**PROCEDURE:**

1. CREATE OR REPLACE PROCEDURE greetings

AS

BEGIN

dbms\_output.put\_line('Hello World!');

END;

1. **EXECUTE** greetings;
2. DROP PROCEDURE procedure-name;

**DROP** PROCEDURE greetings;

**PROCEDURE programs:**

**1.**

DECLARE

a number;

b number;

c number;

PROCEDURE findMin(x IN number, y IN number, z OUT number) IS

BEGIN

IF x < y THEN

z:= x;

ELSE

z:= y;

END IF;

END;

BEGIN

a:= 23;

b:= 45;

findMin(a, b, c);

dbms\_output.put\_line(' Minimum of (23, 45) : ' || c);

END;

/

2.

DECLARE

a number;

PROCEDURE squareNum(x IN OUT number) IS

BEGIN

x := x \* x;

END;

BEGIN

a:= 3;

squareNum(a);

dbms\_output.put\_line(' Square of (3): ' || a);

END;

/

**PACKAGES:**

1. Create a customers table

CREATE TABLE CUSTOMERS(

ID INT NOT NULL,

NAME VARCHAR (20) NOT NULL,

AGE INT NOT NULL,

ADDRESS CHAR (25),

SALARY DECIMAL (18, 2),

PRIMARY KEY (ID)

);

1. Insert values into customers table

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES (1, 'Ramesh', 32, 'Ahmedabad', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'Khilan', 25, 'Delhi', 1500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'kaushik', 23, 'Kota', 2000.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'Chaitali', 25, 'Mumbai', 6500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES (5, 'Hardik', 27, 'Bhopal', 8500.00 );

INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY) VALUES (6, 'Komal', 22, 'MP', 4500.00 );

1. Write a pl/sql program to view the details for a given customers by using **select INTO**

DECLARE

c\_id customers.id%type := 1;

c\_name customers.name%type;

c\_addr customers.address%type;

c\_sal customers.salary%type;

BEGIN

SELECT name, address, salary INTO c\_name, c\_addr, c\_sal

FROM customers

WHERE id = c\_id;

dbms\_output.put\_line

('Customer ' ||c\_name || ' from ' || c\_addr || ' earns ' || c\_sal);

END;

/

OUTPUT:

Customer Ramesh from Ahmedabad earns 2000

PL/SQL procedure completed successfully

1. Using **cursors** update few customers data by adding their salary 500.

DECLARE

total\_rows number(2);

BEGIN

UPDATE customers

SET salary = salary + 500;

IF sql%notfound THEN

dbms\_output.put\_line('no customers selected');

ELSIF sql%found THEN

total\_rows := sql%rowcount;

dbms\_output.put\_line( total\_rows || ' customers selected ');

END IF;

END;

/

OUTPUT:

6 customers selected

PL/SQL procedure successfully completed.

5. Using **Explicit** **cursors** View few customers data from a table.

DECLARE

c\_id customers.id%type;

c\_name customers.name%type;

c\_addr customers.address%type;

CURSOR c\_customers is

SELECT id, name, address FROM customers;

BEGIN

OPEN c\_customers;

LOOP

FETCH c\_customers into c\_id, c\_name, c\_addr;

EXIT WHEN c\_customers%notfound;

dbms\_output.put\_line(c\_id || ' ' || c\_name || ' ' || c\_addr);

END LOOP;

CLOSE c\_customers;

END;

/

OUTPUT:

1 Ramesh Ahmedabad

2 Khilan Delhi

3 kaushik Kota

4 Chaitali Mumbai

5 Hardik Bhopal

6 Komal MP

PL/SQL procedure successfully completed.

**EXCEPTION HANDLING:**

DECLARE

<declarations section>

BEGIN

<executable command(s)>

EXCEPTION

<exception handling goes here >

WHEN exception1 THEN

exception1-handling-statements

WHEN exception2 THEN

exception2-handling-statements

WHEN exception3 THEN

exception3-handling-statements

........

WHEN others THEN

exception3-handling-statements

END;

1. Write a pl/sql program to view customers data by using exception handling

DECLARE

c\_id customers.id%type := 8;

c\_name customerS.Name%type;

c\_addr customers.address%type;

BEGIN

**SELECT** name, address **INTO** c\_name, c\_addr  **FROM** customers  **WHERE** id = c\_id;

DBMS\_OUTPUT.PUT\_LINE ('Name: '|| c\_name);

DBMS\_OUTPUT.PUT\_LINE ('Address: ' || c\_addr);

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

**EXCEPTION HANDLING RAISE:**

DECLARE

exception\_name EXCEPTION;

BEGIN

IF condition THEN

RAISE exception\_name;

END IF;

EXCEPTION

WHEN exception\_name THEN

statement;

END;

1. Write a pl/sql program to view customers data by using exception handling RAISE

DECLARE

c\_id customers.id%type := &cc\_id;

c\_name customerS.Name%type;

c\_addr customers.address%type;

-- user defined exception

ex\_invalid\_id EXCEPTION;

BEGIN

IF c\_id <= 0 THEN

RAISE ex\_invalid\_id;

ELSE

**SELECT**  name, address **INTO** c\_name, c\_addr **FROM** customers  **WHERE** id = c\_id;

DBMS\_OUTPUT.PUT\_LINE ('Name: '|| c\_name);

DBMS\_OUTPUT.PUT\_LINE ('Address: ' || c\_addr);

END IF;

EXCEPTION

WHEN ex\_invalid\_id THEN

dbms\_output.put\_line('ID must be greater than zero!');

WHEN no\_data\_found THEN

dbms\_output.put\_line('No such customer!');

WHEN others THEN

dbms\_output.put\_line('Error!');

END;

/

OUTPUT:

Enter value for cc\_id: -6 (let's enter a value -6)

old 2: c\_id customers.id%type := &cc\_id;

new 2: c\_id customers.id%type := -6;

ID must be greater than zero!

PL/SQL procedure successfully completed.

**Triggers:**

Write a pl/sql program by using triggers find out the salary changes.

CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE DELETE OR INSERT OR UPDATE ON customers

FOR EACH ROW

WHEN (NEW.ID > 0)

DECLARE

sal\_diff number;

BEGIN

sal\_diff := :NEW.salary - :OLD.salary;

dbms\_output.put\_line('Old salary: ' || :OLD.salary);

dbms\_output.put\_line('New salary: ' || :NEW.salary);

dbms\_output.put\_line('Salary difference: ' || sal\_diff);

END;

/

**Packages:**

1. Create package for customers table

CREATE PACKAGE cust\_sal AS

PROCEDURE find\_sal(c\_id customers.id%type);

END cust\_sal;

/

Package created.

1. Create or Replace package body for customers table

CREATE OR REPLACE PACKAGE BODY cust\_sal AS

PROCEDURE find\_sal(c\_id customers.id%TYPE) IS

c\_sal customers.salary%TYPE;

BEGIN

SELECT salary INTO c\_sal

FROM customers

WHERE id = c\_id;

dbms\_output.put\_line('Salary: '|| c\_sal);

END find\_sal;

END cust\_sal;

/

Package body created.

package\_name.element\_name;

1. View the customers table data based on user’s choice

DECLARE

code customers.id%type := &cc\_id;

BEGIN

cust\_sal.find\_sal(code);

END;

/

Enter value for cc\_id: 1

Salary: 3000

PL/SQL procedure successfully completed.

1. Create or replace **package specification** for customers table.

CREATE OR REPLACE PACKAGE c\_package AS

-- Adds a customer

PROCEDURE addCustomer(c\_id customers.id%type,

c\_name customers.Name%type,

c\_age customers.age%type,

c\_addr customers.address%type,

c\_sal customers.salary%type);

-- Removes a customer

PROCEDURE delCustomer(c\_id customers.id%TYPE);

--Lists all customers

PROCEDURE listCustomer;

END c\_package;

/

1. creating the pacakage body

CREATE OR REPLACE PACKAGE BODY c\_package AS

PROCEDURE addCustomer(c\_id customers.id%type,

c\_name customers.Name%type,

c\_age customers.age%type,

c\_addr customers.address%type,

c\_sal customers.salary%type)

IS

BEGIN

**INSERT INTO** customers (id,name,age,address,salary)  **VALUES**(c\_id, c\_name, c\_age, c\_addr, c\_sal);

END addCustomer;

PROCEDURE delCustomer(c\_id customers.id%type) IS

BEGIN

**DELETE FROM** customers  **WHERE** id = c\_id;

END delCustomer;

PROCEDURE listCustomer IS

**CURSOR** c\_customers **is** **SELECT**  name **FROM** customers;

TYPE c\_list is TABLE OF customers.Name%type;

name\_list c\_list := c\_list();

counter integer :=0;

BEGIN

FOR n IN c\_customers LOOP

counter := counter +1;

name\_list.extend;

name\_list(counter) := n.name;

dbms\_output.put\_line('Customer(' ||counter|| ')'||name\_list(counter));

END LOOP;

END listCustomer;

END c\_package;

/

6.using the package

DECLARE

code customers.id%type:= 8;

BEGIN

c\_package.addcustomer(7, 'Rajnish', 25, 'Chennai', 3500);

c\_package.addcustomer(8, 'Subham', 32, 'Delhi', 7500);

c\_package.listcustomer;

c\_package.delcustomer(code);

c\_package.listcustomer;

END;

/

**output**

Customer(1): Ramesh

Customer(2): Khilan

Customer(3): kaushik

Customer(4): Chaitali

Customer(5): Hardik

Customer(6): Komal

Customer(7): Rajnish

Customer(8): Subham

Customer(1): Ramesh

Customer(2): Khilan

Customer(3): kaushik

Customer(4): Chaitali

Customer(5): Hardik

Customer(6): Komal

Customer(7): Rajnish

PL/SQL procedure successfully completed