PLSQL ----procedural SQL

if…..esle

loops

create procedure getData(psal int)

select \*

from mgr10

where sal>psal

getData(3000)

create procedure getproc(psal int,pnm varchar(20))

select \*

from mgr10

where sal>psal and name=pnm

getProc(2000,’SMSITH);

Why we write PLSQL

1.increases security----- since we wrap the query in procedure so the table name is hidden. so it increases the security

2. reduce network traffic

-----instaed of passing multiple queries from front end to backend for one transaction. we may wrap all those queries in procedure and call that procedure.

so frontend will send only one call to procedure all queries will get executed on database server and o/p of transaction will be send to front end

3. to hide complexity

if query is very complex the instead of writing the query multiple times we can store it inside procedure.

Type of blocks in PLS SQL

1. procedure
2. function
3. trigger

procedure --- block which executes code but don’t return anything

function ----- block which returns one value

triggers----- are special blocks which gets executed automatically

-------write a procedure to find how many employees are there in emp

while writing procedure you may pass data to a procedure and you may get output from the procedure

parameters are 3 types

in ---- use in type of parameters to pass data as i/p

these are read only parameters

and if user don’t specify any type then these are in type of parameters.

out ---- use in type of parameters to get data as o/p

these are write only parameters

inout------ ---- use in type of parameters to pass data as i/p and get the data as o/p

these are read-write parameters

--- to get count of employees who work in the given department

delimiter //

create procedure myproc(in pdeptno int , out pcnt int)

begin

set pcnt=0;

select count(\*) into pcnt

from emp

where deptno=pdeptno;

end//

delimiter ;

call myproc(10,@cnt);

select @cnt

-------- While passing parameter if you do not specify type then by default it is in parameter

create procedure myproc(pdeptno int)

-> begin

-> select count(\*)

-> from emp

-> where deptno=pdeptno;

-> end//

-

----- find empno,name,sal,deptno,job for employee by name

Rules for Select into statement :

1. select….., into clause should return only one row.
2. the number of columns select statement is returning should be same as number of variables after into keyword
3. select ….. into statement can be used only in PLSQL

delimiter //

create produre getEmp(in pname varchar(20),out peno int,out psal decimal(9,2),out deptno int,out pjob varchar(20))

begin

select empno,sal,deptno,job into peno,psal,pdeptno,pjob

from emp

where empname=’pename’;

end;

select \* from dept;

delimiter ;

-------write a procedure to calculate bonus of a employee based on name

if sal <2000

bonus=sal+comm\*10%

else

bonus=sal+comm\*15%

delimiter //

create procedure calcBonus(in pname varchar(20))

begin

declare vcomm,vsal,vbonus int;

select sal,comm into vsal,vcomm

from emp

where ename=pname;

if vsal<2000 then

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

else

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

end if;

select pname,vsal,vcomm,vbonus;

end//

call calBonus(‘SMITH’)

syntax of If statement in PLSQL using mysql

every statement in PLSQL has to end with ;

ELSE portion is optional

Simple If

IF condition then

statements

ELSE (condition) then

statements

END IF;

NESTED IF

IF condition then

statements

ELSEIF (condition) then

statements

ELSEIF (condition) then

statements

ELSE

statements

END IF;

How to assign value to a variable

1. using default

declare <variable name> <data type> default <initial value>

example

declare x int default 0;

declare x int;

set x=10;

declare vsal int;

select sal into vsal

from emp

where ename=pename;

scope of variable

there are variables which we declare inside procedure, these variable are called as local to procedure and hence accessible only inside procedure.

but when the variable name starts with @ are called as session variables and accessible till the session finishes.

session finishes once you logout.

Loops in PLSQL

1. while ---- top tested loop

while condition do

statements

end while;

1. Repeat… until condition end repeat;
2. loop …..end loop

example

write a program to print 1,2,3,4,5

delimiter //

create procedure test\_while()

begin

declare x int default 1;

declare data varchar(20);

set data='';

while x<=5 DO

set data=concat(data,x,',');

set x=x+1;

end while;

select data;

end//

delimiter ;

to call procedure

call test\_while();

2. Repeat loop

a. this is bottom tested loop

b. it gets executed minimum once

c. it gets execute until given condition is false, as soon as condition becomes true it terminates the loop

Repeat

statements

until condition

end repeat

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

delimiter //

drop procedure if exists test\_repeat//

delimiter //

drop procedure if exists test\_repeat//

create procedure test\_repeat()

begin

declare x int default 1;

declare data varchar(20);

set data='';

repeat

set data=concat(data,x,',');

set x=x+1;

until x > 5

end repeat;

select data;

end//

1. loop, leave , iterate

------leave is same as break statement in java

leave name of lable

-------iterate is same as continue statement in java

the statements you want to repeat enclose it in

loop … end loop

label1: loop

statements

if condition then

leave label1

end if;

end loop;

--------write procedure test\_loop to display 1,2,3,4,5

delimiter //

drop procedure if exists test\_loop//

create procedure test\_loop()

begin

declare x int default 1;

declare data varchar(20);

set data='';

label1:loop

if x>5 then

leave label1;

end if;

set data=concat(data,x,',');

set x=x+1;

iterate label1;

end loop;

select data;

end//

delimiter ;

-------write a procedure to concatenate only numbers between 11 to 20

which are divisible by 5

delimiter //

drop procedure if exists test\_loop//

create procedure test\_loop()

begin

declare x int default 10;

declare data varchar(20);

set data='';

label1:loop

if x>20 then

leave label1;

end if;

set x=x+1;

if (x mod 5 !=0) then

iterate label1;

else

set data=concat(data,x,','); #data =concat(‘’,1,’,’) data =1,

#data =concat(‘1,’,’2,’,’,’) data =1,2,

end if;

end loop;

select data;

end//

call test\_loop();

How to use cursors

cursors are similar to arrays.

when to use cursor

----- if you want to read data from table row by row and process it;

steps to use cursor

1. declare cursor

declare <cursor\_name> cursor from select statement(which returns multiple rows)

declare continue handler for not found set cset=1; #exception handling

1. open cursor ----when you open the cursor the query bound with cursor gets executed, and store the data in the memory which is pointed by cursorname

open <cursor\_name>

1. fetch a row from cursor

fetch <cursorname> into <list of variables>

1. check whether we reached to end of the cursor then leave the loop

if cset=1 then

leave label1;

end if;

1. process data
2. repeat steps from 3 to 5 till you reach to end of the cursor
3. close the cursor after the loop ends.

close cursor

----write a procedure to display all names of employees who are working in dept 10

delimiter //

drop procedure if exists display\_enames\_dept//

create procedure display\_enames\_dept()

begin

declare cset int default 0;

declare veno int;

declare vename varchar(20);

declare empcur cursor for select empno,ename from emp where deptno=10;

declare continue handler for NOT FOUND set cset=1;

open empcur;

label1:loop

fetch empcur into veno,vename;

if cset=1 then

leave label1;

end if;

select veno,vename;

end loop;

close empcur;

end//

delimiter ;

------- to display empno and name in , separated value as shown below for dept qo employees

7782---CLARK,7839---KING,7934---MILLER,1---kishori,12---Revati,123---deepa,

delimiter //

drop procedure if exists display\_enames\_list//

create procedure display\_enames\_list()

begin

declare cset int default 0;

declare veno int;

declare namelist varchar(200);

declare vename varchar(20);

declare empcur cursor for select empno,ename from emp where deptno=10;

declare continue handler for NOT FOUND set cset=1;

open empcur;

set namelist='';

label1:loop

fetch empcur into veno,vename;

if cset=1 then

leave label1;

end if;

set namelist=concat(namelist,veno,"---",vename,',');

end loop;

select namelist;

close empcur;

end//

------- write procedure to update salary of all managers by 10% all analyst by 15% and all clerks by 30%

update emp

set sal=sal\*1.10

where job=’MANAGER’

update emp

set sal=sal\*1.15

where job=’ANALYST’

update emp

set sal=sal\*1.10

where job=’CLERK’

delimiter //

drop procedure if exists update\_sal//

create procedure update\_sal()

begin

declare cset int default 0;

declare veno,vsal,vcomm int;

declare vjob,vename varchar(20);

declare empcur cursor for select empno,ename,job,sal,comm from emp;

declare continue handler for NOT FOUND set cset=1;

open empcur;

label1:loop

fetch empcur into veno,vename,vjob,vsal,vcomm;

if cset=1 then

leave label1;

end if;

if vjob='MANAGER' then

update emp

set sal=vsal\*1.10+ifnull(vcomm,0)

where empno=veno;

elseif vjob='ANALYST' then

update emp

set sal=vsal\*1.15+ ifnull(vcomm,0)

where empno=veno;

elseif vjob='CLERK' then

update emp

set sal=vsal\*1.30+ ifnull(vcomm,0)

where empno=veno;

end if;

end loop;

close empcur;

select \* from emp;

end//

delimiter ;

delimiter //

drop procedure if exists update\_sal//

create procedure update\_sal(mper int,aper int,cper int)

begin

declare cset int default 0;

declare veno,vsal int;

declare vjob,vename varchar(20);

declare empcur cursor for select empno,ename,job,sal from emp;

declare continue handler for NOT FOUND set cset=1;

open empcur;

label1:loop

fetch empcur into veno,vename,vjob,vsal;

if cset=1 then

leave label1;

end if;

if vjob='MANAGER' then

update emp

set sal=vsal+vsal\*(mper/100)

where empno=veno;

elseif vjob='ANALYST' then

update emp

set sal=vsal+vsal\*(aper/100)

where empno=veno;

elseif vjob='CLERK' then

update emp

set sal=vsal+vsal\*(cper/100)

where empno=veno;

end if;

end loop;

close empcur;

select \* from emp;

end//

call update\_sal(15,18,22) #increase manage sal by 15%, analyst sal by 18% and clerk sal by 22%

------- write procedure to calculate experience of employee and display following

expr <38 junior

expr >=38 and <=39

middle mgnt

else

senior

drop procedure if exists display\_type\_emp//

create procedure display\_type\_emp()

begin

declare cset int default 0;

declare veno,vexp int;

declare vsal decimal(9,2);

declare vhiredate date;

declare vename,vstatus varchar(20);

declare empcur cursor for select empno,ename,hiredate,sal from emp;

declare continue handler for NOT FOUND set cset=1;

open empcur;

label1: loop

fetch empcur into veno,vename,vhiredate,vsal;

if cset=1 then

leave label1;

end if;

set vexp= floor(datediff(curdate(),vhiredate)/365);

if vexp<38 then

set vstatus='Junior';

elseif vexp >=38 and vexp<=39 then

set vstatus='middle mgnt';

else

set vstatus='senior';

end if;

select veno,vename,vhiredate,vexp,vsal,vstatus;

end loop;

close empcur;

end//