-06-29

supermouse

0

思考题 1：

```
SELECT
COUNT(*) AS num, role_main, AVG(hp_max)
FROM
heros
WHERE
hp_max > 6000
GROUP BY role_main
HAVING num > 5
ORDER BY num DESC;
```

思考题 2：

```
SELECT
COUNT(*) AS num,
attack_range,
ROUND(AVG(hp_max + mp_max), 2),
ROUND(MAX(hp_max + mp_max), 2),
ROUND(MIN(hp_max + mp_max), 2)
FROM
heros
WHERE
```

```
hp_max + mp_max > 7000
GROUP BY attack_range
ORDER BY num DESC;
```

2019-06-29

作者回复

正确

2019-06-29