10.MySQL高级开发上（内置函数应用及变量使用）√

1.内置函数的应用

1.1 概念

> 在开发称之为“方法”，将一组逻辑语句防撞在方法体中，对外暴露的方法名。进行调用方法名

1.2 作用

> 1.隐藏代码实现细节
> 2.提高代码的重用性（复用）

1.3 调用方法

> select 函数名(参数）[from 表]
> []中括号是可选项

1.4 关注点

> 1.函数名（常用的函数名称有哪些）
> 2.函数的功能

1.5 分类

查看函数种类方法 `mysql> help Functions;` 或者官方文档

> 单行函数，例如： concat()、length()等。参数只能单个或者表中单行数据
> 分组函数,例如：sum()、count()等。可一次性处理多行
> 其他函数：例如：now()等

1.6 单行函数

1.6.1 字符函数

Length(字符长度)

Bash | Copy

```
作用：获取字节量，收到字符集的影响
show variables like '%char%'; 查看字符集使用. mysql客户端字符集默认utf8
手动传参：
mysql> select length('仁义');
+------------------+
| length('仁义')   |
+------------------+
|                6 |
+------------------+
自动传参：查看表中类名的字节量长度
1.查看表中的列
mysql> desc a;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| id    | int         | YES  |     | NULL    |       |
| name  | varchar(20) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2.针对表中的列名显示字节量
mysql> select * from a limit 1;
+------+------+
| id   | name |
+------+------+
|    1 | zs   |
+------+------+
mysql> select length(name) from a limit 1;
+--------------+
| length(name) |
+--------------+
|            2 |
+--------------+
案例：  判断表种某列的字节,帮助我们确认数据类型是否正确,判断索引是否需要前缀索引。
1.查看表中name列最大的字节长度，来确定我们数据类型的使用，和是否要加前缀索引进行优化
mysql> select max(length(name)) from world.city ;
+--------------------+
| max(length(name)) |
+--------------------+
|                33 |
+--------------------+
```

Concat(拼接字符串)

Bash | Copy

```
作用：拼接字符串
案例一：查看mysql.user将用户名和白名单拼接显示
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| oldguo           | 10.0.0.%  |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
mysql> select concat(user,"@",host) from mysql.user;
+----------------------------+
| concat(user,"@",host)      |
+----------------------------+
| oldguo@10.0.0.%            |
| mysql.infoschema@localhost |
| mysql.session@localhost    |
| mysql.sys@localhost        |
| root@localhost             |
+----------------------------+
案例二：将所有业务库（排除系统库）表的存储引擎替换为innodb
0.查看业务库非innodb的表
mysql> select table_schema,table_name,engine from information_schema.tables
where engine<>'innodb' and table_schema not in ('sys','mysql','information_schema','performance_schema');
1.模拟创建一个myisam表
mysql> create table world.a (id int ) engine=myisam;
2.拼接
mysql> select concat("alter table ",table_schema,".",table_name," engine=innodb;") from information_schema
bles
where engine<>'innodb' and table_schema not in ('sys','mysql','information_schema','performance_schema');
+-----------------------------------------------------------------------+
| concat("alter table ",table_schema,".",table_name," engine=innodb;") |
+-----------------------------------------------------------------------+
| alter table world.a engine=innodb;                                    |
+-----------------------------------------------------------------------+
3.替换
mysql> alter table world.a engine=innodb;
案例三：单库单表方式备份业务库中表
mysql> select concat(" mysqldump -uroot -p123 ",table_schema," ",table_name," >/bak/",table_schema,"_",tab
name,".sql") as table_dump
from information_schema.tables where table_schema='world';
+-----------------------------------------------------------------------------+
| table_dump                                                                  |
+-----------------------------------------------------------------------------+
|  mysqldump -uroot -p123 world a >/bak/world_a.sql                           |
|  mysqldump -uroot -p123 world city >/bak/world_city.sql                     |
|  mysqldump -uroot -p123 world country >/bak/world_country.sql               |
|  mysqldump -uroot -p123 world countrylanguage >/bak/world_countrylanguage.sql |
```

```
+-----------------------------------------------------------------------------------+
```

upper&lower(大小写转化)

Bash | Copy

```
作用：转换大小写
案例：
upper 小写转化为大写
mysql> select upper('abc');
+--------------+
| upper('abc') |
+--------------+
| ABC          |
+--------------+
lower 大写转化为小写
mysql> select lower('CDE');
+--------------+
| lower('CDE') |
+--------------+
| cde          |
+--------------+
```

Substr(截取字符串)

Bash | Copy

```
作用：截取字符串
语法：substr（字符串，position,length）或者substring（字符串，position,length）
                    截取位置 截取长度
案例：
mysql> select substr('李莫愁爱上了陆展元',1,3); 第一个字符截取，截取3个
+-----------------------------------------+
| substr('李莫愁爱上了陆展元',1,3)          |
+-----------------------------------------+
| 李莫愁                                   |
+-----------------------------------------+

mysql> select substr('李莫愁爱上了陆展元',7); 第七个字符截取，截取到最后
+-----------------------------------------+
| substr('李莫愁爱上了陆展元',7)           |
+-----------------------------------------+
| 陆展元                                   |
+-----------------------------------------+
```

Instr（判断字符串的是否出现）

```
作用：返回子集首次次出现的下标索引位置。
语法：select insrt('查看的子集', '搜索的子集中什么信息');
mysql> select INSTR('abcabc','c');  abcabc子集中出现首次c的位置是3
+--------------------+
| INSTR('abcabc','c') |
+--------------------+
|                  3 |
+--------------------+
案例一：判断某个字符串是否在表中某行出现过
mysql> select id, instr(name,'qingdao') as a from world.city where countrycode='CHN' having a>0 ;
+------+---+
| id   | a |
+------+---+
| 1903 | 1 |
+------+---+
案列二：判断某个字符串在表中出现过的次数
mysql> select sum(instr(name,'qingdao')) as a from world.city where countrycode='CHN';
+------+
| a    |
+------+
|    1 |
+------+
```

Trim（掐头去尾字符串）

```
作用：掐头去尾字符串，默认去除空格，可以指定去除的字符
默认去除
mysql> select TRIM(' 柴仁义  ') AS test;
+-----------+
| test      |
+-----------+
| 柴仁义    |
+-----------+
指定字符去除
mysql> select TRIM('a' from 'aaa柴仁义aaa');
+-----------------------------------+
| TRIM('a' from 'aaa柴仁义aaa')    |
+-----------------------------------+
| 柴仁义                            |
+-----------------------------------+
```

Lpad&Rpad(左填充&右填充)

```
作用：Lpad(left pad)左填充 Rpad(right pad)右填充
语法：select lpad&rpad('字符串', '字符长度', '填充的字符');
列子：
左填充：
mysql> select lpad('柴仁义','10','*');
+-----------------------------+
| lpad('柴仁义','10','*')    |
+-----------------------------+
| *******柴仁义               |
+-----------------------------+
右填充：
mysql> select rpad('柴仁义','10','*');
+-----------------------------+
| rpad('柴仁义','10','*')    |
+-----------------------------+
| 柴仁义*******               |
+-----------------------------+
```

Replace（替换）

```
作用：将原字符替换成目标字符
语法：select replace('字符串','被替换的字符','替换的目标字符');
例子
mysql> select REPLACE('谢霆锋***张柏芝','张柏芝','王菲') ;
+-----------------------------------------------------+
| REPLACE('谢霆锋***张柏芝','张柏芝','王菲')            |
+-----------------------------------------------------+
| 谢霆锋***王菲                                         |
+-----------------------------------------------------+
```

1.6.2 数学函数

加减乘除

Bash | Copy

```
加
mysql> select 1+1
    -> ;
+-----+
| 1+1 |
+-----+
|   2 |
+-----+

减
mysql> select 2-1;
+-----+
| 2-1 |
+-----+
|   1 |
+-----+

乘
mysql> select 2*2;
+-----+
| 2*2 |
+-----+
|   4 |
+-----+

除
mysql> select 4/2;
+--------+
| 4/2    |
+--------+
| 2.0000 |
+--------+
```

round（四舍五入）

```
作用： 四舍五入，逢五进一
语法： select round(数字,保留小数后几位);
例子：
1.默认保留到整数位
mysql> select ROUND(3.1415);
+---------------+
| ROUND(3.1415) |
+---------------+
|             3 |
+---------------+
2.保留小数后3位
mysql> select ROUND(3.1415,3);
+-----------------+
| ROUND(3.1415,3) |
+-----------------+
|           3.142 |
+-----------------+
```

ceil （向上取整数）

```
作用：向上取整数
语法：向上取 >= 最小整数
列子1
mysql> select CEIL(3.14);
+------------+
| CEIL(3.14) |
+------------+
|          4 |
+------------+
列子2
mysql> select CEIL(3.00);
+------------+
| CEIL(3.00) |
+------------+
|          3 |
+------------+
例子3
mysql> select CEIL(-3.14);
+-------------+
| CEIL(-3.14) |
+-------------+
|          -3 |
+-------------+
```

floor（向下取整）

```
作用：向下取整数
语法：向下 >= 最大整数
列子1
mysql> select floor(9.99);
+-------------+
| floor(9.99) |
+-------------+
|           9 |
+-------------+
列子2
mysql> select FLOOR(9.00);
+-------------+
| FLOOR(9.00) |
+-------------+
|           9 |
+-------------+
```

truncate（保留整数的小数位数）

```
作用：截断小数点，保留小数点多少位
语法：select truncate(带小数的数字，保留小数多少位)；
mysql> select TRUNCATE(3.15,0);
+------------------+
| TRUNCATE(3.15,0) |
+------------------+
|                3 |
+------------------+
```

mod（取模）

```
作用：取模
算法：select mod(a,b); a为正数则结果为正，a为负数则结果为负
a为正数
mysql> select MOD(10,3);
+-----------+
| MOD(10,3) |
+-----------+
|         1 |
+-----------+

a为负数
mysql> select MOD(-10,3);
+------------+
| MOD(-10,3) |
+------------+
|         -1 |
+------------+
```

rand（随机数）

```
作用：生成某个范围内的随机数
语法：select rand()*10 随机生成默认范围0到9随机数
                    *几（随机范围取值）
rand结合使用实列
1.随机生成0到9之间的整数
floor+rand
mysql> select floor(rand()*10);
2.随机生成1到9之间的整数
ceil+rand=1+floor+rand
mysql> select ceil(rand()*10); 等价于 mysql> select 1+floor(rand()*10);
```

进制换算

```
ascii(str)
bin(n) 二进制转化
oct(n)
hex(n)
```

1.6.3 函数综合案例

Bash | Copy

```
案例一：生成随机密码，随机密码格式要求总共12个，开头是大写字母后面跟上11个数字字母组合
1.随机生成开头大写字母
select substr('ABCDEFGHIJKLMNOPQRSTUVWXYZ',1+floor(rand()*26),1) as test;
       截取函数 截取字符集                    截取位置          截取长度
mysql> select substr('ABCDEFGHIJKLMNOPQRSTUVWXYZ',1+floor(rand()*26),1) as test;
+------+
| test |
+------+
| J    |
+------+
2.随机生成11数字组合，我们可以借用select uuid()随机生成
2.1先进行uuid随机生成的替换
mysql> select replace(uuid(),'-','');
+----------------------------------+
| replace(uuid(),'-','')           |
+----------------------------------+
| d6f245f2986411eb8beb000c29ef43a9 |
+----------------------------------+
2.2 然后截取替换后的uuid结果11个字符。
mysql> select substr(replace(uuid(),'-',''),1+floor(rand()*21),11);
+-------------------------------------------------+
| substr(replace(uuid(),'-',''),1+floor(rand()*21),11) |
+-------------------------------------------------+
| 0986511eb8b                                     |
+-------------------------------------------------+
3.最后将两部分拼接在一起生成我们需要的随机密码
mysql> select concat(substr('ABCDEFGHIJKLMNOPQRSTUVWXYZ',1+floor(rand()*26),1)
,substr(replace(uuid(),'-',''),1+floor(rand()*21),11)) as '随机密码';
+--------------+
| 随机密码     |
+--------------+
| B1eb8beb000c |
+--------------+
案例二：随机生成ip地址
SELECT CONCAT( FLOOR(RAND()*255),".", FLOOR(RAND()*256),".", FLOOR(RAND()*256),".", FLOOR(RAND()*256) ) AS IP
```

1.6.4 日期函数

可直接调用的日期函数

Bash | Copy

```
1.now()
作用：查看当前时间
mysql> select now();
+---------------------+
| now()               |
+---------------------+
| 2021-04-08 20:41:55 |
+---------------------+

2.curdate()
作用：查看当前日期（年月日）
mysql> select curdate();
+------------+
| curdate()  |
+------------+
| 2021-04-08 |
+------------+

3.curtime()
作用：查看当前时间（时分秒）
mysql> select curtime();
+-----------+
| curtime() |
+-----------+
| 20:42:23  |
+-----------+

```

截取时间字段的日期函数

Bash | Copy

```
1.month()
作用：截取当前时间段的月份，以数字的形式显示
mysql> select month(now());
+--------------+
| month(now()) |
+--------------+
|            4 |
+--------------+

2.monthname()
作用：截取当前时间段的月份，以英文月份的形式显示
mysql> select monthname(now());
+------------------+
| monthname(now()) |
+------------------+
| April            |
+------------------+

3.year()
作用：截取当前时间段的年份，以数字的形式显示
mysql> select year(now());
+-------------+
| year(now()) |
+-------------+
|        2021 |
+-------------+


4.day()
作用：截取当前时间段的日期，以数字的形式显示
mysql> select day(now());
+------------+
| day(now()) |
+------------+
|          8 |
+------------+

5.hour()
作用：截取当前时间段的小时，以数字的形式显示
mysql> select hour(now());
+-------------+
| hour(now()) |
+-------------+
|          20 |
+-------------+

6.minute()
作用：截取当前时间段的分钟，以数字的形式显示
mysql> select minute(now());
+---------------+
```

```
| minute(now()) |
+---------------+
|            49 |
+---------------+

7.second()
作用：截取当前时间段的秒，以数字的形式显示
mysql> select second(now());
+---------------+
| second(now()) |
+---------------+
|            19 |
+---------------+

8.UNIX_TIMESTAMP()
作用：
8.1 截取当前时间段的时间戳，以数字的形式显示。
8.2 将日期转化为时间戳
mysql> select UNIX_TIMESTAMP(now());
+-----------------------+
| UNIX_TIMESTAMP(now()) |
+-----------------------+
|            1617886364 |
+-----------------------+

9.FROM_UNIXTIME()
作用：将时间戳转化为日期
mysql> select FROM_UNIXTIME(1617886364);
+---------------------------+
| FROM_UNIXTIME(1617886364) |
+---------------------------+
| 2021-04-08 20:52:44       |
+---------------------------+

案例一：计算人的年龄
mysql> select (year(NOW())-year('1998-07-24 '));
+-----------------------------------+
| (year(NOW())-year('1998-07-24 ')) |
+-----------------------------------+
|                                23 |
+-----------------------------------+

```

STR_TO_DATE()

Bash | Copy

```
作用：指定日期的格式,使mysql可以识别。
把不规则日期格式转化为mysql标准的日期格式
列子一：
mysql> select STR_TO_DATE('5-3 2020','%m-%d %Y') as 'MySQL标准格式';
+--------------------+
| MySQL标准格式      |
+--------------------+
| 2020-05-03         |
+--------------------+
```

格式符	作用
%Y	4位年份，例如：1998
%y	2位年份，例如：98
%m	月份，例如： 01，02,...,12
%c	月份，例如： 1，2,...,12
%d	日期，例如01，02,..,31
%H	24小时制
%h	12小时制
%i	分钟，例如：00-59
%s	秒,例如：00-59

DATE_FORMAT()

```
作用：将日期格式转化为字符串格式
mysql> select DATE_FORMAT(NOW(),'%Y-%m-%d');
+--------------------------------+
| DATE_FORMAT(NOW(),'%Y-%m-%d') |
+--------------------------------+
| 2021-04-08                     |
+--------------------------------+
```

1.6.5 流程控制函数

if函数 （if else）

```
双分支判断条件
mysql> select if(2>1,'yes','no');
+--------------------+
| if(2>1,'yes','no') |
+--------------------+
| yes                |
+--------------------+
案例一：
mysql> select user,if(user='root',"管理员","普通用户") from mysql.user;
+-----------------+-------------------------------------------+
| user            | if(user='root',"管理员","普通用户")        |
+-----------------+-------------------------------------------+
| oldguo          | 普通用户                                   |
| mysql.infoschema | 普通用户                                  |
| mysql.session   | 普通用户                                   |
| mysql.sys       | 普通用户                                   |
| root            | 管理员                                     |
+-----------------+-------------------------------------------+
```

case函数

Bash | Copy

```
用法1: 等值判断
case 表达式
when 等值判断 then 值1
...
else 值N
end
例子:
0.首先创建环境表tt
mysql> create table tt(id int,num int);
mysql> insert into tt values(1,110),(2,119),(3,120);
1.case判断
select
case num
when 110 then CONCAT(num,':抓小偷')
when 119 then CONCAT(num,':救火')
else CONCAT(num,':救人')
end as test from tt;
+---------------+
| test          |
+---------------+
| 110:抓小偷    |
| 119:救火      |
| 120:救人      |
+---------------+
用法2: 范围判断
case
when 条件1 then 结果或语句
when 条件2 then 结果或语句
else
end
列子
统计每门课程:优秀(85分以上),良好(70-85),一般(60-70),不及格(小于60)的学生列表
SELECT course.cname,
GROUP_CONCAT(CASE WHEN sc.score>=85 THEN student.sname END) AS '优秀',
GROUP_CONCAT(CASE WHEN sc.score>=70 AND sc.score<85 THEN student.sname END) AS '良好',
GROUP_CONCAT(CASE WHEN sc.score>=60 AND sc.score<70  THEN student.sname END) AS '一般',
GROUP_CONCAT(CASE WHEN sc.score<60 THEN student.sname END) AS '不及格'
FROM student
JOIN sc
ON student.sno=sc.sno
JOIN course
ON sc.`cno`=course.cno
GROUP BY course.cno
+--------+------------------+--------------------+--------+-----------+
| cname  | 优秀             | 良好               | 一般   | 不及格    |
+--------+------------------+--------------------+--------+-----------+
| linux  | li4,zhao4        | zhang3,wang5,oldboy | ma6    | NULL      |
| python | zhang4           | NULL               | wang5  | zhang3    |
| mysql  | oldp,wang5,zhang4 | oldgirl,ma6,zhao4  | NULL   | zh4,li4   |
+--------+------------------+--------------------+--------+-----------+
```

1.6.6 其他函数

Bash | Copy

```
1.version()
作用：查看当前数据库版本
mysql> select version();
+-----------+
| version() |
+-----------+
| 8.0.20    |
+-----------+

2.database()
作用：查看当前使用（use到的库）的库
mysql> select database();
+------------+
| database() |
+------------+
| world      |
+------------+

3.user()
作用：查看当前数据库登陆用户
mysql> select user();
+----------------+
| user()         |
+----------------+
| root@localhost |
+----------------+

4.uuid()
作用：查看数据库系统随机生成的uuid号码
mysql> select uuid();
+--------------------------------------+
| uuid()                               |
+--------------------------------------+
| 6bc5c1fb-986c-11eb-8beb-000c29ef43a9 |
+--------------------------------------+
```

2.变量的使用

`mysql>` `show variables;` 没有指定是输出全局变量还是会话变量的话，默认就输出会话变量

2.1 系统变量

2.1.1 全局变量

对全局变量的修改会影响到整个服务器

> 全局变量在MYSQL启动的时候由服务器自动将它们初始化为默认值，这些默认值可以通过更改my.ini这个文件来更改。

查看全局变量

> 查看所有全局变量
> mysql> show global variables;
> 模糊查询全局变量的单个变量（常用）
> mysql>show global variables like '%不完整变量名称%'

设置全局变量

```
mysql> set global read_only=1;    对普通用户设置只读 mysql> set global supper_read_only=1 对数据库root用户设置只读
退出会话就可以生效了
mysql> select @@global.read_only;
+--------------------+
| @@global.read_only |
+--------------------+
|                  1 |
+--------------------+
```

2.1.2 会话变量

对会话变量的修改，只会影响到当前的会话（也就是当前的数据库连接）

> 会话变量在每次建立一个新的连接的时候，由MYSQL来初始化。MYSQL会将当前所有全局变量的值复制一份。来做为会话变量。

查看会话变量

> mysql> show session variables;
> 模糊查询会话变量的单个变量
> mysql>show session variables like '%不完整变量名称%'

设值会话变量

```
set session read_only=1;
select @@session.read_only;
```

动态参数和静态参数的区别

可查看官方文档里面对参数的介绍

> 动态：Dynamic=yes 可以在线修改，立即生效
> 静态：Dynamic=no 修好后需重启数据库生效（大多数对目录的修改）

2.2 用户变量（自定义变量）

> 1.可以先在用户变量中保存值然后在以后引用它；这样可以将值从一个语句传递到另一个语句
> 2.用户变量与连接有关，一个客户端定义的变量不能被其它客户端看到或使用。当客户端退出时，该客户端连接的所有变量将自动释放。

2.2.1使用场景

> 针对会话有效，会话任意位置使用。单独设置或者在存储过程函数都可。

2.2.2用户变量的使用

Bash | Copy

```
赋值方式（只能单一赋值）
方式一：mysql> set @var1:=值;
        mysql> select @var1:=值;
方式二：mysql> select count(*)  from world.city into @count;  将sql的结果集赋值给一个用户变量
调用
mysql> select @用户变量名称;
```

2.2.3用户变量案例

Bash | Copy

```
案例一：
0.题目要求我们对这张表的num列进行递归增加
mysql> select * from t1;
+------+------+
| id   | num  |
+------+------+
|    1 | 10   |
|    2 | 25   |
|    3 | 12   |
|    4 | 8    |
+------+------+
1.我们设置用户变量。
mysql> set @sum:=0;
2.再进行调用用户变量。 每当查询一行就会进行递归增加num列的信息
mysql> select id,@sum:=(@sum + num) as s from t1;
+------+------+
| id   | num  |
+------+------+
|    1 | 10   |
|    2 | 35   |
|    3 | 47   |
|    4 | 55   |
+------+------+
案例二：生成随机密码
0. 将uuid中的横杠替换成空，再赋值给str
mysql> select replace(uuid(),'-','') into @str;

0.查看赋值的str
mysql> select @str;
+----------------------------------+
| @str                             |
+----------------------------------+
| 0c0389a199d911eba190000c29ef43a9 |
+----------------------------------+
1.调用str变量截取并下取整11个字符，再将结果赋值给str1
mysql> select substring(@str,floor(rand()*21+1),11) into @str1;
```

2.3 局部变量（自定义变量）

2.3.1使用场景

> 必须在存储过程内部使用，即:begin ....end存储过程流程中

声明

```
DECLARE 变量名 类型;
DECLARE 变量名 类型 DEFAULT 值;
```

赋值

```
方式1:
set var=值;
set var1:=值;
select @var2:=值;

方式2：select into
select count(*) from world.city   into count
```

调用

```
select count;
```

ded61a71ecee.png&title=10.MySQL%E9%AB%98%E7%BA%A7%E5%BC%80%E5%8F%91%E4%B8%8A%EF%BC%88%E5%86%85%E7%BD%AE%E5%87%BD%E6%95%B0%E5%BA%94%E7%94%A8%E5%8F