code blocks in mysql

Why we use PLSQL

1. security
2. speed
3. to reduce traffic
4. to hide complexity

3 types of code blocks

1. procedure -------- this is block of code which does not return any value

no return statement

procedures cannot be used in select statement

1. function

we can call function in select statement

returns single value

1. trigger

the block which gets called implicitly, automatically after or before some DML operation happens on the table

how we write Procedure

1. write a procedure to insert data into a table

#change the delimiter to //

/\* comment \*/

--- comment

delimiter //

create procedure insertDeptRec(pdno int,pdnm varchar(20),pdloc varchar(20))

begin

insert into dept values(pdno,pdnm,pdloc);

end//

delimiter ;

call insertDeptRec(17,’HR’,Mumbai’);

#to see the list of procedure within database

show procedure status where db='iacsdedacmay21';

within a procedure you can pass parameters of 3 types

1. in ------- when you want to pass data as i/p to a procedure

these are readonly parameter

by default the parameters are in type

1. out ------ when you want to get o/p from a procedure then you can use out type parameter

these are write only parameter

1. inout ------ these parameters can be used to pass data as i/p and also get the o/p

these are read write parameters

we can modify the value of these parameters and we can also pass the i/p to the procedure

----write a procedure to find number of employees in dept 10

delimiter //

create procedure getempcnt(in pdno int,out pcnt int,out pmin decimal(9,2))

begin

select count(\*),min(sal) into pcnt,pmin

from emp

where deptno=pdno;

end//

delimiter ;

call getempcnt(10,@cnt,@m);

select @cnt,@m

----to pass count to a procedure and increase the count by 10

delimiter //

create procedure increasecnt(inout pcnt int)

begin

set pcnt=pcnt+10;

end//

set @cnt=12

call increasecnt(@cnt)

select @cnt

-------to find ename,job,deptno,sal of a employee whose id is given

delimiter //

create procedure getemp(in peid int,out pename varchar(20),out pjob varchar(20),out psal decimal(9,2),out pdno int)

begin

select ename,job,sal,deptno into pename,pjob,psal,pdno

from emp

where empno=peid;

end//

delimiter ;

call getemp(7902,@enm,@j,@s,@dno)

select @enm,@j,@s,@dno;

rules for select ……into statement

1. this can be used inside procedure or function
2. select ……into this query should return only one row.
3. number of columns in select statement and number of variable after into should be same

if else

loops

cursors

syntax of if

if condition then

statements

else

statements

end if

or if ….elseif…else

if condition then

statements

elseif condition then

statemenst

elseif condition then

statemenst

else

statements

end if

-----to write a procedure to calculate bonus

sal< 2000 bonus=sal\*10%+comm

otherwise bonus=sal\*15%+comm

delimiter //

create procedure calcBonus(pnm varchar(20))

begin

declare vsal, vcomm, vbonus int;

declare vjob varchar(20);

select sal,comm into vsal,vcomm

from emp

where ename=pnm;

if vsal<2000 then

set vbonus=vsal\*0.10+ifnull(vcomm,0);

else

set vbonus=vsal\*0.15+ifnull(vcomm,0);

end if;

select pnm,vsal,vcomm,vbonus;

end//

delimiter ;

call calBonus(‘WARD’);

scope of variable

1. local variables--- the variables declared inside procedure are call as local variables and can be used only inside procedure
2. session variables------- these are variable whose name starts with @and are called as session variables

these variables are accessible till you logout.

Loops in PLSQL

1. while ------ top tested loop

while condition do

statements

end while;

1. Repeat loop is called as bottom tested.

Repeat

statements

until condition

end repeat;

loop

statement

if condtion then

leave label

end if

end loop

leave ---🡪 break statement in java

iterate--🡪 continue

------ to test while loop

1,2,3,4,5

delimiter //

create procedure test\_while()

begin

declare i int default 0;

declare data varchar(20);

set data=’’;

while i<=5 do

set data=concat(data,i,’,’)

set i=i+1;

end while

select data;

end//

-------to declare variable

declare x int default 0;

declare y int default 20;

declare y int

declare x,y int;

declare vd date;

declare vnm varchar(2);

------Repeat until

1. bottom tested loop
2. it gets executed minimum once
3. it gets executed until given condition is false,as soon as condition becomes true it terminates the loop

------ to print 1,2,3,4,5

delimiter //

create procedure test\_repeat()

begin

declare i int default 1;

declare data varchar(20);

set data=’’;

Repeat

set data=concat(data,i,’,’);

set i=i+1;

until i>5

end repeat;

select data;

end //

-------test loop

----leave is same as break

-----iterate it is same as continue

label1:loop

statements

if condition then

iterate label1;

else

leave label1;

end if

end loop

------ to print 1,2,3,4,5,

delimiter //

create procedure test\_loop()

begin

declare i int default 1;

declare data varchar(20) default ‘’;

xyz:loop

if i>5 then

leave xyz;

end if;

set data=concat(data,i,’,’);

set i=i+1;

end loop;

select data;

end//

---------- write a procedure to display all numbers between 1 to 20 which are divisible by 5

5,10,15,20,

loop ------

if x mod 5==0 then

create procedure test\_loop1()

begin

declare i int default 1;

declare data varchar(20) default ‘’;

label1:loop

if i>20 then

leave label1;

end if;

if i mod 5=0 then

set data=concat(data,i,’,’);

end if;

set i=i+1;

end loop;

select data

end//

or

create procedure test\_loop1()

begin

declare i int default 1;

declare data varchar(20) default ‘’;

label1:loop

if i>20 then

leave label1;

end if;

set i=i+1;

if i mod 5!=0 then

iterate label1;

end if;

set data=concat(data,i,’,’);

end loop;

select data

end//

--------write a procedure to calculate experience of an employee whose name is given

delimiter //

create procedure calexp(in pname varchar(20),out pexp int,out pdt date)

begin

select hiredate into pdt

from emp

where ename=pname;

set pexp=floor(datediff(hiredate,curdate())/365);

select pexp;

end//

delimiter ;

call calexp(‘SMITH’,@e,@dt);

select @e,@dt;