MySQL数据库函数

1.字符函数

1.concat（）函数是将两个字符进行连接起来。

```
mysql> select concat(id,age) as patn from tp1;
+------------------+
|     path         |
|------------------|
|    lijinze19     |
|    zhanbinbin18  |
|    zhangweijie18 |
|    yanjiayi17    |
+------------------+
```

2.concat_ws()是使用指定的分割符进行切割如使用\则两个字符之间就有\分隔开 其次 这个函数可以运用到多个函数当中。

```
mysql> select concat_ws('/'.'id','age','sex',);
+---------------------------------------+
|   concat_ws('/','id','age','sex')     |
|---------------------------------------|
|            id/age/user                |
+---------------------------------------+
```

3.format（）是将数字格式化，在数字后面定义几个数则保留几位小数；

```
mysql> select format(123123.4567,3);
+--------------------------+
|  format(123123.4567,3)   |
+--------------------------+
|   123123.457             |
+--------------------------+
msyql> select format(123123.4567,2)
+--------------------------+
|   format(123123.4567,2)  |
+--------------------------+
|   123123.46              |
+--------------------------+
```

4.lower()&upper() 这两个函数是转化大小写的 lower()是将定义都转化为小写 upper() 则是将定义转化为大写的函数。

```
msyql> select lower('LMONKEY');
+-------------------------+
|     lower('LMONKEY')    |
+-------------------------+
|      lmonkey            |
+-------------------------+

mysql> select upper('ljz');
+-------------------------+
|       upper('ljz')      |
+-------------------------|
|       LJZ               |
+-------------------------+
```

5.left（）和 right（）函数则是分别取字段左侧多少的字符和字段右侧多少的字符。

```
mysql> select left(lijinze,2);
+-------------------------+
|      left(lijinze,2)    |
+-------------------------|
|       li                |
+-------------------------+

mysql> select right(lijinze,3);
+-------------------------+
|      right(lijinze,3)   |
+-------------------------+
|          nze            |
+-------------------------+



// 同时此函数可以跟前面的lower&upper一起使用
mysql>  select upper(left('lijinze',3));
+-----------------------------------+
|      upper(left('lijinze',3))     |
+-----------------------------------+
|             LIJ                   |
+-----------------------------------+
```

2.字符串函数

1.length()是获取字符串长度的函数。

```
mysql> select LENGTH('LIJINZE');
+----------------------+
|   LENGTH('LIJINZE')  |
+----------------------+
|                     7|
+----------------------+
```

2.LTRIM()和RTRIM()函数分别是为删除字符串前的空格，字符串后面的空格，TRIM是吧前后的空格都删除。TRIM(LEADING是删除字符前面的符号，TRAILING是删除字符后面的的符号。BOTH是删除字符段两边的符号)。

```
mysql> select LTRIM(' LIJINZE');
+----------------------+
|    LTRIM(' LIJINZE')  |
+----------------------+
|      LIJINZE          |
+----------------------+

//运用LENGTH来判断//
mysql> select LENGTH(LTRIM(' LIJINZE'));
+---------------------------------+
|     LENGTH(LTRIM(' LIJINZE'))   |
+---------------------------------+
|                              7  |
+---------------------------------+

mysql> select RTRIM('LIJINZE ');
+---------------------------------+
|       RTRIM('LIJINZE ')         |
+---------------------------------+
|            LIJINZE              |
+---------------------------------+

mysql> select TRIM(' LIJINZE ');
+---------------------------------+
|         TRIM(' LIJINZE ')       |
+---------------------------------+
```

```
|             LIJINZE             |
+---------------------------------+

mysql> select TRIM(LEADING '/' FROM '///LIJINZE///');
+-----------------------------------------------------------+
|          TRIM(LEADINNG '/' FROM '///LIJINZE///');      |
+-----------------------------------------------------------+
|               LIJINZE///                                  |
+-----------------------------------------------------------+

mysql> select TRIM(TRAILING '/' FROM '///LIJINZE///');
+------------------------------------------------------------+
|           TRIM(TRAILING '/' FROM '///LIJINZE///');      |
+------------------------------------------------------------+
|                   ///LIJINZE                               |
+------------------------------------------------------------+


mysql> select TRIM(BOTH '/' FROM '///LIJINZE///');
+------------------------------------------------------------+
|             TRIM(BOTH'/' FROM '///LIJINZE///');         |
+------------------------------------------------------------+
|                     LIJINZE                                |
+------------------------------------------------------------+
```

3.replace函数（'截取所属的字符段'，'截取目标'，'替换的内容'）截取函数是用其他来替换自己想要换掉的字符段。

```
mysql> select REPLACE('LIJINZE','LIJIN','GOD');
+------------------------------------------------------------+
|             REPLACE('LIJIZNE','LIJIN','GOD');            |
+------------------------------------------------------------+
|                          GODZE                             |
+------------------------------------------------------------+
```

4.substring（） 格式：substring（'字符段'，截取字符段从哪一位开始'）

```
mysql> select SUBSTRING('LIJINZE',2);
+--------------------------------------------------------------+
|               SUBSTRING('LIJINZE',2)                       |
+--------------------------------------------------------------+
|                          IJINZE                            |
+--------------------------------------------------------------+
```

5.[NOT]LIKE 格式：LIKE('%_%'); 如果查询首位的字符则格式为__%，如果查询字符段中间的字符则为'%-%', 如果查询的字符在字符段的末尾则查询方式为'%-'，（特例：如果所需查询的字符包含%则格式为'%1%%',ESCAPE'1';)

```
mysql> select 'lijinze' LIKE '%ze'
+--------------------------------------------------------------+
|        'lijinze' LIKE '%ze'                                |
+--------------------------------------------------------------+
|                                                           1|
+--------------------------------------------------------------+
```

3.运算符与函数

1.ceil()函数进一取整。将小数点之后的所有数字化零取整进一位。

```
mysql> SELECT CEIL(3.0001);
+-----------------------------+
|         CEIL(3.0001)        |
+-----------------------------+
|                           4 |
+-----------------------------+
```

2.DIV()整数除法 只进行整数位的运算除法，舍去小数点之后的数字。

```
mysql> SELECT 10  DIV 3;
+-----------------------------+
|        10 DIV  3            |
+-----------------------------+
|                   3         |
+-----------------------------+
```

3.FLOOR()舍去法取整，将小数点后面的全部舍去只保留整数位 （没有四舍五入）

```
mysql> SELECT FLOOR(123.123);
+-----------------------------+
|        FLOOR(123.123)       |
+-----------------------------+
|                 123         |
+-----------------------------+
```

4.mod取余函数等于%

```
mysql> SELECT 10 MOD 3;
+-----------------------------+
|          10 MOD  3          |
+-----------------------------+
|                 1           |
+-----------------------------+
```

5.POWER()幂运算。

```
mysql> SELECT POWER(2,3);
+----------------------------------+
|            POWER(2,3)            |
+----------------------------------+
|                8                 |
+----------------------------------+
```

6.ROUND()四舍五入，当没有指定保留几位小数时 直接四舍五入到整数位，如果需要保存到特定的小数点后则需要给他进行指定。

```
mysql> SELECT ROUND(3.1415926);
+----------------------------------+
|           ROUND(3.1415926)       |
+----------------------------------+
|                 3                |
+----------------------------------+


mysql> SELECT ROUND(3.1415926,3);
+-----------------------------------+
|            ROUND(3.1415926,3)     |
+-----------------------------------+
|              3.142                |
+-----------------------------------+
```

7.TRUNCATE()截取函数 必须给出截取的位数此函数才能正常运行 如果截取的位数为正数则从小数点开始向小数点后开始截取所需的位数，如果截取的位数为负数，则从小数点开始向前进行截取相应的位数。

```
mysql> SELECT TRUNCATE(123.123,2);
+-----------------------------------+
|            TRUNCATE(123.123,2)    |
+-----------------------------------+
|               123.12              |
+-----------------------------------+


mysql> SELECT TRUNCATE(123.123,-2);
+------------------------------------+
|           TRUNCATE(123.123,-2)     |
+------------------------------------+
|               100                  |
+------------------------------------+


mysql> SELECT TRUNCATE(1234.12,-1);
+-----------------------------------+
|            TRUNCATE(1234.12,-1)   |
+-----------------------------------+
|                1230               |
+-----------------------------------+
```

3.比较运算符与函数

1.BETWEEN...AND.. 函数 表达 是否在这个范围内。

```
mysql> SELECT 123 BETWEEN 120 AND 200;
+--------------------------+
| 123 BETWEEN 120 AND 200 |
+--------------------------+
|                        1 |
+--------------------------+
1 row in set (0.00 sec)
```

2.IN()函数表示想要的值是否在这个值的范围内

```
mysql> SELECT 2 IN (1,2,3,4,5,6,7);
+----------------------+
| 2 IN (1,2,3,4,5,6,7) |
+----------------------+
|                    1 |
+----------------------+
1 row in set (0.00 sec)
```

3.is null 表示这个是否为空。

```
mysql> SELECT '0' IS NULL;
+-------------+
| '0' IS NULL |
+-------------+
|           0 |
+-------------+
1 row in set (0.00 sec)

mysql> SELECT '10' IS NULL;
+--------------+
| '10' IS NULL |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)
```

4.日期时间函数

1.now（）用来显示当前的日期和时间。

```
mysql> SELECT NOW();
+---------------------+
| NOW()               |
+---------------------+
| 2020-11-27 20:11:16 |
+---------------------+
1 row in set (0.00 sec)
```

2.curdate（） curtime（） 前者表示当前的日期 后者表示当前的时间。

```
mysql> SELECT CURDATE();
+-----------------------------+
|         CURDATE()           |
+-----------------------------+
|            2020-11-27       |
+-----------------------------+


mysql> SELECT CURTIME();
+-----------------------------+
|           CURTIME()         |
+-----------------------------+
|             21:07           |
+-----------------------------+
```

3.date_add() 表示日期的变化

```
mysql> SELECT DATE_ADD('2019-11-27',INTERVAL 365 DAY);
+-----------------------------------------+
| DATE_ADD('2019-11-27',INTERVAL 365 DAY) |
+-----------------------------------------+
| 2020-11-26                              |
+-----------------------------------------+
1 row in set (0.00 sec)
```

4.datediff表示日期差值

```
mysql> SELECT DATEDIFF('2020-11-27','2020-12-12');
+-------------------------------------+
| DATEDIFF('2020-11-27','2020-12-12') |
+-------------------------------------+
|                                 -15 |
+-------------------------------------+
1 row in set (0.00 sec)
```

5.date_format()日期格式化

```
ysql> SELECT DATE_FORMAT('2020-11-27','11/27/2020');
+---------------------------------------+
| DATE_FORMAT('2020-11-27','11/27/2020') |
+---------------------------------------+
| 11/27/2020                            |
+---------------------------------------+
 row in set (0.00 sec)

ysql> SELECT DATE_FORMAT('2020-11-27','%m%d%Y');
+-----------------------------------+
| DATE_FORMAT('2020-11-27','%m%d%Y') |
+-----------------------------------+
| 11272020                          |
+-----------------------------------+
 row in set (0.00 sec)

ysql> SELECT DATE_FORMAT('2020-11-27','%m/%d/%Y');
+-------------------------------------+
| DATE_FORMAT('2020-11-27','%m/%d/%Y') |
+-------------------------------------+
| 11/27/2020                          |
+-------------------------------------+
 row in set (0.00 sec)
```

5.信息函数

这几个信息函数里面最重要的就是第三个 last_insert_id();

```
mysql> select * from beaster;
+------+
| id   |
+------+
|    1 |
|    2 |
+------+
2 rows in set (0.00 sec)

mysql> select last_insert_id();
+------------------+
| last_insert_id() |
+------------------+
|                0 |
+------------------+
1 row in set (0.00 sec)
```

6.聚合函数


```
#ave 必须要配合表使用 select avg(num) from user;
#count与avg用法一样 select round(avg(num),2) from user;
#round与avg合用保留俩位小数
```

MAX（MIN）只能配合到分组里面使用，直接查询会报错，可通过关闭严格模式来查询,MIN用法一样

```
SET sql_mode='';
select username ,max(1) from user;#查询的username会不准确，我这里是因为其他用户名为空
+----------+---------+
| username | max(1)  |
+----------+---------+
| xixi     | 123.46  |
+----------+---------+
select * from user where aa=(select MAX(aa) from user);
+----------+------+---------+--------+------+---------+
| username | num  | lmonkey | 1      | sex  | content |
+----------+------+---------+--------+------+---------+
| xixi     | 127  | 10      | 123.46 | nv   | lmonkey |
+----------+------+---------+--------+------+---------+
#先找到最大值，再去查询信息就准确了
```

7.加密函数


```
select md5('123');
+----------------------------------+
| md5('123')                       |
+----------------------------------+
| 202cb962ac59075b964b07152d234b70 |
+----------------------------------+
```