**Slide no 7**

delimiter $$

create procedure update\_sal (IN dept int)

begin

update dept\_sal set totalsalary= ( select sum(sal) from employee where dno=dept ) where dpt\_no=dept;

End $$

Delimeter ;

call update\_sal(20);

show procedure status

drop procedure update\_sal ;

call update\_sal(20);

**Slide no 14 out parameter**

delimiter $$

CREATE PROCEDURE my\_proc\_OUT (OUT highest\_salary INT)

BEGIN

SELECT MAX(SAL) INTO highest\_salary FROM employee;

END$$

CALL my\_proc\_OUT(@max\_sal)

select @max\_sal

**slide no 15 inout parameter**

drop procedure my\_proc\_INOUT

delimiter $$

CREATE PROCEDURE my\_proc\_INOUT (INOUT dpt\_name varchar(100))

BEGIN

SELECT concat (dname , " department is at location ",location) INTO dpt\_name FROM department WHERE dname = dpt\_name;

END$$

select \* from department

set @dept\_name="purchase"

call my\_proc\_INOUT(@dept\_name);

select @dept\_nameDELIMITER $$

17

multiple employee id. The employee name will be stored into INOUT parameter employee\_name.

CREATE DEFINER=`root`@`127.0.0.1`

PROCEDURE `GetEmployeeName`(INOUT user\_id varchar(16),

IN emp\_id integer)

BEGIN

DECLARE uname varchar(16);

SELECT name INTO uname

FROM employee

WHERE id = emp\_id ;

IF emp\_id = "1000"

THEN

SET user\_id = "Scott";

ELSEIF emp\_id = "2000"

THEN

SET user\_id = "Palash";

ELSEIF emp\_id = "3000"

THEN

SET user\_id = "Diana";

END IF;

END

Case Statement

DELIMITER $$

CREATE PROCEDURE my\_proc\_CASE

(IN emp\_id INT)

BEGIN

declare salary int;

declare empname character varying(10);

select sal ,name into salary ,empname from employee where id=emp\_id ;

CASE

WHEN (salary >1000)then select concat(empname, " is having salary greater than 1000 ", " ämount is ", salary );

WHEN (salary <1000) then select concat(empname, " is having salary less than 1000 ", " ämount is ", salary );

ELSE select concat(empname, " is having salary equl to 1000 ", " ämount is ", salary );

END CASE;

END$$

call my\_proc\_CASE (,3000) ;

elimiter $$

CREATE procedure proc\_while\_example(in param int)

BEGIN

declare str VARCHAR(255) default '';

declare x INT default 0;

SET x = 1;

WHILE x <= param DO

SET str = CONCAT(str,x, " ");

SET x = x + 1;

END WHILE;

select str;

end $$

CREATE procedure repeat\_loop\_example(In param int)

BEGIN

DECLARE x INT;

DECLARE str VARCHAR(255);

SET x = param;

SET str = 'Start ';

REPEAT

SET str = CONCAT(str,x,',');

SET x = x - 1;

UNTIL x <= 0

END REPEAT;

SELECT str;

END//

call repeat\_loop\_example(1)

----Leave loop example

drop procedure loop\_leave\_example

delimiter //

CREATE procedure loop\_leave\_example(In param int)

BEGIN

DECLARE x INT;

DECLARE str VARCHAR(255);

SET x = 1 ;

SET str='';

loop\_lable :loop

if x>param then

leave loop\_lable;

end if;

set x=x+1;

if (x mod 2) !=0 then

ITERATE loop\_lable;

else

set str= CONCAT(str,x,',');

end if;

end loop;

SELECT str;

END//

BEGIN

DECLARE x INT;

DECLARE str VARCHAR(255);

SET x = 1 ;

SET str='';

loop\_lable :loop

if x>param then

leave loop\_lable;

end if;

set x=x+1;

if (x mod 2) !=0 then

ITERATE loop\_lable;

else

set str= CONCAT(str,x,',');

end if;

end loop;

SELECT str;

END//

**Functions!!**

*drop function giveRaise(double,double)*

*Delimiter $$*

*Create function giveRaise(old\_sal double, hike\_percent double) returns double*

*Begin*

*Declare new\_sal double;*

*Set New\_sal=old\_sal\* (1+ (hike\_percent/100));*

*Return new\_sal;*

*End$$*

*select name,sal,giveRaise(sal,10) from employ*

Eample using Cursor

delimiter $$

Create procedure updatesal\_cur()

Begin

Declare done int default 0;

Declare current\_dnum int;

Declare dnumcur cursor for select dno from employee;

Declare continue handler for not found set done=1;

Open dnumcur;

Repeat

fetch dnumcur into current\_dnum;

Update dept\_sal set totalsalary=(select sum(sal) from employee where dno= current\_dnum)

Where dpt\_no= current\_dnum;

Until done =1

End repeat;

Close dnumcur;

End$$

call updatesal\_cur()

select \* from dept\_sal

*DELIMITER $$*

*CREATE PROCEDURE insertEmployeeDetails\_continue(IN EmpID INT, IN InputEmpName character VARYING (30) )*

*BEGIN*

*DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SELECT " SQL Error occured";*

*INSERT INTO Employee (ID,Name,sal,dno)*

*VALUES (EmpID,InputEmpName,10000.00,30 );*

*SELECT \* FROM Employee;*

*END$$*

*DELIMITER $$*

*CREATE PROCEDURE insertEmployeeDetails\_exit(IN EmpID INT, IN InputEmpName character VARYING (30) )*

*BEGIN*

*DECLARE exit HANDLER FOR SQLEXCEPTION SELECT "Error occurred";*

*INSERT INTO Employee (ID,Name,sal,dno)*

*VALUES (EmpID,InputEmpName,10000.00,30 );*

*SELECT \* FROM Employee;*

*END$$*

*Trigger:*

*delimiter $$*

*create trigger updatesal after insert on employee for each row*

*begin*

*if new.dno is not null then*

*update dept\_sal set totalsalary=TOTALSALARY+ new.sal*

*where dno=new.dno;*

*end if;*

*end $$*

*transaction*

*set autocommit=on;*

*start transaction;*

*update dept\_sal set totalsalary=230;*

*select \* from dept\_sal*

*rollback;*

*commit*