# A close-up of a logo Description automatically generated

# Name: Apurba Koirala

# Reg no: 22BCE3799

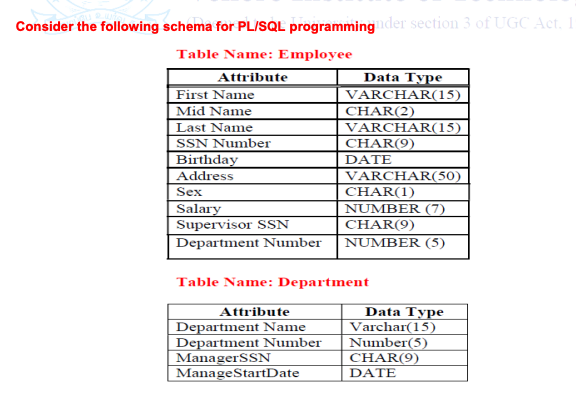
# Subject Code: BCSE302P

# Course Title: Database Systems

# Lab Slot: L33 + L34

# Guided by: Dr. Shashank Mouli Satapathy

**Exercise - 1**



Constructing Schema :

A computer screen shot of a code

Description automatically generated

A computer code on a black background

Description automatically generated A computer screen with white text

Description automatically generated

A white text with black text

Description automatically generated with medium confidence

1.

Code:

DECLARE v\_empno NUMBER := 2; v\_salary NUMBER; BEGIN

SELECT salary INTO v\_salary

FROM Employee

WHERE empno = v\_empno;

DBMS\_OUTPUT.PUT\_LINE('Salary: $' || v\_salary);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Employee not found.');

END;

Output:

A computer screen with white text

Description automatically generated

2.

Code:

DECLARE v\_empno NUMBER := 3;

BEGIN

DELETE FROM Employee

WHERE empno = v\_empno;

IF SQL%ROWCOUNT > 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Employee deleted successfully.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Employee not found.');

END IF;

END;

/

Output:

A computer screen with white text

Description automatically generated

3.

Code:

DECLARE

v\_current\_date DATE := SYSDATE;

BEGIN

DELETE FROM Employee

WHERE months\_between(v\_current\_date, birthday) / 12 > 60;

DBMS\_OUTPUT.PUT\_LINE('Employees with age >60 deleted successfully.');

END;

/

A computer screen with white text

Description automatically generated

4.

Code:

BEGIN

FOR emp\_rec IN (SELECT \*

FROM Employee

WHERE salary < 1000)

LOOP

DBMS\_OUTPUT.PUT\_LINE('Employee: ' || emp\_rec.first\_name || ' ' || emp\_rec.last\_name);

END LOOP;

END;

/

Output:

A computer screen with white text

Description automatically generated

5.

Code:

BEGIN

DELETE FROM Employee

WHERE salary < 2000;

DBMS\_OUTPUT.PUT\_LINE('Records deleted successfully.');

END;

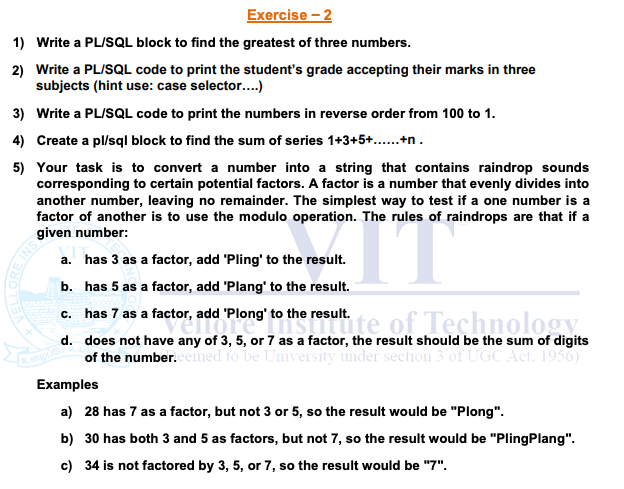
/

Output:

A computer screen with white text

Description automatically generated

**Exercise – 2**

****

1.

Code:

DECLARE num1 NUMBER := 10; num2 NUMBER := 5; num3 NUMBER := 8; greatest NUMBER;

BEGIN

IF num1 >= num2 AND num1 >= num3 THEN greatest := num1;

ELSIF num2 >= num1 AND num2 >= num3 THEN greatest := num2; ELSE greatest := num3;

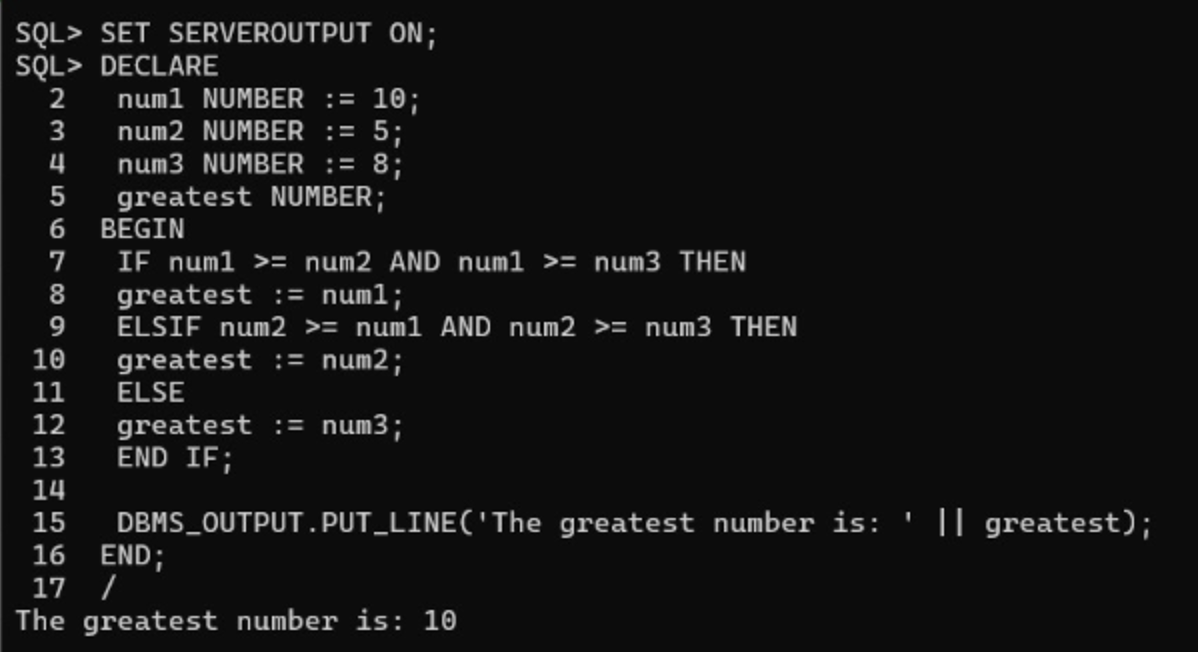
END IF;

DBMS\_OUTPUT.PUT\_LINE('The greatest number is: ' || greatest);

END;

/

Output:



2.

Code:

DECLARE marks1 NUMBER := 80; marks2 NUMBER := 75; marks3 NUMBER := 90; average NUMBER; grade VARCHAR2(2); BEGIN

average := (marks1 + marks2 + marks3) / 3;

CASE

WHEN average >= 90 THEN grade := 'A';

WHEN average >= 80 THEN grade := 'B';

WHEN average >= 70 THEN grade := 'C';

WHEN average >= 60 THEN grade := 'D';

ELSE grade := 'F';

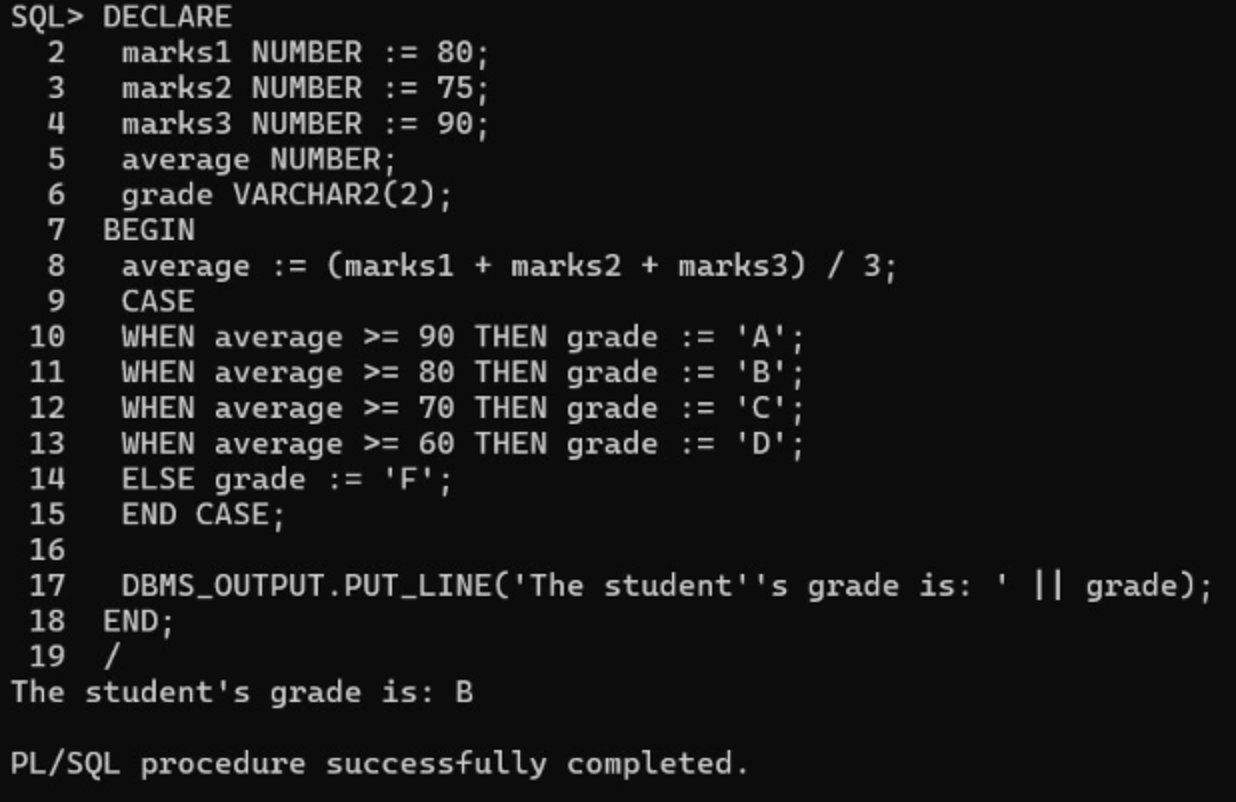
END CASE;

DBMS\_OUTPUT.PUT\_LINE('The student''s grade is: ' || grade);

END;

/

Output:



3.

Code:

DECLARE

counter NUMBER := 100;

BEGIN

WHILE counter >= 1 LOOP

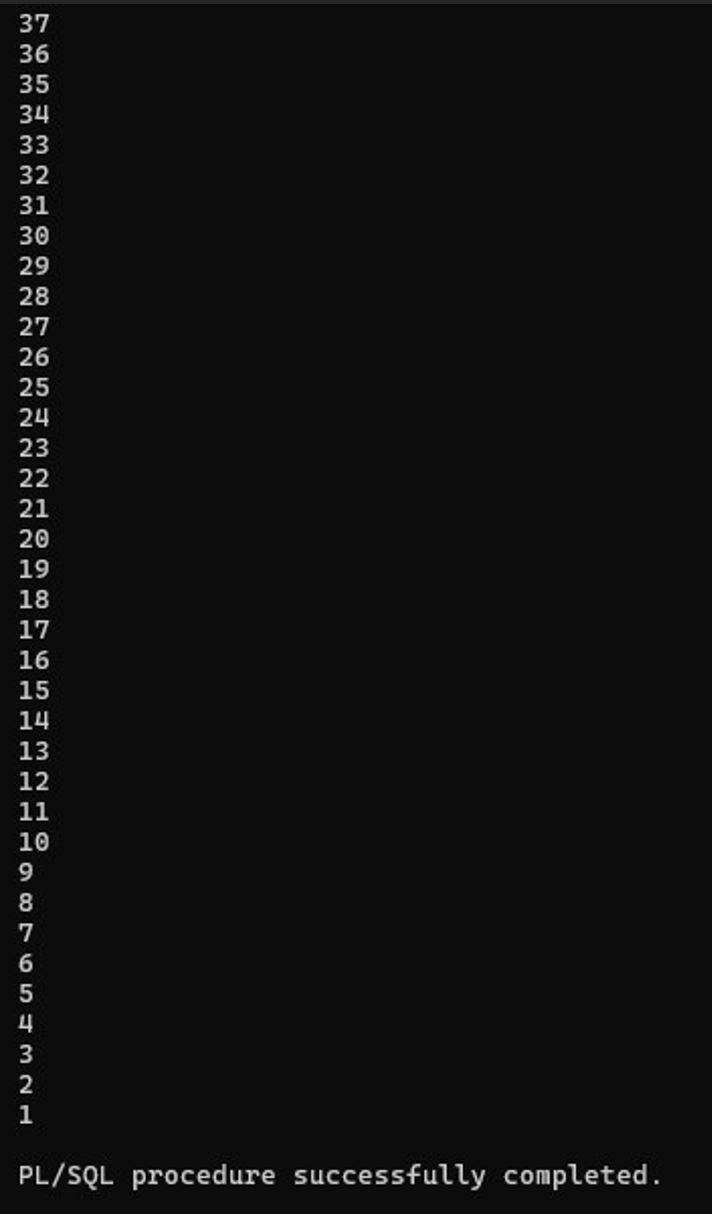
DBMS\_OUTPUT.PUT\_LINE(counter); counter := counter - 1;

END LOOP;

END;

/





4.

Code:

DECLARE

n NUMBER := 40; sum\_odd NUMBER := 0; num NUMBER := 1; BEGIN

WHILE num <= n LOOP sum\_odd := sum\_odd + num; num := num + 2;

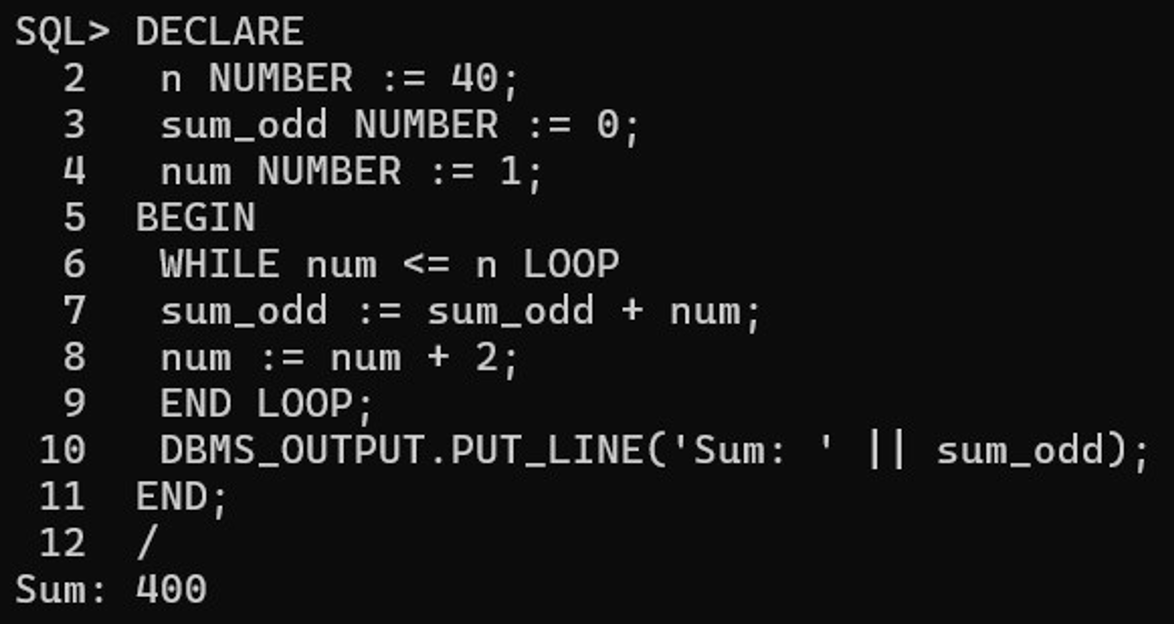
END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Sum: ' || sum\_odd);

END;

/

Output:



5.

Code:

DECLARE number\_to\_convert NUMBER := 28; result VARCHAR2(100) := '';

BEGIN

IF MOD(number\_to\_convert, 3) = 0 THEN

result := result || 'Pling';

END IF;

IF MOD(number\_to\_convert, 5) = 0 THEN

result := result || 'Plang';

END IF;

IF MOD(number\_to\_convert, 7) = 0 THEN

result := result || 'Plong';

END IF;

IF result = '' THEN

result := TO\_CHAR(number\_to\_convert);

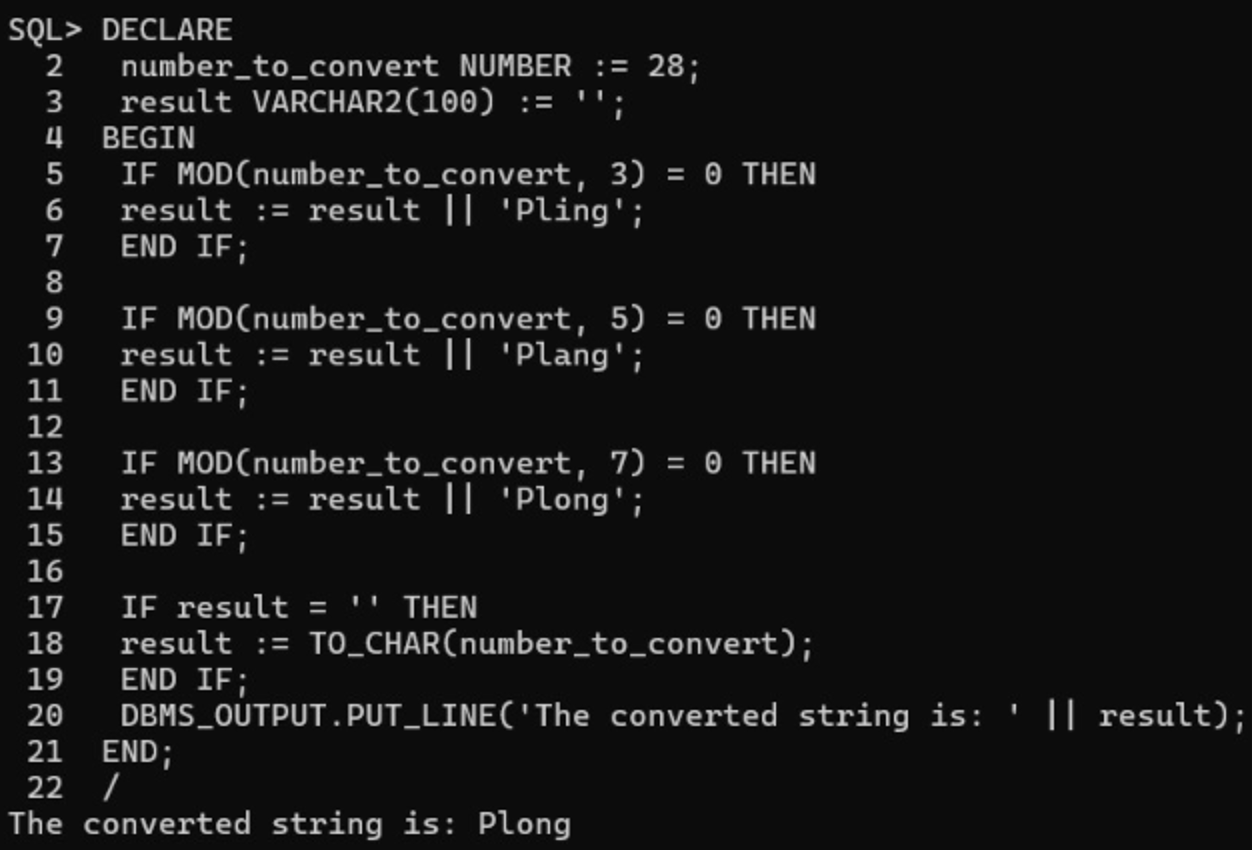
END IF;

DBMS\_OUTPUT.PUT\_LINE('The converted string is: ' || result);

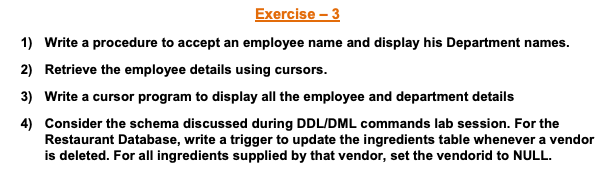
END;

/

Output:



**Exercise – 3**



1.

Code:

CREATE OR REPLACE PROCEDURE GetEmployeeDepartments ( p\_first\_name IN Employee.first\_name%TYPE, p\_last\_name IN Employee.last\_name%TYPE

)

IS

BEGIN

FOR dept\_rec IN (

SELECT d.department\_name

FROM Employee e

JOIN Department d ON e.department\_number = d.department\_number

WHERE e.first\_name = p\_first\_name

AND e.last\_name = p\_last\_name

) LOOP

DBMS\_OUTPUT.PUT\_LINE('Department: ' || dept\_rec.department\_name);

END LOOP;

EXCEPTION

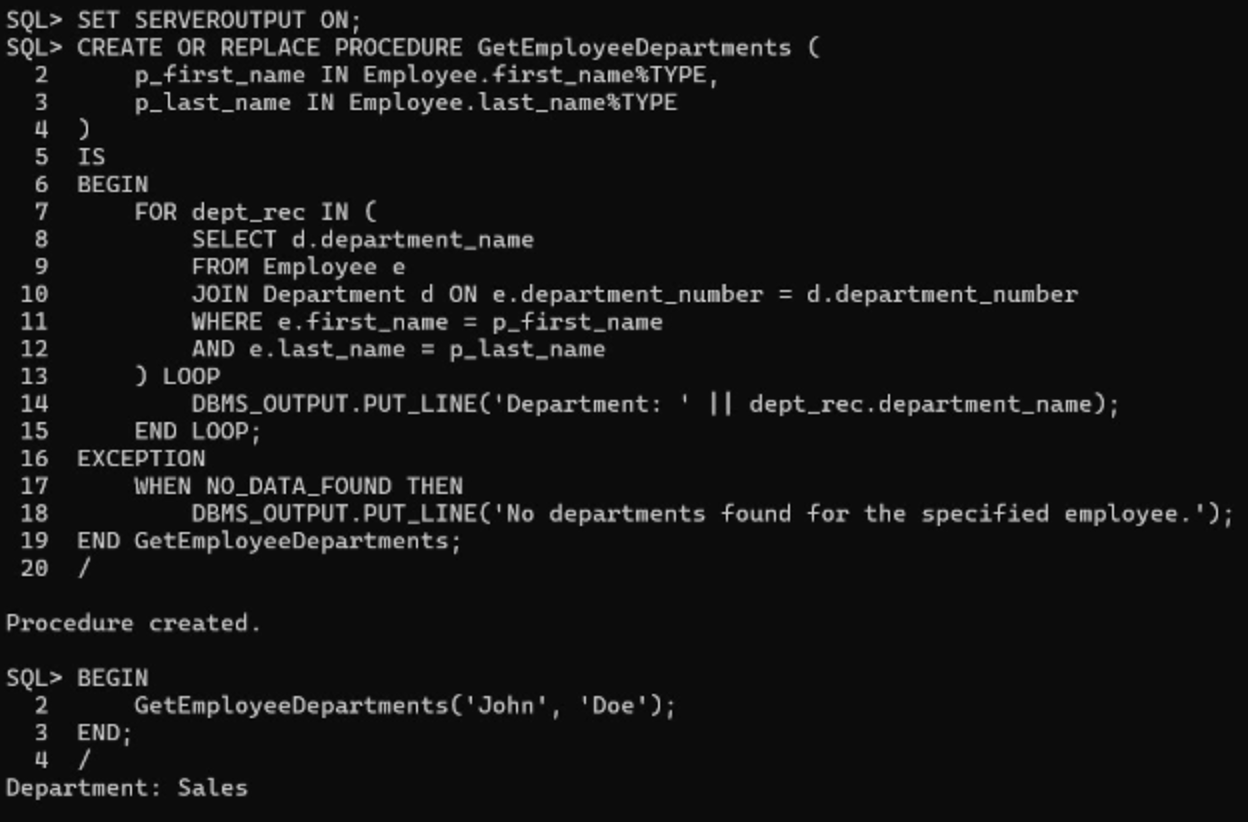
WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('No departments found for the specified employee.');

END GetEmployeeDepartments;

/

Output:



2.

Code:

DECLARE

CURSOR emp\_cursor IS

SELECT empno, first\_name, mid\_name, last\_name, ssn\_number, birthday, address, sex, salary, supervisor\_ssn, department\_number

FROM Employee;

v\_empno Employee.empno%TYPE; v\_first\_name Employee.first\_name%TYPE; v\_mid\_name Employee.mid\_name%TYPE; v\_last\_name Employee.last\_name%TYPE; v\_ssn\_number Employee.ssn\_number%TYPE; v\_birthday Employee.birthday%TYPE; v\_address Employee.address%TYPE; v\_sex Employee.sex%TYPE; v\_salary Employee.salary%TYPE; v\_supervisor\_ssn Employee.supervisor\_ssn%TYPE; v\_department\_number Employee.department\_number%TYPE;

BEGIN

OPEN emp\_cursor;

LOOP

FETCH emp\_cursor INTO v\_empno, v\_first\_name, v\_mid\_name, v\_last\_name, v\_ssn\_number, v\_birthday, v\_address, v\_sex, v\_salary, v\_supervisor\_ssn, v\_department\_number;

EXIT WHEN emp\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Employee No: ' || v\_empno);

DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_first\_name || ' ' || v\_mid\_name || ' ' || v\_last\_name);

DBMS\_OUTPUT.PUT\_LINE('SSN: ' || v\_ssn\_number);

DBMS\_OUTPUT.PUT\_LINE('Birthday: ' || TO\_CHAR(v\_birthday, 'YYYY-MM-DD'));

DBMS\_OUTPUT.PUT\_LINE('Address: ' || v\_address);

DBMS\_OUTPUT.PUT\_LINE('Sex: ' || v\_sex);

DBMS\_OUTPUT.PUT\_LINE('Salary: $' || v\_salary);

DBMS\_OUTPUT.PUT\_LINE('Supervisor SSN: ' || v\_supervisor\_ssn); DBMS\_OUTPUT.PUT\_LINE('Department Number: ' || v\_department\_number);

DBMS\_OUTPUT.PUT\_LINE('-----------------------------');

END LOOP;

CLOSE emp\_cursor;

END;

/

Output:



3.

Code:

DECLARE

CURSOR emp\_dept\_cursor IS

SELECT e.empno, e.first\_name, e.mid\_name, e.last\_name, e.ssn\_number, e.birthday, e.address, e.sex, e.salary,

e.supervisor\_ssn, e.department\_number, d.department\_name, d.manager\_ssn, d.manager\_start\_date FROM Employee e

JOIN Department d ON e.department\_number = d.department\_number;

v\_empno Employee.empno%TYPE; v\_first\_name Employee.first\_name%TYPE; v\_mid\_name Employee.mid\_name%TYPE; v\_last\_name Employee.last\_name%TYPE; v\_ssn\_number Employee.ssn\_number%TYPE; v\_birthday Employee.birthday%TYPE; v\_address Employee.address%TYPE; v\_sex Employee.sex%TYPE; v\_salary Employee.salary%TYPE; v\_supervisor\_ssn Employee.supervisor\_ssn%TYPE; v\_department\_number Employee.department\_number%TYPE; v\_department\_name Department.department\_name%TYPE; v\_manager\_ssn Department.manager\_ssn%TYPE; v\_manager\_start\_date Department.manager\_start\_date%TYPE;

BEGIN

OPEN emp\_dept\_cursor;

LOOP

FETCH emp\_dept\_cursor INTO v\_empno, v\_first\_name, v\_mid\_name, v\_last\_name, v\_ssn\_number, v\_birthday, v\_address, v\_sex, v\_salary, v\_supervisor\_ssn, v\_department\_number, v\_department\_name, v\_manager\_ssn, v\_manager\_start\_date;

EXIT WHEN emp\_dept\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Employee No: ' || v\_empno);

DBMS\_OUTPUT.PUT\_LINE('Name: ' || v\_first\_name || ' ' || v\_mid\_name || ' ' || v\_last\_name);

DBMS\_OUTPUT.PUT\_LINE('SSN: ' || v\_ssn\_number);

DBMS\_OUTPUT.PUT\_LINE('Birthday: ' || TO\_CHAR(v\_birthday, 'YYYY-MM-DD'));

DBMS\_OUTPUT.PUT\_LINE('Address: ' || v\_address);

DBMS\_OUTPUT.PUT\_LINE('Sex: ' || v\_sex);

DBMS\_OUTPUT.PUT\_LINE('Salary: $' || v\_salary);

DBMS\_OUTPUT.PUT\_LINE('Supervisor SSN: ' || v\_supervisor\_ssn);

DBMS\_OUTPUT.PUT\_LINE('Department Number: ' || v\_department\_number);

DBMS\_OUTPUT.PUT\_LINE('Department Name: ' || v\_department\_name);

DBMS\_OUTPUT.PUT\_LINE('Manager SSN: ' || v\_manager\_ssn);

DBMS\_OUTPUT.PUT\_LINE('Manager Start Date: ' || TO\_CHAR(v\_manager\_start\_date, 'YYYY-MM-DD'));

DBMS\_OUTPUT.PUT\_LINE('-----------------------------');

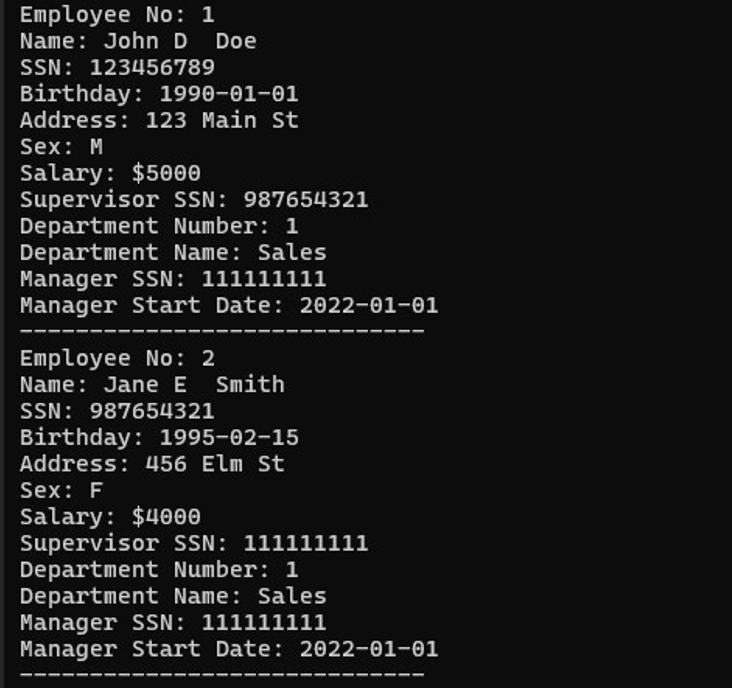
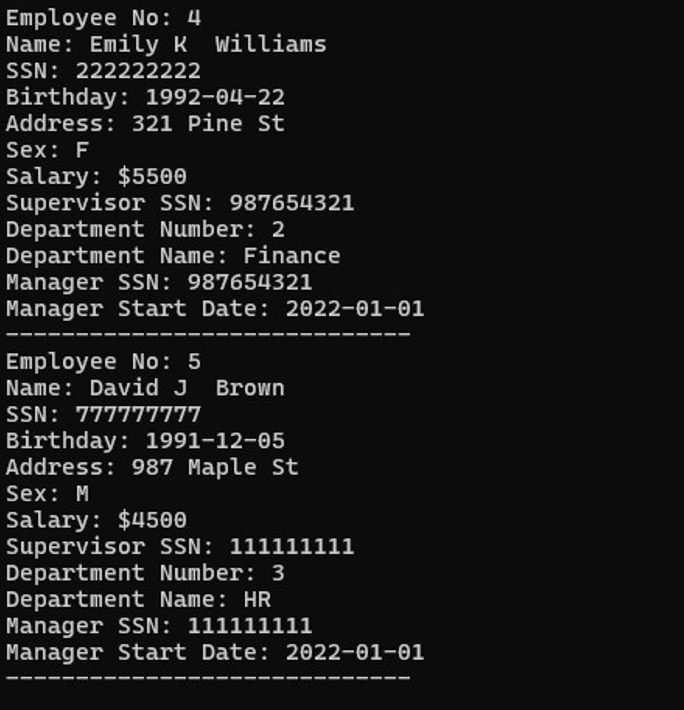
END LOOP;

CLOSE emp\_dept\_cursor;

END;

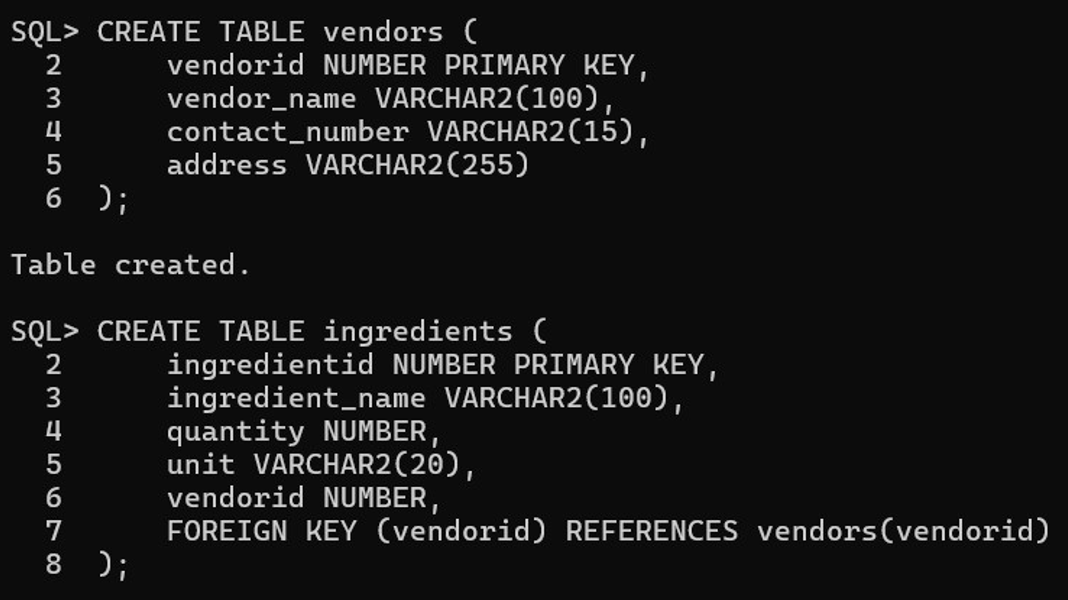
/

Output:

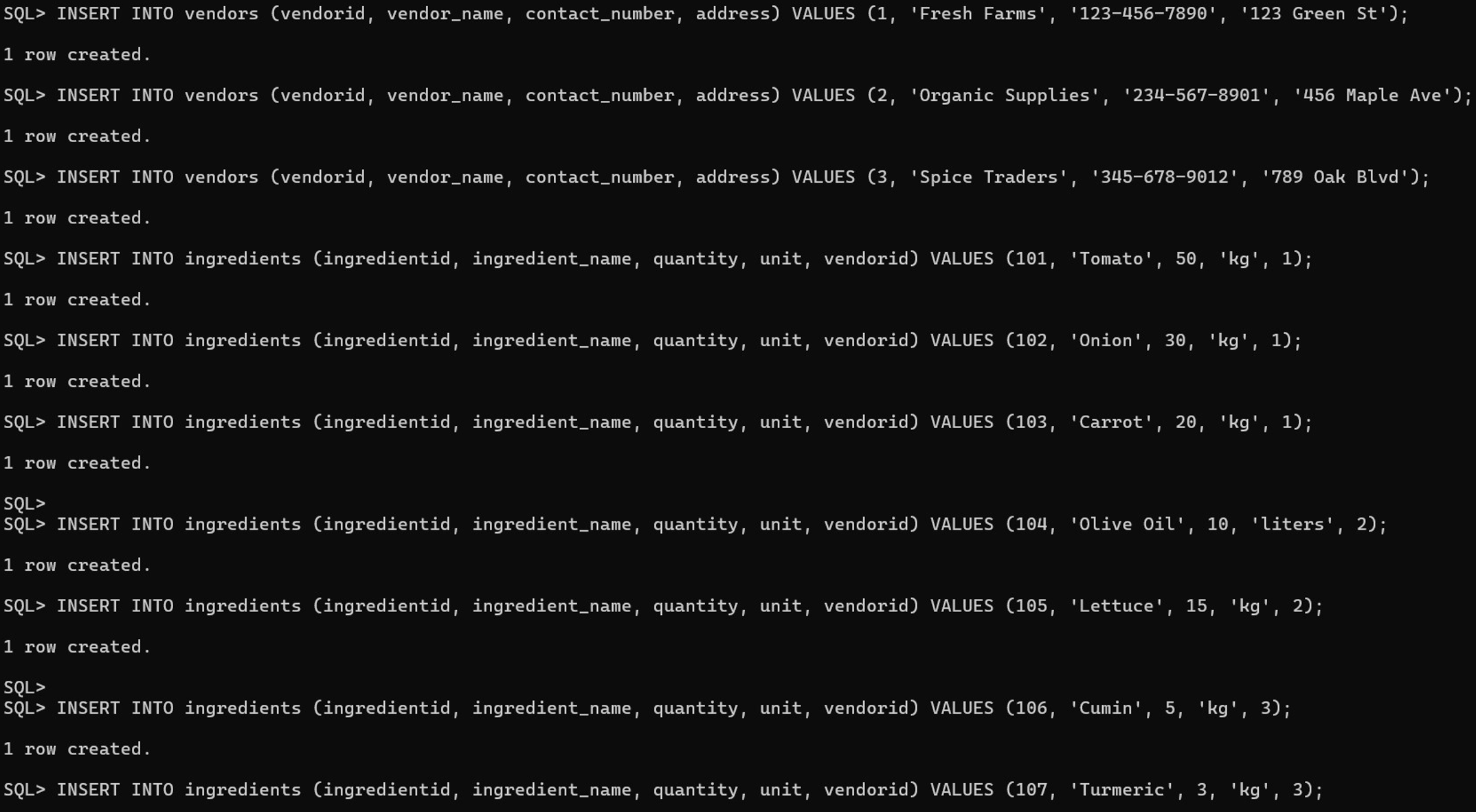
 

4.

For Creating Tables:



Inserting data:



Creating Trigger:

CREATE OR REPLACE TRIGGER update\_ingredients\_on\_vendor\_delete

AFTER DELETE ON vendors

FOR EACH ROW

BEGIN

UPDATE ingredients

SET vendorid = NULL

WHERE vendorid = :OLD.vendorid;

END;

/

Before deleting vendorid = 1 contents:



After deleting vendorid = 1 contents:

