一　mysql视图

１，基础

－－什么是视图

虚拟表

内容与真实的表相似，有字段有记录

视图并不再数据库中以存储的数据形式存在

行和列的数据来自定义视图时查询所引用的基表，并且在具体引用视图时动态生成

更新视图的数据，就是更新基表的数据

更新基表的数据，视图的数据也会跟着改变

－－视图优点

简单

　　用户不需要关系视图中的数据如何查询获得，视图中的数据已经是过滤好的符合条件的结果集

安全

　　用户只能看到视图中的数据

数据独立

一旦视图结构确定，可以屏蔽表结构对用户的影响

－－视图使用限制

不能在视图上创建索引

在视图的from子句中不能使用子查询

以下情形中的视图是不可更新的

包含以下关键字的sql语句：聚合函数(sum,min,max,avg,count等),distinct,group by,having,union或union all

长量视图，join,from一个不能更新的视图

Where子句的子查询引用了from子句中的表

使用了临时表

２，命令

－－创建视图

Create view 视图名称　as sql查询;

Create view 视图名称(字段列表)　as sql查询;

提示：在视图表中不定义字段名的话，默认使用基表的字段名，若定义字段名的话，视图表的字段名必须和基表的字段个数相等

－－查看视图

Show table status;

Show table status where comment=’view’\G;

Show create view 视图名;

－－使用视图

Select 字段列表　from 视图名　where 条件;

Insert into 视图名(字段名列表)　values(字段值列表);

Update 视图名 set 字段名＝值 .. Where 条件;

Delete from 视图名 where 条件;

提示：

(1)对视图操作即是对基本表操作，反之亦然

(2)结合2张表的view 无法update

(3)如果想让view 可以更新，建view 语句需要满足以下条件

没有 表结合

没有 集合演算 如 UNION ALL / INTERSECT / MINUS

没有 GROUP BY

没有 集合函数

没有 DISTINCT

3，视图进阶

Merge:替换方式（默认）

Remptable:具体化方式

二　mysql存储过程

编译执行

１，前言

－－存储过程介绍

存储过程，相当于是mysql语句组成的脚本

－－存储过程优点

提高性能

可减轻网络负担

２，基本使用

－－创建存储过程

语法格式：

Delimiter　 // 修改命令行结束符为//

Create procedure 名称()

Begin

..功能代码

End

//

Delimiter ;

提示：存储过程是在库里执行

例子：

mysql> delimiter //

mysql> create procedure p1()

-> begin

-> select \* from db9.user limit 3;

-> create table if not exists db9.a(id int);

-> end

-> //

mysql> delimiter ;

－－调用／删除存储过程

格式：

调用

Call 存储过程名(); #存储过程没有参数时，()可以省略

删除

Drop procedure 存储过程名;　　#

例子：

mysql> call p1();

+----+--------+--------+------+------+---------+-------+---------------+

| id | name | passwd | uid | gid | content | home | shell |

+----+--------+--------+------+------+---------+-------+---------------+

| 1 | moot | x | 0 | 0 | root | /root | /bin/bash |

| 2 | bin | x | 1 | 1 | bin | /bin | /sbin/nologin |

| 3 | daemon | x | 2 | 2 | daemon | /sbin | /sbin/nologin |

mysql> drop procedure p1;

－－查看

格式：

Show procedure status;

Select db,name,type from mysql.proc where name=”存储过程名”;

例子：

mysql> show procedure status\G;

mysql> select db,name,body,type from mysql.proc where name="p1"\G;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

db: db9

name: p1

body: begin

select \* from db9.user limit 3;

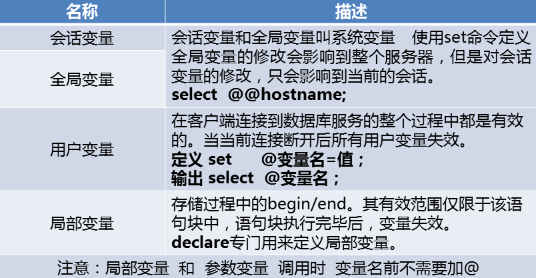
create table if not exists db9.a(id int);

end

type: PROCEDURE

3，存储过程进阶

－－变量类型



例子：

##会话变量

mysql> show session variables like '%buffer%';　#查看会话变量

##全局变量

mysql> show global variables like '%password%';　#查看全局变量

mysql> set global validate\_password\_length=6;　#设置全局变量值

##用户变量

mysql> set @x=7;

mysql> select @x;

+------+

| @x |

+------+

| 7 |

mysql> select name into @x from db9.user where id=1;　#查询结果赋值给x

mysql> select @x;

+------+

| @x |

+------+

| moot |

##局部变量

mysql> delimiter //

mysql> create procedure say()

-> begin

-> declare x int default 9;

-> declare y int;

-> declare n int;

-> set y=1;

-> select x;

-> select y;

-> select count(id) into n from db9.user;

-> end

-> //

mysql> delimiter ;

mysql> call say();

| x |

+------+

| 9 |

+------+

+------+

| y |

+------+

| 1 |

+------+

| n |

+------+

| 10 |

+------+

－－参数类型

调用参数时，名称前也不需要加@

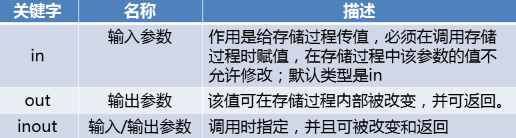
格式：

Create procedure 名称(

类型　参数名 数据类型,

类型　参数名 数据类型

)



例子：

##in

mysql> set @shell='/bin/bash';

mysql> delimiter //

mysql> create procedure jluo(in shellname char(30))

-> begin

-> declare x int default 0;

-> select count(name) into x from db9.user where shell=shellname;

-> select x;

-> end

-> //

mysql> delimiter ;

mysql> call jluo(@shell);

+------+

| x |

+------+

| 2

##out

mysql> delimiter //

mysql> create procedure db9.p7(in shellname char(30),out number int)

-> begin

-> select count(1) into number from db9.user where shell=shellname;

-> select number;

-> end

-> //

mysql> delimiter ;

mysql> set @x=7;

mysql> call p7("/bin/bash",@x);

##inout

mysql> delimiter //

mysql> create procedure db9.p8(inout x char(30))

-> begin

-> select x;

-> select name into x from db9.user where id=1;

-> select x;

-> end

-> //

mysql> delimiter ;

mysql> set @x='bb';

mysql> call p8(@x);

| x |

+------+

| bb |

| x |

+------+

| moot |

－－算数运算



例子：

mysql> delimiter //

mysql> create procedure db9.p10(in bash char(30),in nologin char(30))

-> begin

-> declare x int default 0;

-> declare y int default 0;

-> declare z int default 0;

-> select count(name) into x from db9.user where shell=bash;

-> select count(name) into y from db9.user where shell=nologin;

-> set z=x+y;

-> select z;

-> end

-> //

mysql> delimiter ;

mysql> call p10("/bin/bash","/sbin/nologin");

+------+

| z |

+------+

| 37 |

－－条件测试

数值比较



例子：

逻辑比较



－－选择结构

单分支：

If 条件测试　then

代码....

.....

End if;

多分支：

If 条件测试 then

代码１....

......

Else

代码２...

......

End if;

例子：

mysql> delimiter //

mysql> create procedure db9.p13(in linenum int)

-> begin

-> if 1=linenum then

-> select \* from db9.user where id=1;

-> else

-> select \* from db9.user where id=2;

-> end if;

-> end

-> //

mysql> delimiter ;

mysql> call p13(1);

+----+------+--------+------+------+---------+-------+-----------+

| id | name | passwd | uid | gid | content | home | shell |

+----+------+--------+------+------+---------+-------+-----------+

| 1 | moot | x | 0 | 0 | root | /root | /bin/bash |

－－循环结构

While 条件判断　do

循环体

　.......

End while;

死循环：loop

Loop

循环体

　....

End loop;

Repeat

循环体

　.....

Until 条件判断 #条件成立结束循环

End repeat;

控制循环的执行

Leave 标签名　　跳出循环

Iterate 标签名　　跳过本次循环，执行下一次循环

例子：

##while

mysql> delimiter //

mysql> create procedure db9.p16()

-> begin

-> declare i int default 1;

-> while i<= 5 do

-> select \* from db9.user where id=1;

-> set i=i+1;

-> end while;

-> end

-> //

mysql> delimiter ;

mysql> call p16();

##repeat

mysql> delimiter //

mysql> create procedure p18()

-> begin

-> declare i int default 1;

-> repeat

-> select \* from db9.user where id=1;

-> set i=i+1;

-> until i>3

-> end repeat;

-> end

-> //

mysql> delimiter ;

mysql> call p18();

##控制循环

mysql> delimiter //

mysql> create procedure p20()

-> begin

-> declare x int default 1;

-> label1:while x<=10 do

-> if x=3 then

-> select x;

-> leave label1;

-> end if;

-> set x=x+1;

-> end while;

-> end

-> //

mysql> delimiter ;

综合练习：

mysql> delimiter //

mysql> create procedure db9.sumuser()

-> begin

-> declare x int default 1;

-> declare j int default 0;

-> declare i int default 0;

-> declare sumline int;

-> select count(id) into sumline from db9.user;

-> while x<=sumline do

-> select uid into j from db9.user where id=x;

-> if j%2=0 then

-> set i=i+1;

-> end if;

-> set x=x+1;

-> end while;

-> select i;

-> end

-> //

mysql> delimiter ;

mysql> call sumuser();

字符串拼接

Set d\_tab=CONCAT(“drop table db5.”,t\_name);

Set @k=d\_tab;

Prepare stmt from @k;

Execute stmt;

Deallocate prepare stmt

删表：

mysql> delimiter //

mysql> create procedure droptab(in nametab char(10))

-> begin

-> set @k=CONCAT("drop table db5.",nametab);

-> prepare stmt from @k;

-> execute stmt;

-> deallocate prepare stmt;

-> end

-> //

mysql> delimiter ;

mysql> call droptab("t1");