1. **Area of Circle**

declare

pi constant number(4,2):=3.14;

radius number(2);

area number(14,2);

begin

radius :=3;

while radius <=7

loop

area := pi\*power(radius,2);

insert into Circle values(radius,area);

radius := radius+1;

end loop;

end;

/

**2.**[**Insert Record using Anonymous Block**](https://cognizant.tekstac.com/mod/vpl/view.php?id=34248)

BEGIN

INSERT INTO Department(DEPARTMENT\_ID,DEPARTMENT\_NAME,LOCATION\_ID)

select max(DEPARTMENT\_ID)+10,'TESTING','CHN-102' from Department;

END;

/

## 3. Update Location

begin

update department

set LOCATION\_ID='HQ-BLR-101'

WHERE LOCATION\_ID LIKE 'HQ%';

END;

/

## 4. Delete Record(s) at a particular location

begin

delete from department

WHERE LOCATION\_ID='CHN-102';

END;

/

## 5. Display department names using Cursors

SET SERVEROUTPUT ON;

DECLARE v\_dept department.department\_name%type;

CURSOR c\_dept is SELECT department\_name FROM department order by

department\_name asc;

BEGIN dbms\_output.put\_line('Department Names are :');

OPEN c\_dept;

LOOP

FETCH c\_dept INTO v\_dept;

EXIT WHEN c\_dept%notfound;

dbms\_output.put\_line(v\_dept);

END LOOP;

CLOSE c\_dept; END;

**6. Department Highest Salary Expenditure**

set serveroutput on;

declare

cursor c\_depts is select DEPT, sum(SALARY) as sumsal from employee group by DEPT;

v\_depts c\_depts%rowtype;

begin

open c\_depts;

dbms\_output.put\_line('Department-wise salary expenditure:');

loop

fetch c\_depts into v\_depts;

exit when c\_depts%notfound;

dbms\_output.put\_line(v\_depts.DEPT || ' department, total salary is ' || v\_depts.sumsal );

end loop;

close c\_depts;

end;

/

## 7. Department Details

SET SERVEROUTPUT ON;

DECLARE

v\_dept department.department\_id%type;

w\_dept department.department\_name%type;

x\_dept department.location\_id%type;

CURSOR c\_dept is SELECT department\_id, department\_name, location\_id FROM department order by

department\_id asc;

BEGIN dbms\_output.put\_line('Department Details are :');

OPEN c\_dept;

LOOP

FETCH c\_dept INTO v\_dept, w\_dept, x\_dept;

EXIT WHEN c\_dept%notfound;

dbms\_output.put\_line(v\_dept || ', ' || w\_dept || ', ' || x\_dept);

END LOOP;

CLOSE c\_dept;

END;

/

## 8. Procedure with Exception Handling

create or replace PROCEDURE CHECK\_AGE\_ELIGIBILITY(

v\_id IN EMPLOYEE.EMPID%TYPE,

v\_name IN EMPLOYEE.EMPNAME%TYPE,

v\_age IN EMPLOYEE.AGE%TYPE) as

not\_of\_age exception;

begin

if v\_age>=18 then

insert into EMPLOYEE(empid,empname,age) values(v\_age,v\_name,v\_age);

dbms\_output.put\_line('Age valid - Record inserted');

else

raise not\_of\_age;

end if;

exception

when not\_of\_age then

dbms\_output.put\_line('Age invalid - Record not inserted');

end;

/

## 9. Calculate increment using Function

create or replace function calculate\_increment

(incperc in number,

salary in number

)

return number

as

begin

return salary \* incperc/100;

end;

/

## 10. Remove Employee records - Procedures

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE DELETE\_EMPLOYEE(V\_DEPT IN EMPLOYEE.DEPT%TYPE)

IS

DeptNotFoundException EXCEPTION;

BEGIN

DELETE FROM EMPLOYEE WHERE DEPT = V\_DEPT;

IF SQL%FOUND THEN

DBMS\_OUTPUT.PUT\_LINE(SQL%ROWCOUNT || ' Employee record(s) got deleted.');

ELSE

RAISE DeptNotFoundException;

END IF;

EXCEPTION

WHEN DeptNotFoundException

THEN

DBMS\_OUTPUT.PUT\_LINE( 'No Records found.');

END;

/

## 11. Concatenate Employee names - Functions

create or replace function concat\_name(v\_id employee.emp\_id%type)

return varchar2

is

retval varchar2(25);

begin

select first\_name || last\_name

into retval

from employee

where emp\_id = v\_id;

return retval;

end;

/

## 12. Package with a Procedure to update salary

CREATE OR REPLACE PACKAGE emp\_designation

AS

PROCEDURE emp\_details (design employee.designation%TYPE, incentive NUMBER);

END emp\_designation;

/

CREATE OR REPLACE PACKAGE BODY emp\_designation

AS

PROCEDURE emp\_details (design employee.designation%TYPE, incentive NUMBER)

IS

BEGIN

UPDATE employee

SET employee.salary = employee.salary + incentive

WHERE employee.designation = design;

DBMS\_OUTPUT.put\_line(SQL%ROWCOUNT || ' employee(s) are updated.');

END emp\_details;

END emp\_designation;

/

## 13. Insert a Record - Triggers

create or replace trigger trigger1

after insert on employee

for each row

begin

dbms\_output.put\_line('NEW EMPLOYEE DETAILS INSERTED');

end;

## 14. Delete a Record - Triggers

create or replace trigger trg\_bd\_emp

before delete on employee

for each row

begin

insert into employee\_archive (empid, empname, managerid, deleted\_date)

values (:old.empid, :old.empname, :old.managerid, sysdate);

end trg\_bd\_emp;

/