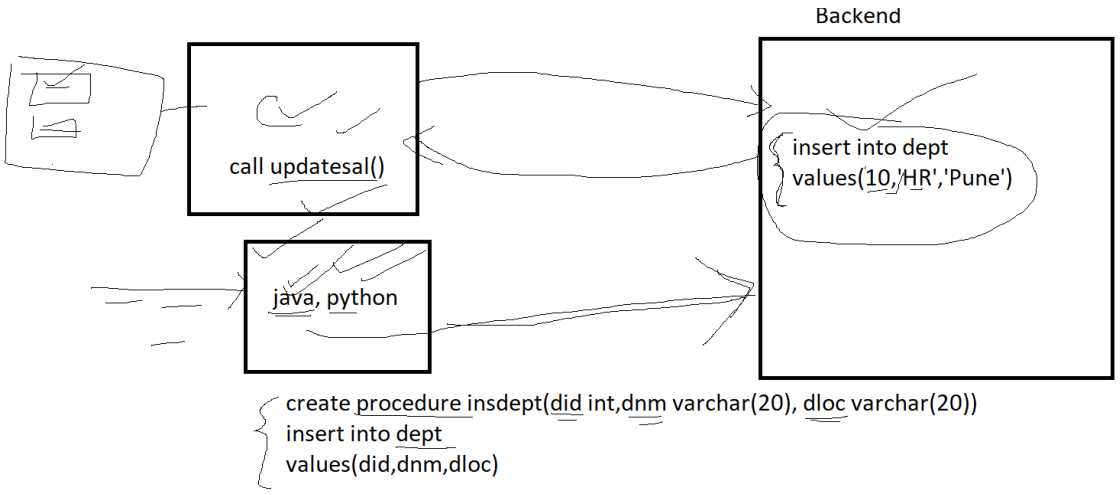
Procedures and functions

Why to use PL SQL

- Queries can be stored in a named block, can be called multiple time, it increases reusability
- It reduces network traffic
- It also increases the performance efficiency
- It increases security, because the developer need not know the syntax and need not know the table names



Procedures, functions, and triggers

Procedure----- named block which do not return any value

Functions----- named block which returns single value as o/p

Triggers------- named block which gets executed automatically for some SQL statement

In MySQL triggers can be written only For DML statements, but in oracle triggers can be written at database level, or for DDL statements or for DML statements and for views

In procedures and function we can pass parameters

These parameters are of 3 types

In --- these parameters are read only parameters

And can be used only for passing data as input

Out----- these are write only parameters

The values can be assigned to these parameters inside the procedure

Inout--- these are read and write parameters, and can be used for passing data as input and its value can be changed inside the procedure

To declare variables

Declare var_name <data type> default <default value>

Declare s int;

Declare s int default 0;

To assign the value to variable

1. Using select …. Into, but the select statement should return single row otherwise it gives error

   Select sal into vsal
   From emp
   Where empno=7902
2. Set x=2;
3. Set x=x+10

To display procedure code

show create procedure updatesal;

Using in parameters

Delimiter $$

Create procedure insdept(in pdid int, in pdnm varchar(20),in pdloc varchar(20))

Begin

  Insert into dept values(pdid,pdnm,pdloc);

End$$

Delimiter ;

Call insdept(12,'HR','Pune');

1. Using out parameters
   create procedure caldept(out pcnt int)
   -> begin
   -> select count(*) into pcnt
   -> from dept;
   -> end$$
   Query OK, 0 rows affected (0.02 sec)

```
mysql> call caldept(@c);
   -> $$
Query OK, 1 row affected (0.02 sec)

mysql> select @c;
   -> $$
+------+
| @c   |
+------+
|    5 |
+------+
1 row in set (0.01 sec)
```

2. Write a procedure to find how many employees are there, and maximum salary in the given department

```
Delimiter $$

Create procedure findcntdept(in pdid int,out pcnt int, out pmax decimal(9,2))

Begin

    Select count(*), max(sal) into pcnt,pmax

    From emp

   Where deptno=pdid;

End$$

Delimiter ;

Call findcntdept(20,@c,@m)
```

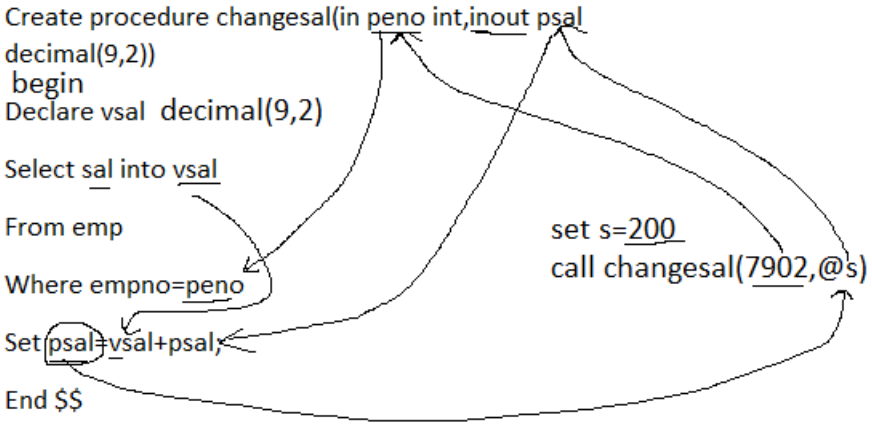
3. Use inout parameters
   Write procedure to find increased salary by given value and get new salary of given emp

```
delimiter $$
create procedure changesal(in peno int,inout psal decimal(9,2))
begin
declare vsal decimal(9,2);
Select sal into vsal
From emp
Where empno=peno;
set psal=vsal+psal;
select psal;
end $$
delimiter ;
```

To execute the procedure

```
Set @s=200
Call changesal(7902,@s);
Select @s;
```

```
Delimiter $$

Create procedure changesal(in peno int,inout psal
decimal(9,2))
 begin
Declare vsal  decimal(9,2)

Select sal into vsal

From emp

Where empno=peno

Set psal=vsal+psal;

End $$
```



4. Write a procedure to update salary of given empno, by given bonus

```
delimiter $$
create procedure updatesal(in peno int,inout bonus decimal(9,2))
begin

  update emp
  set sal=sal+bonus
  where empno=peno;

  select sal
  from emp
  where empno=peno;

 select peno, bonus;
 end$$
delimiter ;
```

Using if---else

Syntax

Mysql If else statement

```
IF expression THEN
  statements;
ELSE
  else-statements;
```

END IF;

Using If ----else--------

IF expression THEN

  statements;

ELSEIF elseif-expression THEN

  elseif-statements;

...

ELSE

  else-statements;

END IF;

Using loops

Whilte

Repeate

Loop…endloop

1. Display “need improvement” if comm is null or 0
   Print ok if com >0 and < 500
   Good if commission >=500 and < 1000
   Excellent otherwise

   Use case statement or write procedure
   Select empno,ename,sal,comm,case when comm is null or comm=0 then “need improvement”
   When comm<500 then ‘ok’
   When comm>=500 and comm<100 then ‘good’
   Else ‘excellent’ end remark
   From emp;

   Using procedure
   Select empno,ename,sal,comm,case when comm is null or comm=0 then 'need improvement'
   When comm<500 then 'ok'
   When comm>=500 and comm<1000 then 'good'
   Else 'excellent' end remark
   From emp;

```
delimiter $$
create procedure assign_remark(in peno int)
begin
 declare remark varchar(20) default '';
 declare veno int;
 declare venm varchar(20);
 declare vcomm decimal(9,2);
 select empno,ename,comm into veno,venm,vcomm
 from emp
 where empno=peno;

 if vcomm is null or vcomm=0 then
    set remark='need improvement';
 elseif vcomm > 0 and vcomm < 500 then
    set remark='ok';
 elseif vcomm>=500 and vcomm<1000 then
    set remark='good';
 else
    set remark='excellent';
 end if;
 select veno,venm,vcomm,remark;

end$$
delimiter ;
```

2. Write procedure to update sal by 10% if job is manager
   20% if job is Analyst
   30% if job is clerk for given empno
   And display updated salary, name , empno and deptno;

```
Create procedure updatesalbyEno(in peno int)
Begin
declare vnm varchar(20);
declare vsal decimal(9,2);
declare vdno int ;
declare vjob varchar(20);

select ename,sal,job,deptno into vnm,vsal,vjob,vdno
from emp
where empno=peno;

if vjob='MANAGER' then
```

```
        set vsal=vsal+vsal*0.10;
    elseif vjob='ANALYST' then
        set vsal=vsal+vsal*0.20;
    elseif vjob='CLERK' then
        set vsal=vsal+vsal*0.30;
    end if;
    update emp
    set sal=vsal
    where empno=peno;
    select vnm,vsal,vjob,vdno,peno;
    end $$
```

Loops in mysql

1. While loop
    a. It is top tested loop
    b. Will continue execution till the condition is true

```
While condition do
        statements
End while;
```

2. Repeat loop
    a. It is bottom tested loop
    b. Will continue execution till the condition is false

```
REPEAT
 statements
Until <condition>
End Repeate
```

3. Loop …. End loop
    a. It is infinite loop and hence to break the loop use leave statement
    b. Iterate ---- similar to continue statement

```
Loop_label:Loop
 If x<5 then
        Leave Loop_label
End if;
End loop;
```

To print 1,2,3,4,5, using while, repeate and loop---end loop;

```
delimiter $$
create procedure display_num()
begin
  # to declare variables
  declare x int default 1;
  declare str varchar(20) default '';
```

```
—this is comment
while x<=5 do
    set str=concat(str,x,',');
    set x=x+1;
  end while;
  str=substr(str,1,length(str)-1);
  select str;

end$$
delimiter ;

delimiter $$
create procedure display_num()
begin
  declare x int default 1;
  declare str varchar(20) default '';
  while x<=5 do
    set str=concat(str,x,',');
    set x=x+1;
  end while;
  str=substr(str,1,length(str)-1);
  select str;

end$$
delimiter ;

delimiter $$
create procedure display_num_loop()
begin
  declare x int default 1;
  declare str varchar(20) default '';
  loop1:loop
    set str=concat(str,x,',');
    set x=x+1;
  if x>5 then
     leave loop1;   # it is equivalent to break statement
  end if;
  end loop;
  set str=substr(str,1,length(str)-1);
  select str;

end$$
delimiter ;
```

4. Write a procedure to accept 2 numbers as i/p start and stop
   Display all numbers divisible 3 between the given range.
   delimiter $$
   Create procedure divisible_by_3(in start int, in stop int)

```
begin
declare str varchar(20) default '';
declare x int default 0;
set x=start;
while x <= stop do
  if x%3=0 then
    set str=concat(str,x,',');
  end if;
  set x=x+1;
end while;
select str;
end$$
delimiter ;
```