# S.I.E.S College of Arts, Science and Commerce Sion(W), Mumbai – 400 022.

**CERTIFICATE**

This is to certify that Master **Lingraj Ramu Namal**

Roll No. **SCS2223054** Has successfully completed the necessary course of experiments in the subject of **DATABASE MANAGEMENT SYSTEMS** during the academic year **2022 – 2023** complying with the requirements of **University of Mumbai**, for the course of **F.Y.BSc. Computer Science [Semester-1]**

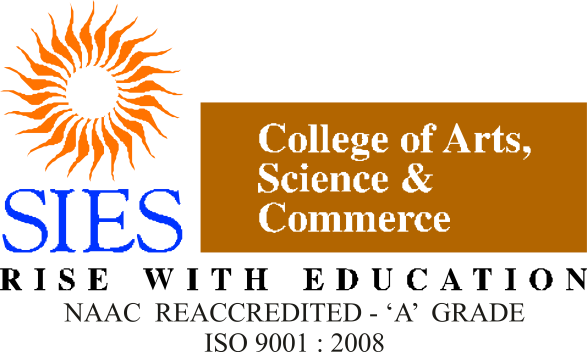
Prof. In-Charge

# Dr. Abuzar Ansari

**(DATABASE MANAGEMENT SYSTEMS)**

Examination Date: Examiner’s Signature & Date:

Head of the Department



# Prof. Manoj Singh

College Seal And Date

DBMS Practical Journal

# Practical 1 )c’

Aim :- Creating and working with Insert/Update/Delete Trigger using Before/After clause. Question :- Create table emp (eno, ename, hrs, pno, super\_no) & project (pname, pno, thrs, head\_no) where thrs is the total hours and is the derived attribute. Its value is the sum of all employees working on that project. eno and pno are primary keys, head\_no is foreign key to emp relation. Insert 10 tuples and write triggers to do the following :

* Creating a trigger to insert new employee tuple and display the new total hours from project table.
* Creating a trigger to change the hrs of existing employee and display the new total hours from project table.
* Creating a trigger to change the project of an employee and display the new total hours from project table.
* Creating a trigger to delete the project of an employee

# Code:

CREATE TABLE project (pno NUMBER(3) PRIMARY KEY, pname VARCHAR(20), thrs NUMBER(4), super\_no NUMBER(2));

CREATE TABLE emp (eno NUMBER(3) PRIMARY KEY, ename VARCHAR(25), hrs

NUMBER(2), pno NUMBER(2) REFERENCES project(pno), super\_no NUMBER(2));

/\* A. INSERT TRIGGER \*/

CREATE OR REPLACE TRIGGER THRS AFTER INSERT ON emp

FOR EACH ROW WHEN(NEW.pno IS NOT NULL) BEGIN

UPDATE project SET thrs = thrs + :NEW.hrs WHERE pno = :NEW.pno; END;

/

/\* B. Update Trigger \*/

CREATE OR REPLACE TRIGGER emp\_update\_trigger AFTER UPDATE ON emp

FOR EACH ROW WHEN(NEW.pno IS NOT NULL) BEGIN

UPDATE project SET thrs = thrs - :OLD.hrs + :NEW.hrs WHERE pno = :NEW.pno; END;

/

/\* C. Delete Trigger \*/

CREATE OR REPLACE TRIGGER emp\_delete\_trigger BEFORE DELETE ON emp

FOR EACH ROW WHEN(OLD.pno is NOT NULL) BEGIN

UPDATE project SET thrs = thrs - :OLD.hrs WHERE pno = :OLD.pno; END;

/

/\* Main \*/

INSERT INTO project VALUES(1, 'College Website', 0, 1); INSERT INTO project VALUES(2, 'Anime Website', 0, 1); INSERT INTO project VALUES(3, 'Dragon Website', 0, 1);

INSERT INTO emp VALUES(1, 'Harsh', 6, 1, 1);

INSERT INTO emp VALUES(2, 'Varun', 5, 2, 7);

INSERT INTO emp VALUES(3, 'Aneesh', 8, 3, 2);

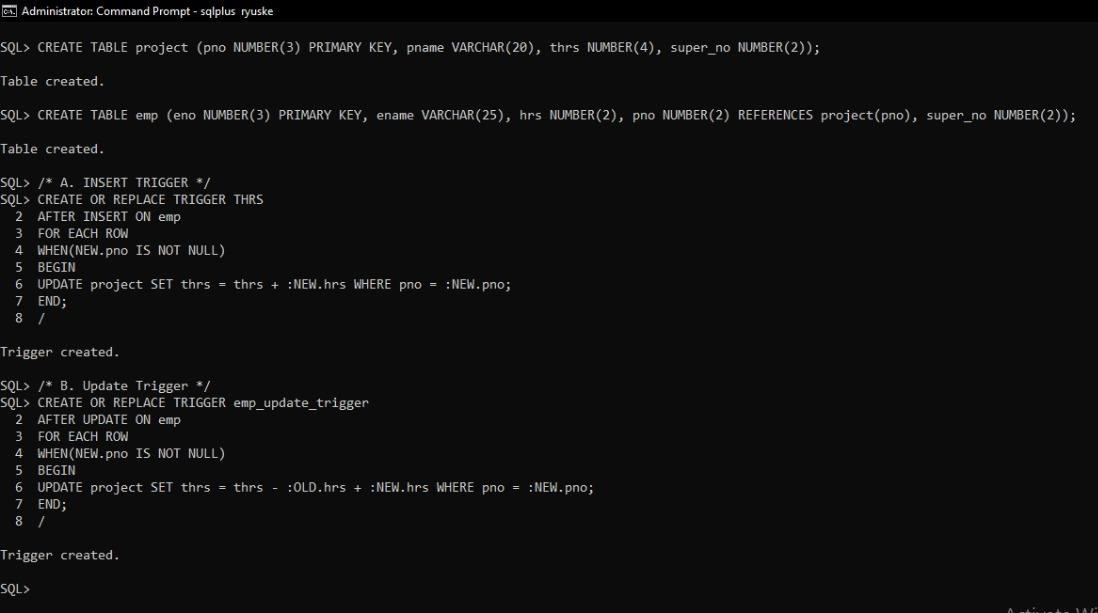
select \* from project;

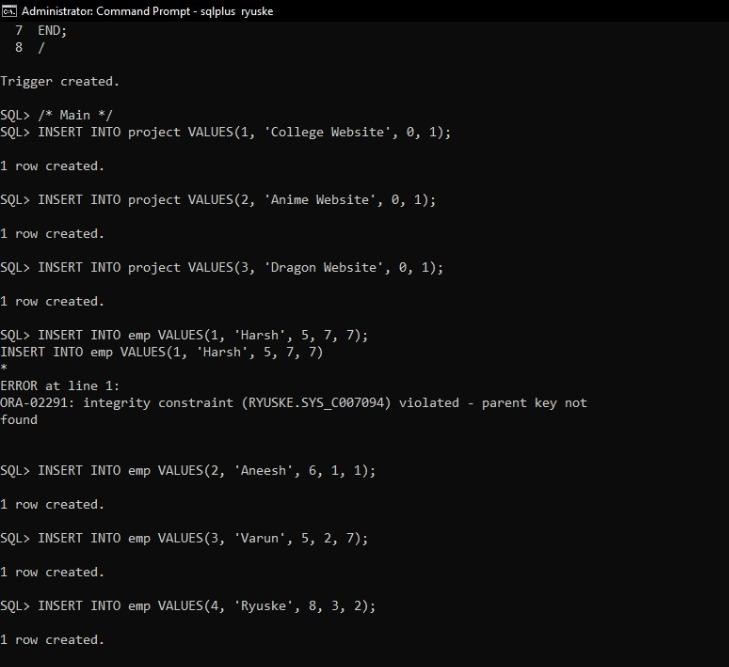
UPDATE emp SET hrs = 7 WHERE eno = 1; SELECT \* from project;

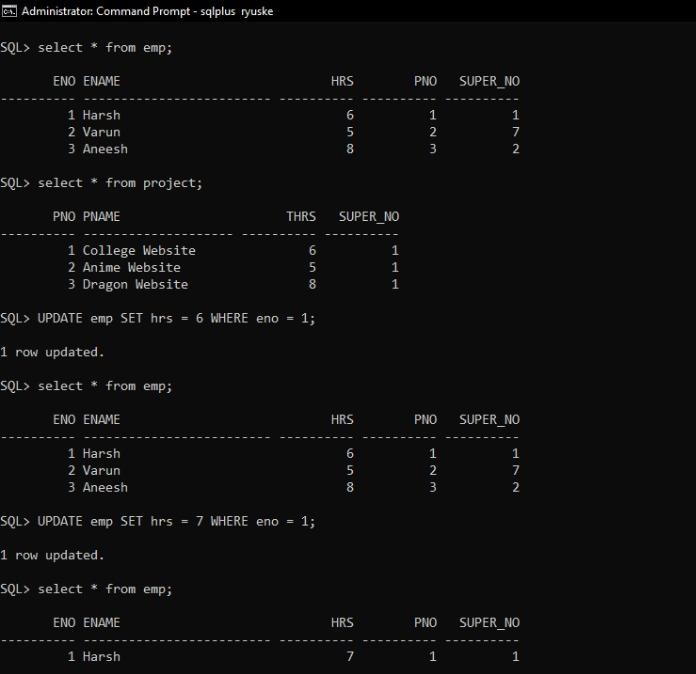
DELETE FROM emp WHERE eno = 3;

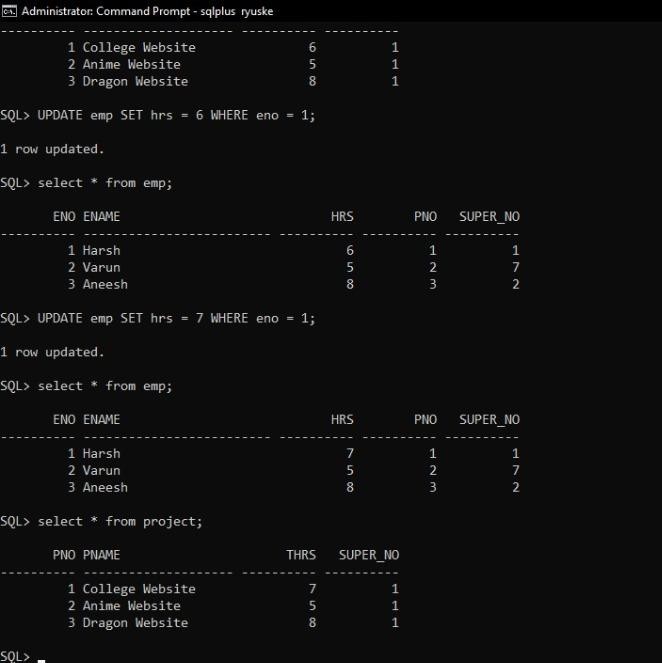
SELECT \* FROM project;

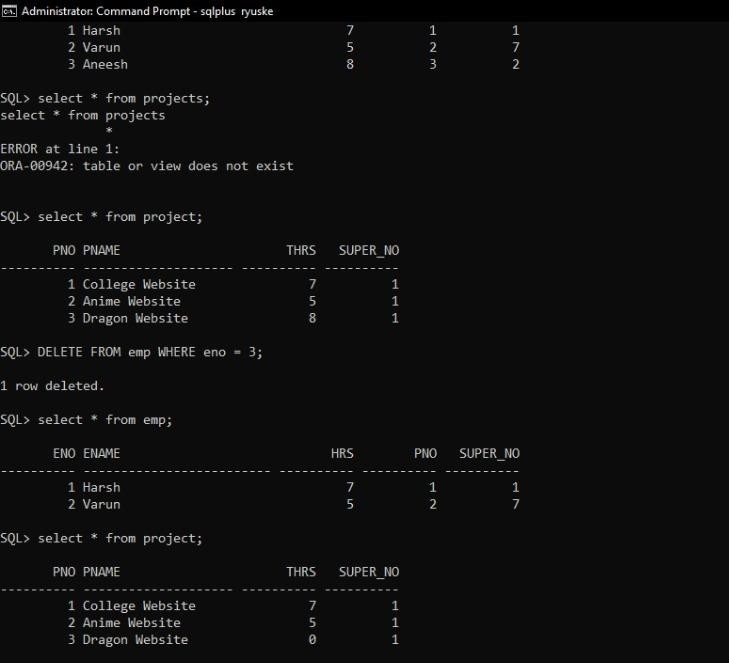
**Now Lets put Inputs and check their Outputs:**











# Practical 2

) c’ Aim :- Writing PL/SQL Blocks with basic programming constructs by including following : Sequential Statements unconstrained loop. Question :- Write a PL/SQL block to insert 100 values in the table using loop [exit when] also auto increment the supplier id with the help of sequence. Table : Supplier {supid number(5) , suppname varchar2 (15)}

# Input:

CREATE SEQUENCE SUPPID MINVALUE 1

MAXVALUE 9999

START WITH 10

INCREMENT BY 5

CACHE 20;

CREATE TABLE SUPPLIER (SUPPID number(5), SUPPNAME varchar2(15)); DECLARE

V\_COUNTER NUMBER(5):=0; V\_SUPPID SUPPLIER.SUPPID%TYPE;

V\_SUPPNAME SUPPLIER.SUPPNAME%TYPE:='HARSH'; BEGIN

LOOP

INSERT INTO SUPPLIER (SUPPNAME,SUPPID) VALUES (V\_SUPPNAME,SUPPID.NEXTVAL); V\_COUNTER:=V\_COUNTER+1;

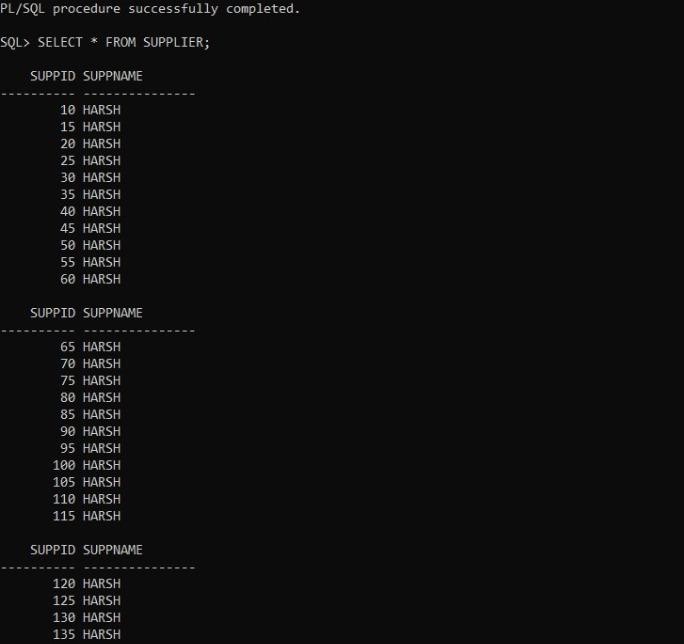
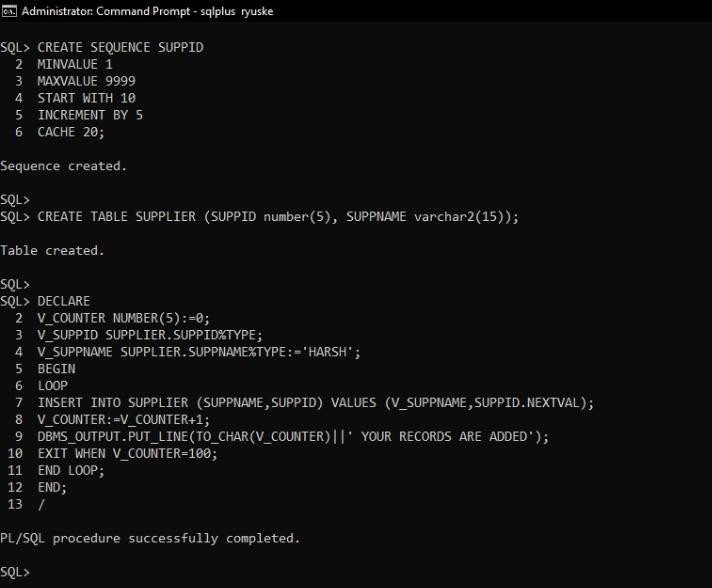
DBMS\_OUTPUT.PUT\_LINE(TO\_CHAR(V\_COUNTER)||' YOUR RECORDS ARE ADDED');

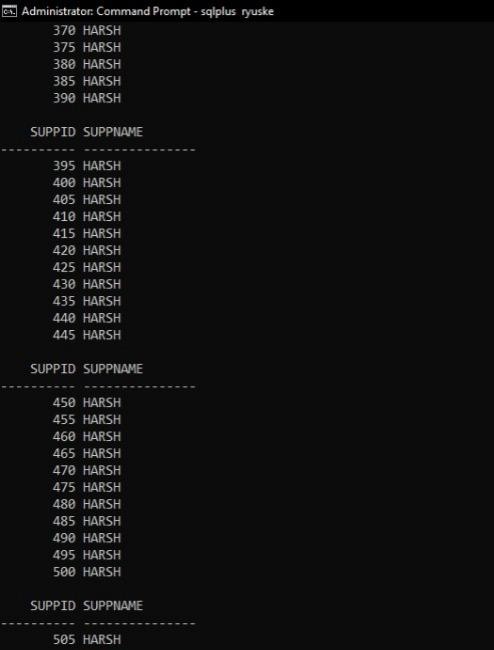
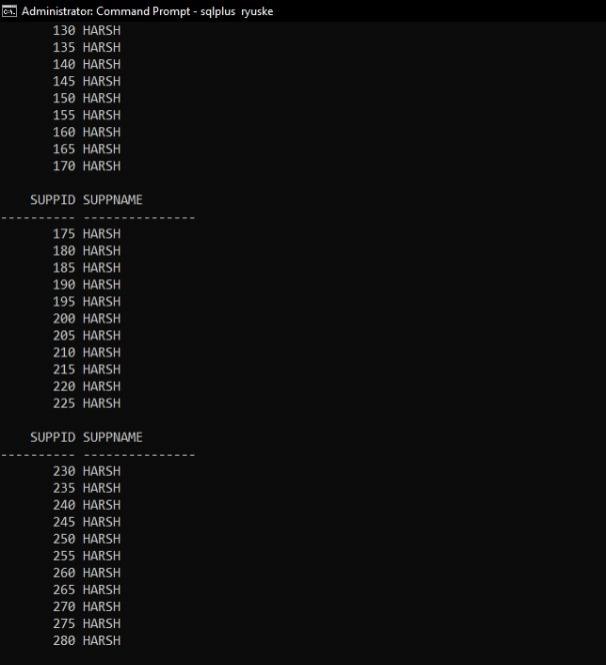
EXIT WHEN V\_COUNTER=100; END LOOP;

END;

/

# Applying the input and output:





**Practical 3**

)c’ Aim :- Creating simple Sequences with clauses like START WITH, INCREMENT BY, MAXVALUE, MINVALUE, CYCLE | NOCYCLE, CACHE | NOCACHE, ORDER |

NOORECER. Creating and using Sequences for tables.

Question : - Write PL/SQL block to insert 10 records in Employee table , empno should be auto incremented , should begin with 10, the employee number should recycle after reaching to empno 50. Empno should increment by 10. [use sequence for empno ]

CREATE TABLE emp(eno NUMBER(5), ename VARCHAR(15));

CREATE SEQUENCE practical3\_sequence START WITH 10 CYCLE MINVALUE 10 MAXVALUE 50 INCREMENT BY 10 CACHE 2;

BEGIN

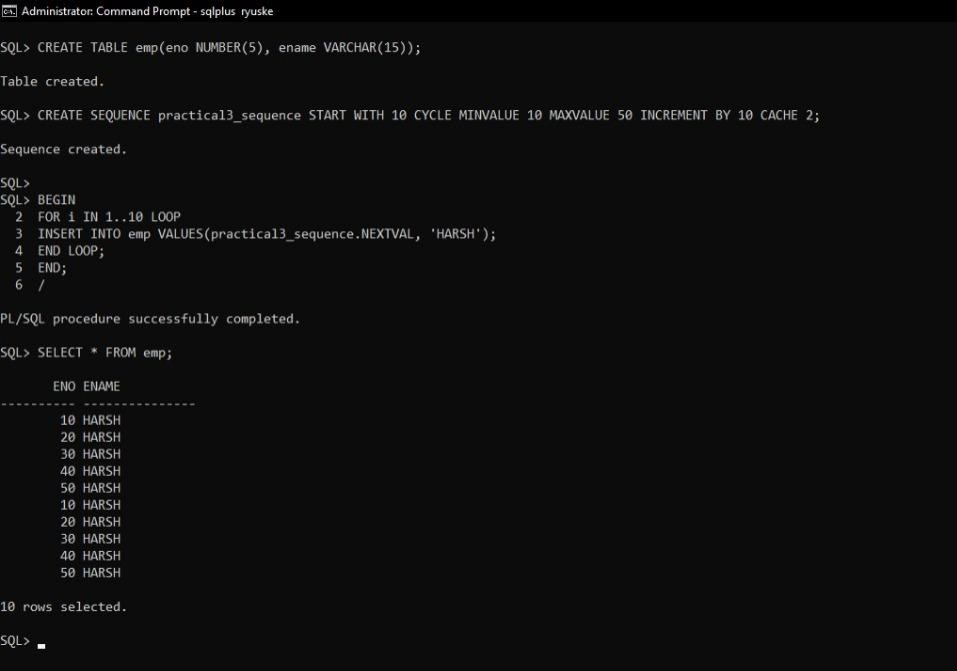
FOR i IN 1..10 LOOP

INSERT INTO emp VALUES(practical3\_sequence.NEXTVAL, 'HARSH'); END LOOP;

END;

/

SELECT \* FROM emp;



# Practical 04:

Aim: Writing PL/SQL Blocks with basic programming constructs by including following: If...then...Else, IF...ELSIF...ELSE... END IF Case statement

Question:-. Write a PL/SQL block to increase the salary of the employee by 15% whose date of joining is 12 December 2000 otherwise increase salary by 5%. Table : Employee { empno number(10), ename varchar2(10), salary number(10)} PL/SQL:- About ‘If...then...Else, IF...ELSIF...ELSE... END IF”

DROP TABLE emp;

CREATE TABLE emp (eno NUMBER(3) PRIMARY KEY, ename VARCHAR2(25), salary

NUMBER(6), join\_date DATE);

INSERT INTO emp VALUES(1, 'Harsh', 15000,TO\_DATE('20030412','YYYYMMDD')); INSERT INTO emp VALUES(2, 'Ryuske' , 15000, TO\_DATE('20001212','YYYYMMDD'));

INSERT INTO emp VALUES(3, 'Zangetu' , 20000, TO\_DATE('20020805','YYYYMMDD'));

INSERT INTO emp VALUES(4, 'Luffy' , 10000, TO\_DATE('20020806','YYYYMMDD'));

DECLARE CURSOR emp\_cursor IS SELECT eno, salary, join\_date FROM emp; v\_eno emp.eno%TYPE; v\_salary emp.salary%TYPE;

v\_join\_date emp.join\_date%TYPE; BEGIN

OPEN emp\_cursor;

LOOP

FETCH emp\_cursor INTO v\_eno, v\_salary, v\_join\_date; IF emp\_cursor%NOTFOUND THEN

EXIT;

ELSIF v\_join\_date = TO\_DATE('20001212','YYYYMMDD') THEN

UPDATE emp SET salary = v\_salary\*1.15 WHERE eno = v\_eno; ELSE

UPDATE emp SET salary = v\_salary\*1.05 WHERE eno = v\_eno; END IF;

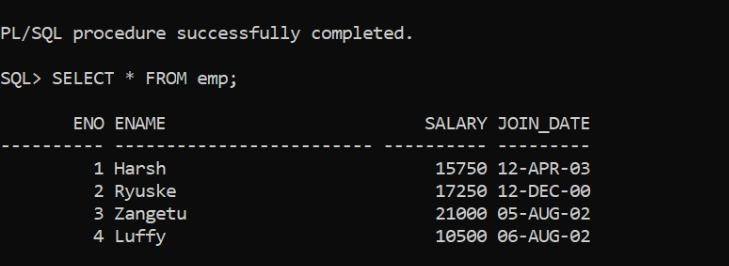
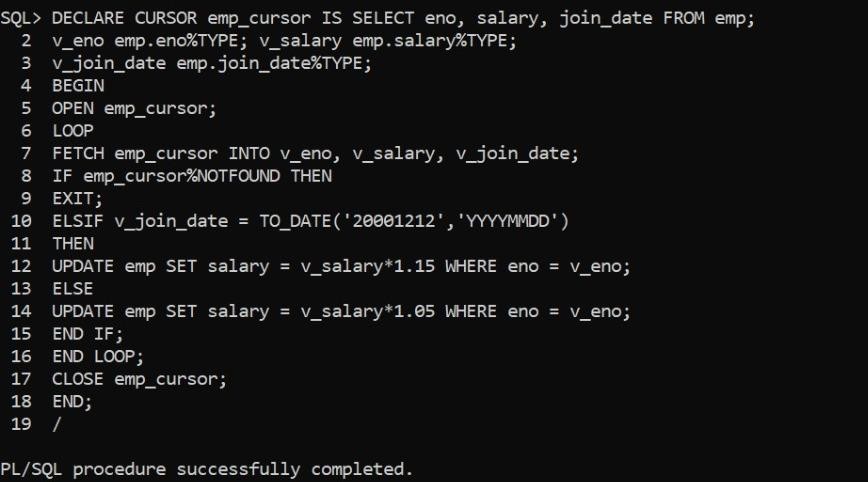
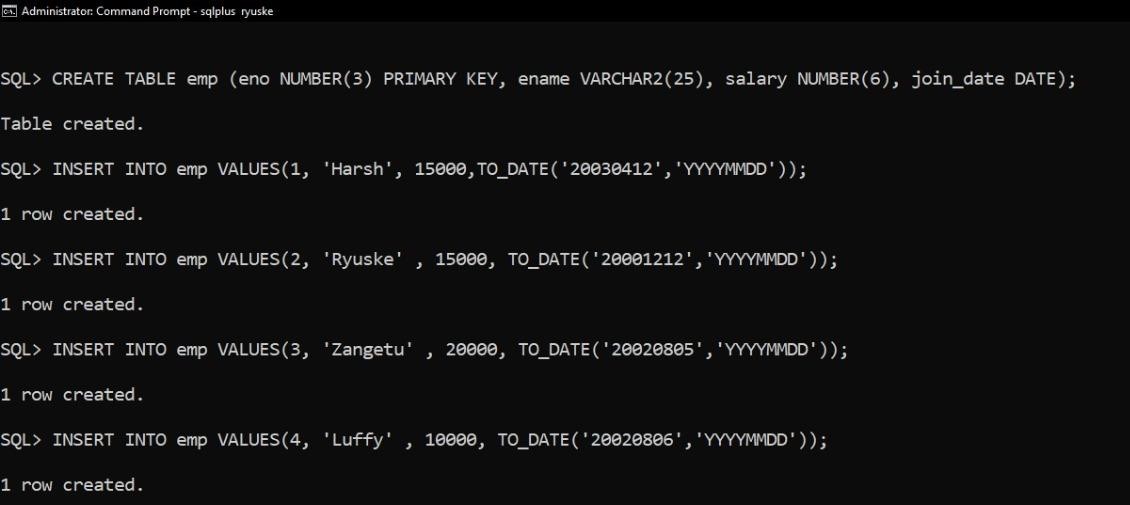
END LOOP;

CLOSE emp\_cursor;

END;

/

# Let’s implement the code:



**Practical 05**

# Aim:

Create a table lecturer with one of the attribute as major subject. Write a PL/SQL block with CASE WHEN statement with variable which print the course name depending upon the major

subject for the specified lecturer id.

Table : Lecturer { lid NUMBER(4), lname varchar2(14), majorsubject varchar2(10), coursename varchar2(15)}

# Code:

SET SERVEROUTPUT ON

CREATE TABLE lecturer (lid NUMBER(4) PRIMARY KEY, lname VARCHAR2(14),

majorsubject VARCHAR2(10));

INSERT INTO lecturer VALUES(1, 'HARSH', 'CS'); INSERT INTO lecturer VALUES(2, 'ANEESH', 'A.I'); INSERT INTO lecturer VALUES(3, 'RYUSKE', 'BMM'); INSERT INTO lecturer VALUES(4, 'ZANGETSU', 'CB');

DECLARE

v\_lid lecturer.lid%TYPE := &v\_lid; v\_lname lecturer.lname%TYPE;

v\_majorsubject lecturer.majorsubject%TYPE; BEGIN

SELECT lname, majorsubject INTO v\_lname, v\_majorsubject FROM lecturer WHERE lid = v\_lid;

CASE v\_majorsubject

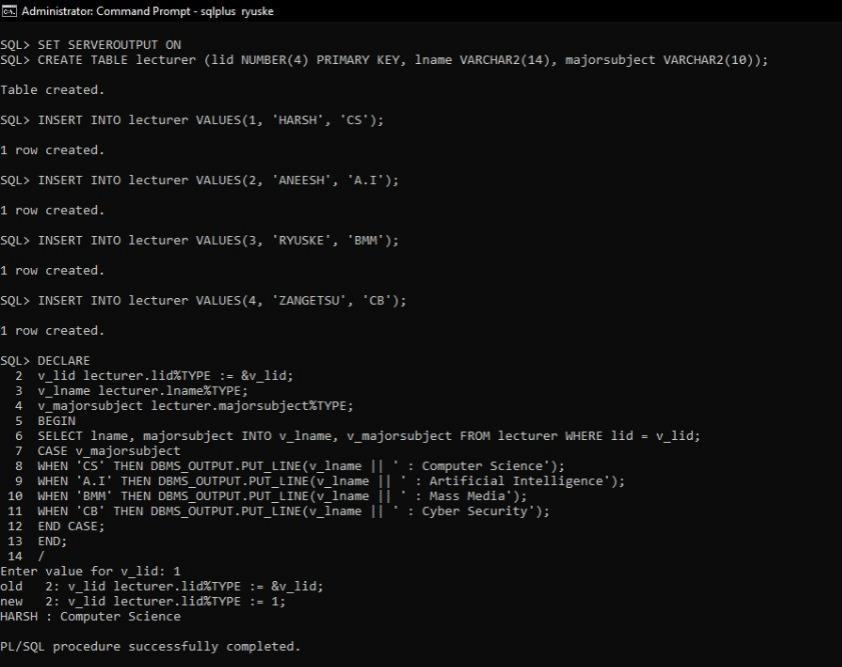
WHEN 'CS' THEN DBMS\_OUTPUT.PUT\_LINE(v\_lname || ' : Computer Science'); WHEN 'A.I' THEN DBMS\_OUTPUT.PUT\_LINE(v\_lname || ' : Artificial Intelligence'); WHEN 'BMM' THEN DBMS\_OUTPUT.PUT\_LINE(v\_lname || ' : Mass Media'); WHEN 'CB' THEN DBMS\_OUTPUT.PUT\_LINE(v\_lname || ' : Cyber Security');

END CASE;

END;

/

**Let’s implement and check the output:**



# Practical 6:

Aim :- Writing PL/SQL Blocks with basic programming constructs for

following Iterative Structure : While-loop Statements For-loop Statements. Question : - Write a PL/SQL block to print department wise total number of employees along the total salary paid to that department. [ USE : 1) While loop 2)For loop ]

Table(s) : Employee { empno number(4) primary key, ename varchar2(10), salary number(10), deptno number(4) reference key} Department { deptno number(4) primary key , dname varchar2(4) }

Code:

SET SERVEROUTPUT ON

CREATE TABLE department(deptno NUMBER(4) PRIMARY KEY, dname VARCHAR2(40));

INSERT INTO department VALUES(1, 'Computer Science'); INSERT INTO department VALUES(2, 'Artificial Intelligence'); INSERT INTO department VALUES(3, 'Data Science');

CREATE TABLE employee (empno NUMBER(4) PRIMARY KEY, ename VARCHAR2(10), salary NUMBER(10), deptno NUMBER(4) REFERENCES

department(deptno));

INSERT INTO employee VALUES(1, 'Harsh', 17000, 2); INSERT INTO employee VALUES(2, 'Aneesh' , 15000, 1); INSERT INTO employee VALUES(3, 'Vinit' , 70000, 3); INSERT INTO employee VALUES(4, 'Varun' , 50000, 2); INSERT INTO employee VALUES(5, 'Adarsh' , 60000, 2); INSERT INTO employee VALUES(6, 'Kartik' , 18000, 3); INSERT INTO employee VALUES(7, 'Shariq', 90000, 1); INSERT INTO employee VALUES(8, 'Vedant' , 20000, 3); INSERT INTO employee VALUES(9, 'Yash' , 14000, 1); INSERT INTO employee VALUES(10, 'Om' , 21000, 1); INSERT INTO employee VALUES(11, 'Surya' , 13000, 2); INSERT INTO employee VALUES(12, 'Kabilan' , 17000, 3); INSERT INTO employee VALUES(13, 'Ivin' , 37000, 2); INSERT INTO employee VALUES(14, 'Ryuske' , 89000, 1); INSERT INTO employee VALUES(15, 'Luffy' , 89000, 3); INSERT INTO employee VALUES(16, 'Senku' , 56000, 2);

DECLARE

CURSOR department\_cursor IS SELECT deptno, dname FROM department; v\_deptno department.deptno%TYPE;

v\_dname department.dname%TYPE; v\_salary\_sum NUMBER(10); v\_employee\_count NUMBER(3); BEGIN

OPEN department\_cursor;

FETCH department\_cursor INTO v\_deptno, v\_dname;

WHILE department\_cursor%FOUND LOOP

SELECT COUNT(empno), SUM(salary) INTO v\_employee\_count, v\_salary\_sum FROM employee

WHERE deptno = v\_deptno;

DBMS\_OUTPUT.PUT\_LINE(v\_deptno || ' ' || v\_dname || ' Total Salary: ' || v\_salary\_sum || ' Employee Count:' || v\_employee\_count);

FETCH department\_cursor INTO v\_deptno, v\_dname; END LOOP;

CLOSE department\_cursor; END;

/

DECLARE

CURSOR department\_cursor IS SELECT \* FROM department; v\_salary\_sum NUMBER(10);

v\_employee\_count NUMBER(3); v\_department\_row department%ROWTYPE; BEGIN

FOR v\_department\_row IN department\_cursor LOOP

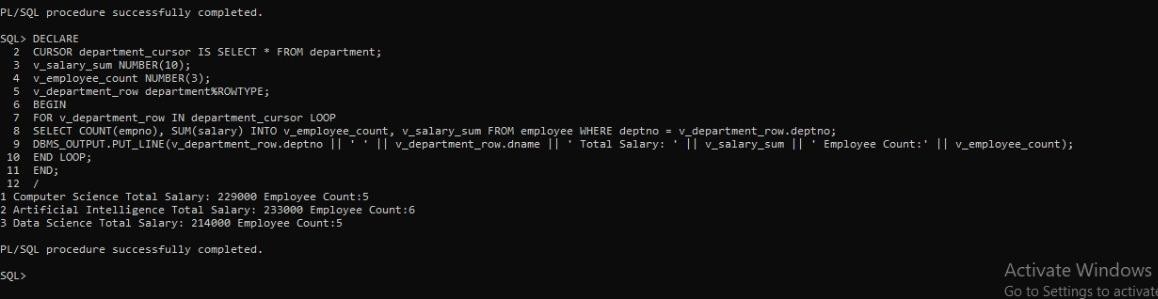
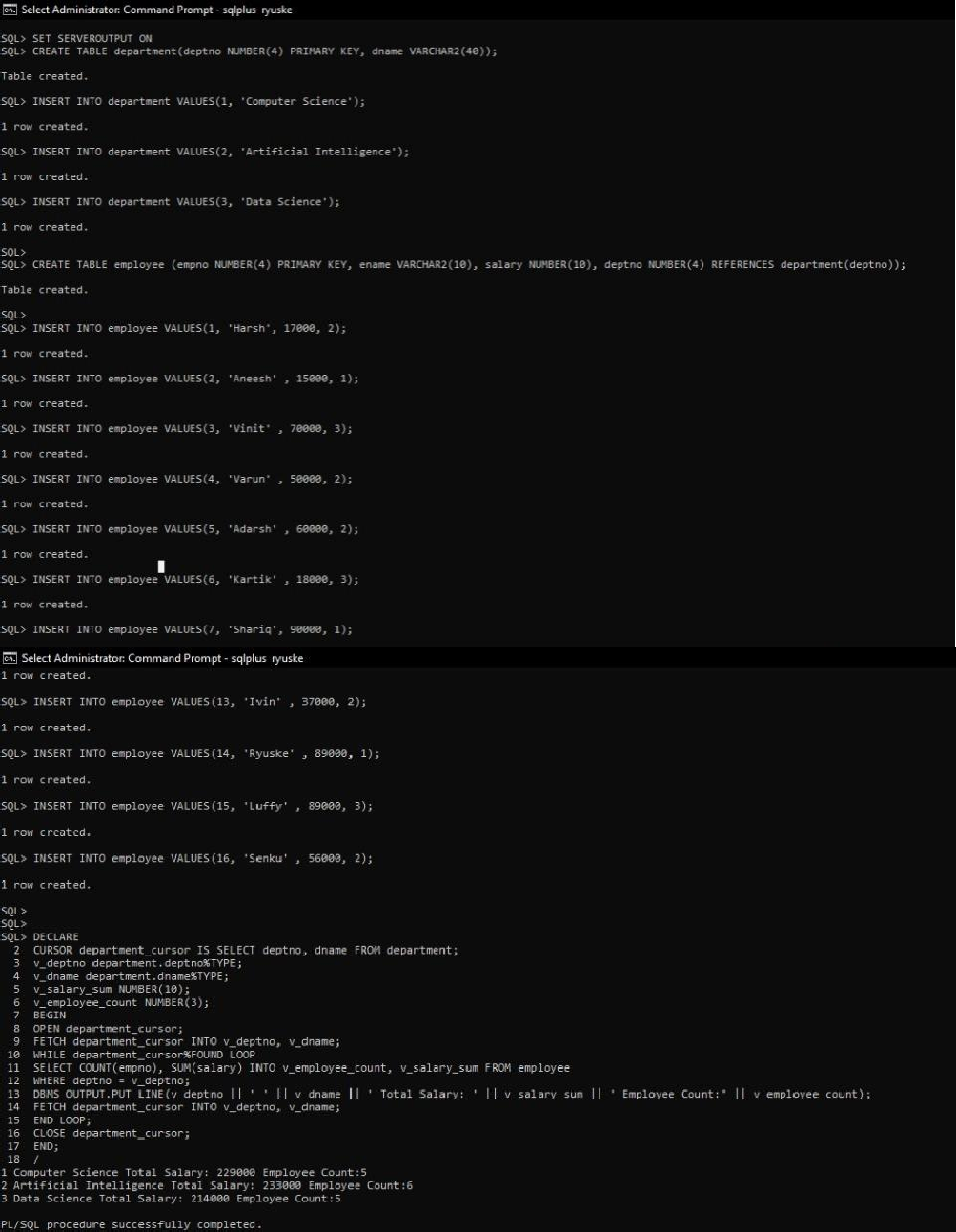
SELECT COUNT(empno), SUM(salary) INTO v\_employee\_count, v\_salary\_sum FROM employee WHERE deptno = v\_department\_row.deptno; DBMS\_OUTPUT.PUT\_LINE(v\_department\_row.deptno || ' ' || v\_department\_row.dname || ' Total Salary: ' || v\_salary\_sum || ' Employee Count:' || v\_employee\_count);

END LOOP;

END;

/

# Let’s implement the code and check the output:



**Practical 7**

) c’ Aim :- Writing PL/SQL Blocks with basic programming constructs by including a GoTO to jump out of a loop and NULL as a statement inside IF Question : - Write a PL/SQL block to print employees name and number excluding empno 10 and 15 along their department name. [USE : Goto statement] Table employee { empno number(5) primary key , ename varchar2(15), deptno number(4) reference key } Department {deptno number(4) primary key

, dname varchar2(4)}

SET SERVEROUTPUT ON

CREATE TABLE department(deptno NUMBER(4) PRIMARY KEY, dname VARCHAR2(40)); INSERT INTO department VALUES(1, 'Computer Science');

INSERT INTO department VALUES(2, 'Artificial Intelligence'); INSERT INTO department VALUES(3, 'Data Science');

CREATE TABLE employee (empno NUMBER(4) PRIMARY KEY, ename VARCHAR2(10), salary NUMBER (10), deptno NUMBER(4) REFERENCES department (deptno));

INSERT INTO employee VALUES(1, 'Harsh', 15000, 2); INSERT INTO employee VALUES(2, 'Vinit' , 15000, 3); INSERT INTO employee VALUES(3, 'Aneesh' , 20000, 2);

INSERT INTO employee VALUES(4, 'Snowsan' , 10000, 2); INSERT INTO employee VALUES(5, 'Nayan' , 10000, 2);

INSERT INTO employee VALUES(6, 'Saishyam' , 16000, 3); INSERT INTO employee VALUES(7, 'Pranav', 20000, 1); INSERT INTO employee VALUES(8, 'Varun' , 30000, 3); INSERT INTO employee VALUES(9, 'Yash' , 11000, 1); INSERT INTO employee VALUES(10, 'Umesh' , 21000, 1); INSERT INTO employee VALUES(11, 'Tarun' , 15000, 2);

INSERT INTO employee VALUES(12, 'Gayathri' , 15000, 3); INSERT INTO employee VALUES(13, 'Aryan' , 37000, 2);

INSERT INTO employee VALUES(14, 'Siddhant' , 89000, 1); INSERT INTO employee VALUES(15, 'Ryuske' , 89000, 3);

INSERT INTO employee VALUES(16, 'Zangetsu' , 56000, 2);

SET SERVEROUTPUT ON DECLARE

CURSOR employee\_cursor IS SELECT \* from employee; v\_employee\_row employee%ROWTYPE;

v\_dname department.dname%TYPE; BEGIN

FOR v\_employee\_row IN employee\_cursor LOOP

IF v\_employee\_row.empno = 10 OR v\_employee\_row.empno = 15 THEN GOTO skip;

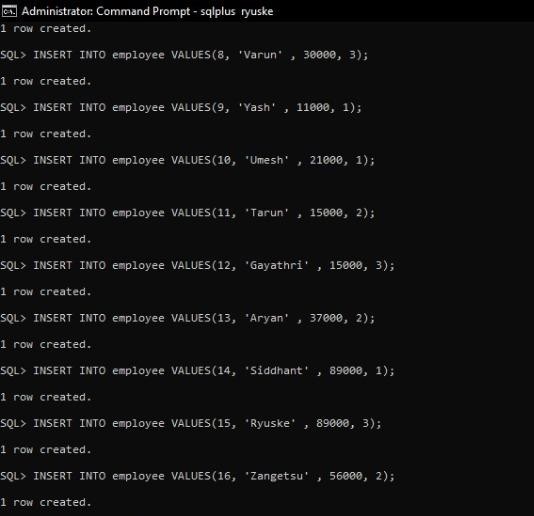
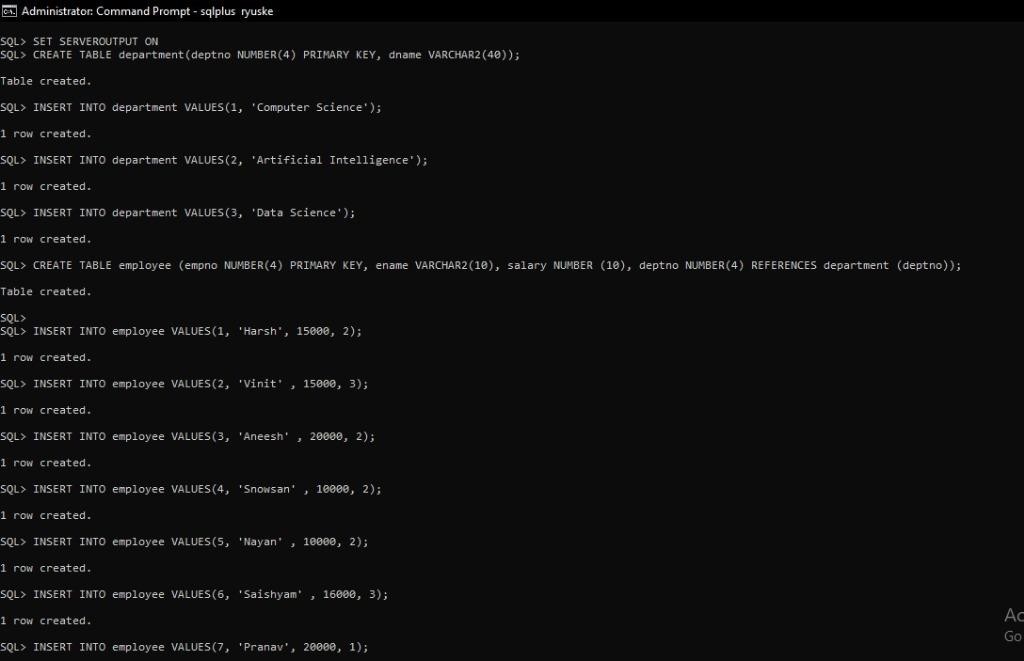
END IF;

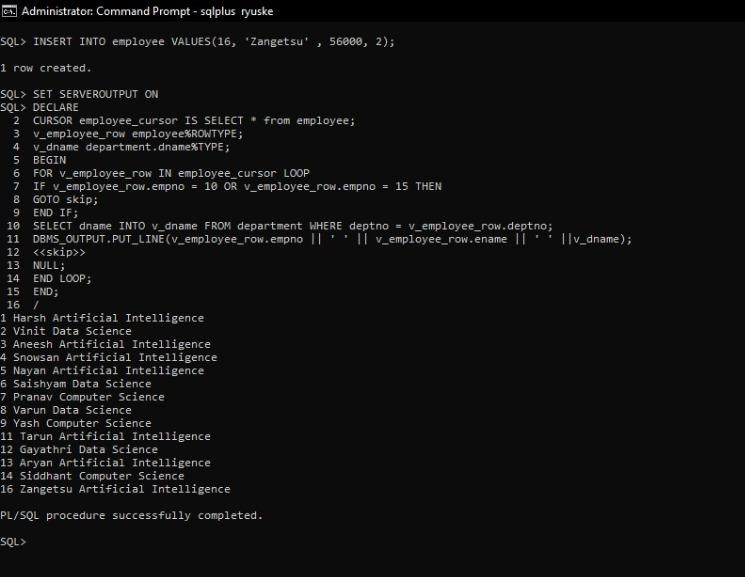
SELECT dname INTO v\_dname FROM department WHERE deptno = v\_employee\_row.deptno; DBMS\_OUTPUT.PUT\_LINE(v\_employee\_row.empno || ' ' || v\_employee\_row.ename || ' ' ||v\_dname);

<<skip>> NULL; END LOOP; END;

/

**Let’s Implement the code:**





# Practical 8:

)’c **Aim :**- Writing Procedures in PL/SQL Block a. Create an empty procedure, replace a procedure and call procedure b. Create a stored procedure and call it c. Define procedure to insert data Question

: - The HRD manager has decided to rise the salary of employee by 0.15. write PL/SQL block to accept the employee number and update the salary of that employee through procedure. Display appropriate message based on the existence of the record in employee table. Table : employee { empno number(5) primary key , ename varchar2(15) }

Question: Write PL/SQL procedure to accept the employee number and display employees name along salary and his department name. Table employee { empno number(5) primary key , ename varchar2(15), deptno number(4) reference key } Department { deptno number

(4) primary key , dname varchar2(4) }

# Code:

CREATE TABLE department(deptno NUMBER(4) PRIMARY KEY, dname VARCHAR2(40)); INSERT INTO department VALUES(1, 'Computer Science');

INSERT INTO department VALUES(2, 'Artificial Intelligence'); INSERT INTO department VALUES(3, 'Data Science');

CREATE TABLE employee (empno NUMBER(4) PRIMARY KEY, ename VARCHAR2(10), salary NUMBER (10), deptno NUMBER(4) REFERENCES department(deptno));

INSERT INTO employee VALUES(1, 'Harsh', 15000, 2); INSERT INTO employee VALUES(2, 'Varun' , 15000, 1); INSERT INTO employee VALUES(3, 'Vinit' , 20000, 3); INSERT INTO employee VALUES(4, 'Luffy' , 10000, 2);

INSERT INTO employee VALUES(5, 'Zangetsu' , 10000, 2); INSERT INTO employee VALUES(6, 'Starfus' , 16000, 3); INSERT INTO employee VALUES(7, 'Vegil', 20000, 1);

INSERT INTO employee VALUES(8, 'Manoj sir' , 30000, 3); INSERT INTO employee VALUES(9, 'Dragon' , 11000, 1); INSERT INTO employee VALUES(10, 'Naruto' , 21000, 1); INSERT INTO employee VALUES(11, 'Katze' , 15000, 2);

INSERT INTO employee VALUES(12, 'Gayathri' , 15000, 3); INSERT INTO employee VALUES(13, 'Kamal' , 37000, 2); INSERT INTO employee VALUES(14, 'Harish' , 89000, 1); INSERT INTO employee VALUES(15, 'Hitesh' , 89000, 3); INSERT INTO employee VALUES(16, 'Ajit' , 56000, 2);

SET SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE increase\_salary (p\_empno employee.empno%TYPE) IS v\_salary employee.salary%TYPE;

v\_ename employee.ename%TYPE; BEGIN

SELECT ename, salary INTO v\_ename, v\_salary FROM employee WHERE empno = p\_empno; UPDATE employee SET salary = v\_salary\*1.15 WHERE empno = p\_empno; DBMS\_OUTPUT.PUT\_LINE('Incremented Salary for ' || v\_ename || ' from ' || v\_salary || ' to ' || v\_salary \* 1.15);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Record Not Found'); END;

/

DECLARE

v\_empno employee.empno%TYPE := &v\_empno; BEGIN

increase\_salary(v\_empno); END;

/

CREATE OR REPLACE PROCEDURE display\_employee (p\_empno employee.empno%TYPE) IS v\_ename employee.ename%TYPE;

v\_salary employee.salary%TYPE; v\_deptno employee.deptno%TYPE; v\_dname department.dname%TYPE; BEGIN

SELECT ename, salary, deptno INTO v\_ename, v\_salary, v\_deptno FROM employee WHERE empno

= p\_empno;

SELECT dname INTO v\_dname FROM department WHERE deptno = v\_deptno; DBMS\_OUTPUT.PUT\_LINE(v\_ename || ' ' || v\_salary || ' ' || v\_dname); EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Record Not Found'); END;

/

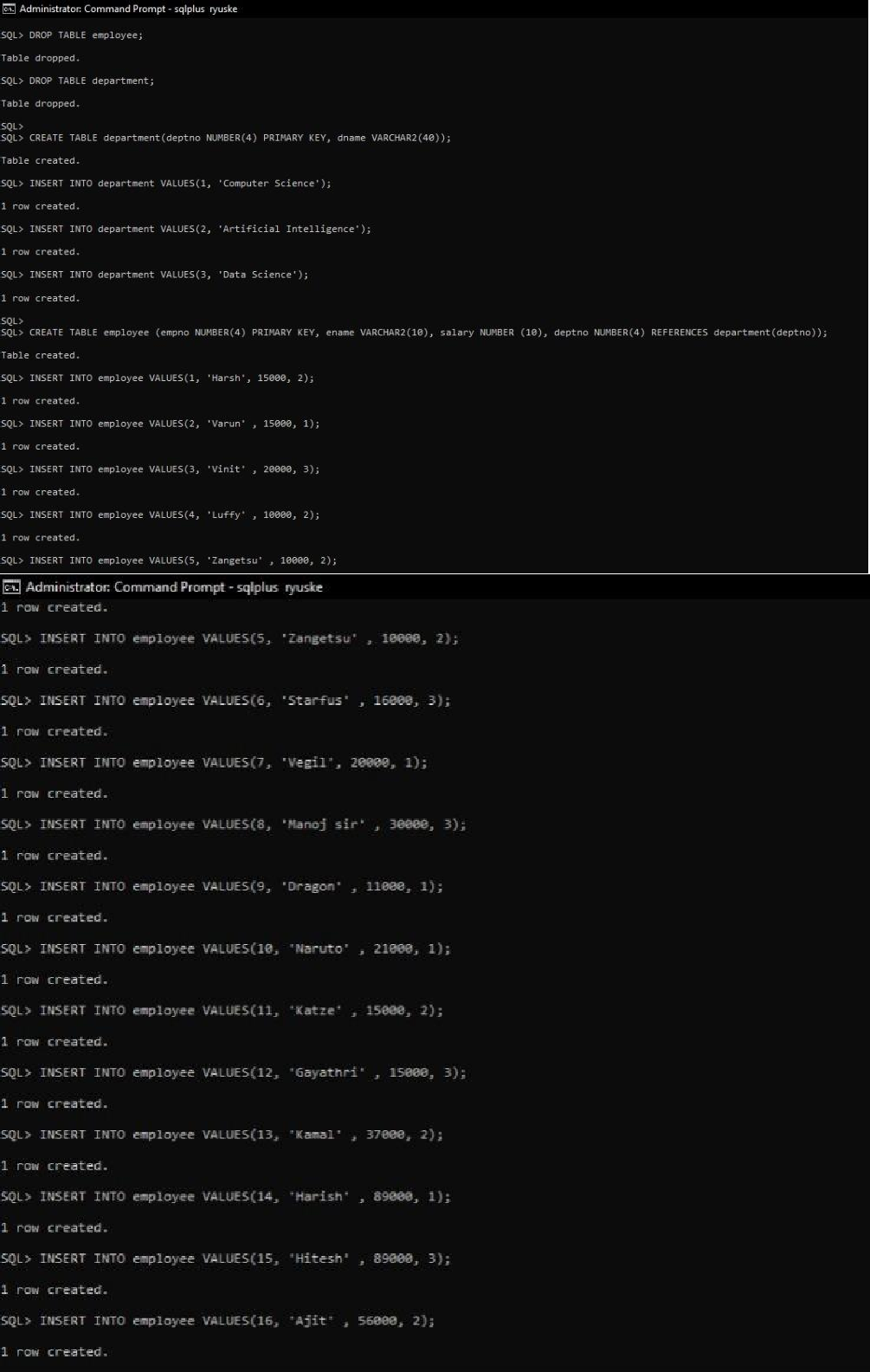
DECLARE

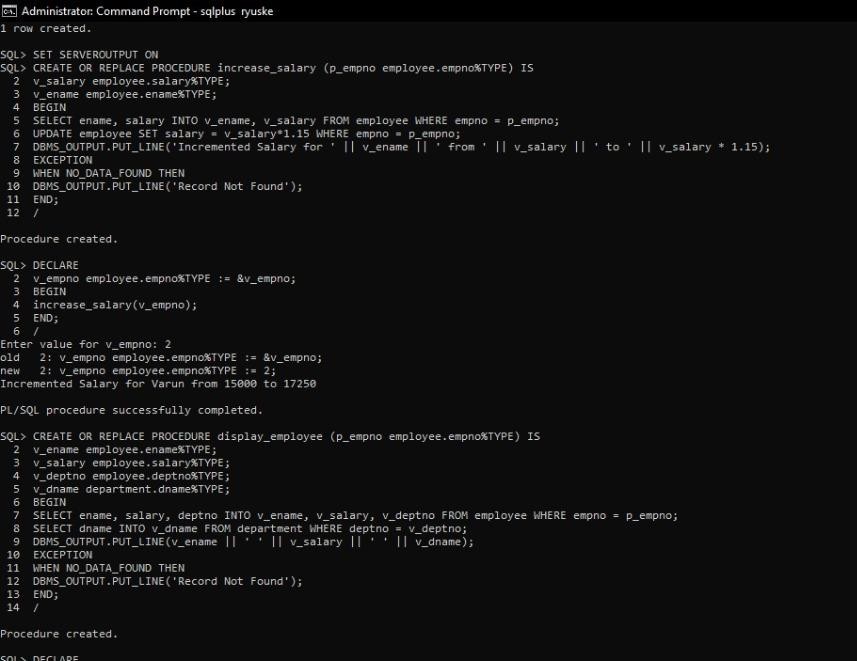
v\_empno employee.empno%TYPE := &v\_empno; BEGIN

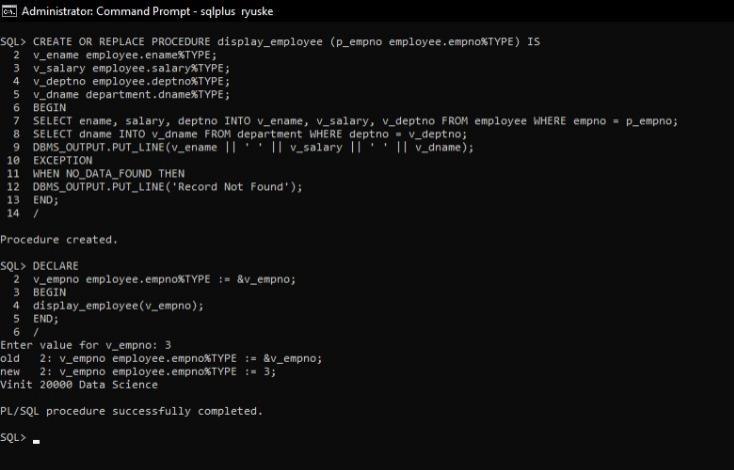
display\_employee(v\_empno); END;

/

**Let’s check the output:**







# Practical 9

)’c Aim :- Writing Functions in PL/SQL Block. Question : -Write PL/SQL function to accept department id and display total number of employees working in that department along the maximum salary of that department. Table employee { empno number(5) primary key , ename varchar2(15), deptno number(4) reference key } Department { deptno number

(4) primary key , dname varchar2(4) } Writing a recursive Functions in PL/SQL Block Question : Write PL/SQL recursive function to list top ten employees name along their salary. Table employee { empno number(5) primary key , ename varchar2(15) }

# Code:

SET SERVEROUTPUT ON

CREATE TABLE department(deptno NUMBER(4) PRIMARY KEY, dname VARCHAR2(40));

INSERT INTO department VALUES(1, 'Computer Science'); INSERT INTO department VALUES(2, 'Artificial Intelligence'); INSERT INTO department VALUES(3, 'Data Science');

CREATE TABLE employee (empno NUMBER(4) PRIMARY KEY, ename VARCHAR2(10), salary NUMBER (10), deptno NUMBER(4) REFERENCES

department(deptno));

INSERT INTO employee VALUES(1, 'Harsh', 15000, 2); INSERT INTO employee VALUES(2, 'Vinit' , 15000, 3); INSERT INTO employee VALUES(3, 'Aneesh' , 20000, 2); INSERT INTO employee VALUES(4, 'Varun' , 10000, 2); INSERT INTO employee VALUES(5, 'Ramesh' , 10000, 2); INSERT INTO employee VALUES(6, 'Naagin' , 16000, 3);

INSERT INTO employee VALUES(7, 'Abuzar sir', 20000, 1); INSERT INTO employee VALUES(8, 'Ryuske' , 30000, 3);

INSERT INTO employee VALUES(9, 'Zangetsu' , 11000, 1); INSERT INTO employee VALUES(10, 'Vikram' , 21000, 1); INSERT INTO employee VALUES(11, 'KGF' , 15000, 2);

INSERT INTO employee VALUES(12, 'Gayathri' , 15000, 3); INSERT INTO employee VALUES(13, 'Kamal' , 37000, 2); INSERT INTO employee VALUES(14, 'Nitish' , 89000, 1); INSERT INTO employee VALUES(15, 'Hitesh' , 89000, 3); INSERT INTO employee VALUES(16, 'Ajit' , 56000, 2);

SET SERVEROUTPUT ON

CREATE OR REPLACE FUNCTION practical9a\_function(p\_deptno department.deptno%TYPE)

RETURN VARCHAR2 IS

v\_employee\_count NUMBER(2); v\_max\_salary employee.salary%TYPE; BEGIN

SELECT COUNT(empno), MAX(salary) INTO v\_employee\_count, v\_max\_salary FROM employee

WHERE deptno = p\_deptno;

IF v\_employee\_count = 0 THEN RETURN 'No Records';

ELSE

RETURN 'Employee Count: ' || v\_employee\_count || ' Max Salary: ' || v\_max\_salary; END IF;

END;

/

DECLARE

v\_deptno department.deptno%TYPE := &v\_deptno; BEGIN

DBMS\_OUTPUT.PUT\_LINE(practical9a\_function(v\_deptno)); END;

/

CREATE OR REPLACE FUNCTION practical9b\_function(counter NUMBER) RETURN VARCHAR2

IS

v\_ename employee.ename%TYPE; v\_salary employee.salary%TYPE; BEGIN

IF counter = 11 THEN RETURN 'Completed'; END IF;

SELECT ename, salary INTO v\_ename, v\_salary FROM employee WHERE empno = counter;

DBMS\_OUTPUT.PUT\_LINE(v\_ename || ' ' || v\_salary); RETURN practical9b\_function(counter + 1);

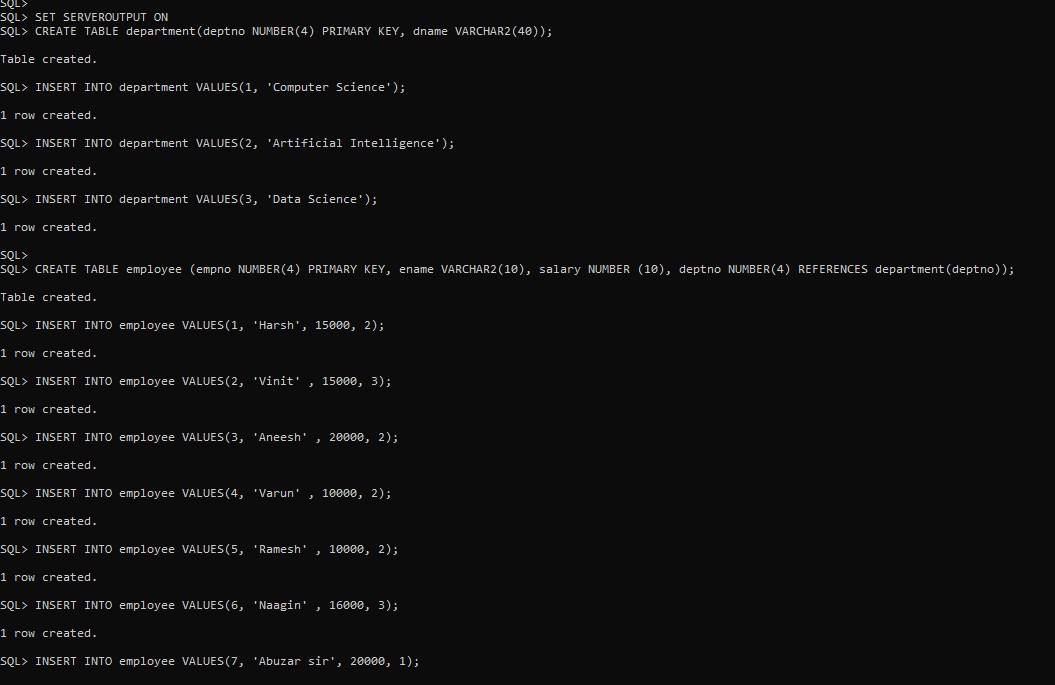
END;

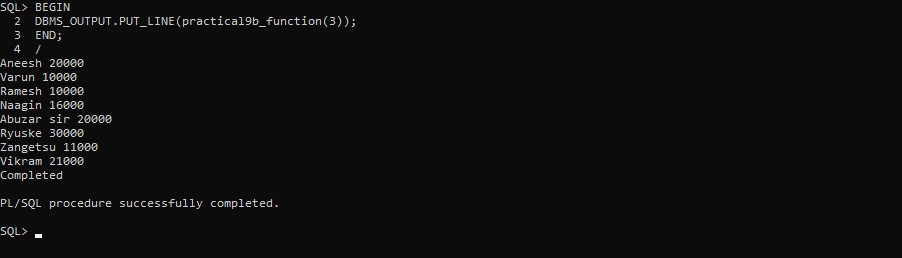
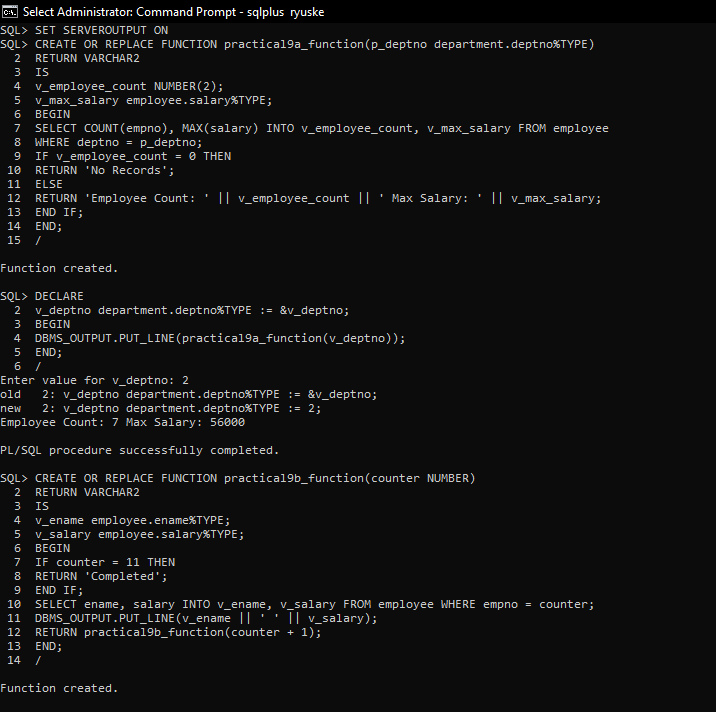
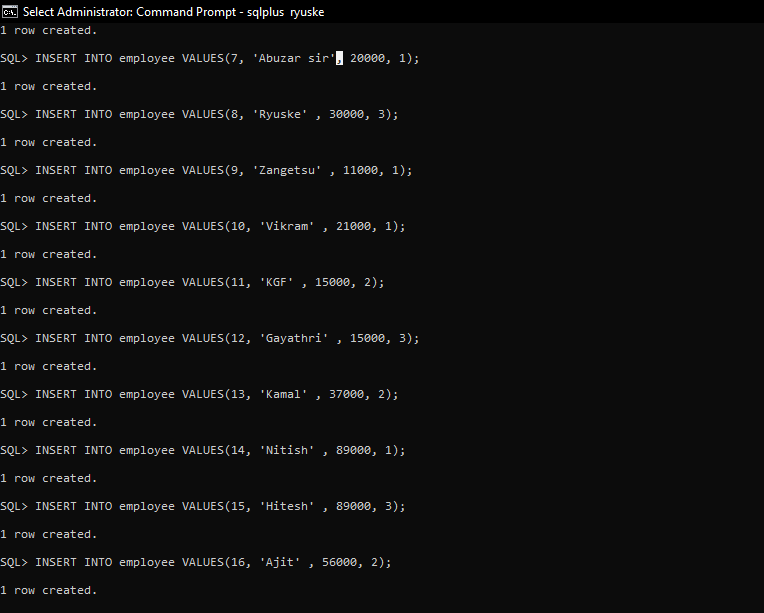
/ BEGIN

DBMS\_OUTPUT.PUT\_LINE(practical9b\_function(3)); END;

/

# Let’s check output with our codes:





**Practical** º⬛ c’ ) Aim :- Write a PL/SQL block to auto insert 10 records in an employee table. Keep updating salary of Blake and Clark by Rs.2000 and Rs.1500 respectively. As long as total salary does not exceed to 20000. As soon as total salary reaches to more than 20000 then undo the Last salary updates of Blake and Clark

**Code:**

CREATE TABLE EMP4 (EMPNO NUMBER(5) PRIMARY KEY , ENAME VARCHAR2(15), SALARY NUMBER(10));

INSERT INTO EMP4 VALUES(001,’Blake’,900); INSERT INTO EMP4 VALUES(002,’Clark’,1000); INSERT INTO EMP4 VALUES(003,’Aneesh’,200); INSERT INTO EMP4 VALUES(004,’Harsh’,250); INSERT INTO EMP4 VALUES(005,’Varun’,350); INSERT INTO EMP4 VALUES(006,’Tarun’,300); INSERT INTO EMP4 VALUES(007,’Ankita’,500); INSERT INTO EMP4 VALUES(008,’Shri Hari’,550); INSERT INTO EMP4 VALUES(009,’Kartik’,600); INSERT INTO EMP4 VALUES(010,’Ryuske’,650);

DECLARE

TOTAL\_SAL NUMBER(9); BEGIN

INSERT INTO EMP4 VALUES(011,'DAVID',1000); SAVEPOINT NO\_UPDATE;

UPDATE EMP4 SET SALARY = SALARY+2000 WHERE ENAME = 'BLAKE'; UPDATE EMP4 SET SALARY = SALARY + 1500 WHERE ENAME = 'CLARK'; SELECT SUM(SALARY) INTO TOTAL\_SAL FROM EMP4;

IF TOTAL\_SAL>2000 THEN

ROLLBACK TO SAVEPOINT NO\_UPDATE; END IF;

COMMIT;

END;

/

**Let’s check the code:**

