DBMS

PL/SQL

LAB ASSIGNMENT – 4



Ananya Agarwal

102083036

2CO14

EMP(ENO,ENAME, JOB, SALARY, COMMISSION, DEPTNO)

1. Write a PL/SQL code to create a stored procedure fire\_employee to delete employee on the basis of employee number.

CREATE TABLE EMP(EMPID NUMBER(3), ENAME VARCHAR(15), JOB VARCHAR(15), SAL

NUMBER(8));

INSERT INTO EMP VALUES(100, 'Anirudh', 'SDE', 1000000);

INSERT INTO EMP VALUES(101, 'Parth', 'Cloud Expert', 380000);

select \* from emp

create or replace procedure fire\_employee (eid NUMBER) AS

begin

delete from emp where empid = eid;

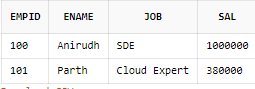
end;

begin

fire\_employee(100);

end;

select \* from emp;



1. Write a PL/SQL code of a local procedure raise\_salary, which accepts two parameters empid and bonus to be added to the salary. It increases the salary of the employee and update it to the database.

CREATE TABLE EMP(EMPID NUMBER(3), ENAME VARCHAR(15), JOB VARCHAR(15), SAL

NUMBER(8));

INSERT INTO EMP VALUES(100, 'Anirudh', 'SDE', 1000000);

DECLARE

data emp%rowtype;

newSal number;

procedure raise\_salary(eid in number, salary in number) as

begin

select \* into data from emp where empid = eid;

newSal := data.sal + salary;

update emp set sal = newSal where data.empid = eid;

end;

begin

raise\_salary (100, 32000);

end;

select \* from emp;

C:\Users\User\Desktop\3.PNG

3. Write a function to add two numbers.

create or replace function addition(x in number, y in number) return number is

result number;

begin

result := x+y;

return result;

end;

declare

num1 number;num2 number;

begin

num1:=34;

num2:=61;

dbms\_output.put\_line(num1|| ' + ' ||num2|| ' = ' || addition(num1, num2));

end;

C:\Users\User\Desktop\1.PNG

1. Write a stored function that accepts department number and return total salary of that department.

CREATE TABLE EMP(ENO NUMBER, ENAME VARCHAR(20),

JOB VARCHAR(15), SALARY NUMBER, DEPTNO NUMBER);

declare

total number;

sal number;

dno number;

function salary(dept\_no number)

return number

is

total number;

begin

select sum(sal) into total from emp where deptno = dept\_no;

return total;

end;

begin

dno := 20;

sal := salary(dno);

dbms\_output.put\_line('Total Salary for deptno '|| dno || ' is' ||sal);

end;

C:\Users\User\Desktop\1.PNG

5. Create a trigger on the emp table, which shows the old values and new value of ename after every updation on ename of emp table.

CREATE TABLE EMP(ENO NUMBER, ENAME VARCHAR(20), JOB VARCHAR2(20));

INSERT INTO EMP VALUES(1,'anirudh', 'sde');

INSERT INTO EMP VALUES(2,'parth', 'cloud architect');

CREATE OR REPLACE TRIGGER EMP\_UPDATE\_OLD\_AND\_NEW\_VAL BEFORE UPDATE OF ENAME

ON EMP

FOR EACH ROW

BEGIN

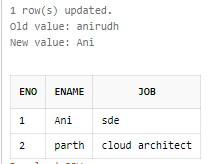
DBMS\_OUTPUT.PUT\_LINE('Old value: '||:old.ename);

DBMS\_OUTPUT.PUT\_LINE('New value: '||:new.ename);

END;

UPDATE EMP SET ENAME = 'Ani' WHERE ENO = 1;

SELECT \* FROM EMP;



6. Create a trigger so that no operation can be performed on EMP table on Sunday.

CREATE TABLE EMP(ENO NUMBER, ENAME VARCHAR(20), JOB VARCHAR2(20));

CREATE OR REPLACE TRIGGER SUNDAY\_CLOSED

BEFORE INSERT OR DELETE OR UPDATE

ON EMP

DECLARE

day varchar2(10);

BEGIN

day:=TRIM(TO\_CHAR(SYSDATE,'day'));

IF day='sunday' THEN

RAISE\_APPLICATION\_ERROR(-20016,'No Work will be done on Sunday');

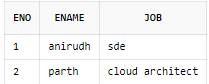
END IF;

END;

INSERT INTO EMP VALUES(1,'anirudh', 'sde');

INSERT INTO EMP VALUES(2,'parth', 'cloud architect');

SELECT \* FROM EMP;



7. Write a trigger to ensure that commission of employee cannot be greater than his salary.

CREATE TABLE EMP(ENO NUMBER PRIMARY KEY, ENAME VARCHAR(20),

JOB VARCHAR2(15), SALARY NUMBER, COMMISSION NUMBER, DEPTNO NUMBER);

CREATE OR REPLACE TRIGGER COMMISSION\_RESTRICT

BEFORE INSERT OR UPDATE

ON EMP

FOR EACH ROW

BEGIN

IF (:new.COMMISSION>:new.SALARY)THEN

RAISE\_APPLICATION\_ERROR(-20016,'COMMISSION CANNOT BE GREATER THAN

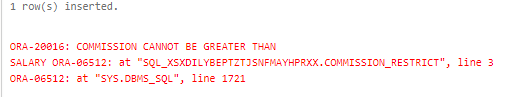
SALARY');

END IF;

END;

INSERT INTO EMP VALUES(1,'anirudh', 'sde', 100000, 20000, 10);

UPDATE EMP SET COMMISSION = 300000 WHERE DEPTNO = 10;



8. Write a trigger to implement foreign key constraint on DEPTNO column of EMP table.

CREATE TABLE EMP(ENO NUMBER, ENAME VARCHAR(20),

JOB VARCHAR(15), SALARY NUMBER, DEPTNO NUMBER);

CREATE OR REPLACE TRIGGER PRIMARY\_KEY\_ENO

BEFORE INSERT OR UPDATE OF ENO

ON EMP

FOR EACH ROW

DECLARE

var EMP.ENO%TYPE;

BEGIN

IF (:new.ENO IS NULL)THEN

RAISE\_APPLICATION\_ERROR(-20016,'ENO CANNOT BE A NULL VALUE');

END IF;

SELECT ENO INTO var FROM EMP WHERE ENO = :new.ENO;

RAISE\_APPLICATION\_ERROR(-20017,'ENO SHOULD BE UNIQUE');

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

NULL;

END;

INSERT INTO EMP VALUES(1,'anirudh', 'sde', 120000, 10);

INSERT INTO EMP VALUES(1,'sparsh', 'cloud expert', 90000, 10);

INSERT INTO EMP VALUES(2,'shivam', 'android dev', 100000, 11);

INSERT INTO EMP VALUES(NULL, 'bhagyesh', 'clerk', 5000, 12);

UPDATE EMP SET ENO = NULL;

SELECT \* FROM EMP;

