**Working with Functions**

**Q: PL/SQL Function that computes and returns the maximum of two values.**

DECLARE

a number;

b number;

c number;

FUNCTION findMax(x IN number, y IN number)

RETURN number

IS

z number;

BEGIN

IF x > y THEN

z:= x;

ELSE

Z:= y;

END IF;

RETURN z;

END;

BEGIN

a:= 23;

b:= 45;

c := findMax(a, b);

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

END;

/

**Output:**

Maximum of (23,45): 45

PL/SQL procedure successfully completed.

**Q:PL/SQL Recursive Functions**

DECLARE

num number;

factorial number;

FUNCTION fact(x number)

RETURN number

IS

f number;

BEGIN

IF x=0 THEN

f := 1;

ELSE

f := x \* fact(x-1);

END IF;

RETURN f;

END;

BEGIN

num:= 6;

factorial := fact(num);

dbms\_output.put\_line(' Factorial '|| num || ' is ' || factorial);

END;

/

**Output:**

Factorial 6 is 720

PL/SQL procedure successfully completed.

**Working with Cursors**:

Q: **to update the table and increase salary of each customer by 5000 USING Implicit Cursors.**

**Create customers table and have records:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **NAME** | **AGE** | **ADDRESS** | **SALARY** |
| 1 | Ramesh | 23 | Allahabad | 20000 |
| 2 | Suresh | 22 | Kanpur | 22000 |
| 3 | Mahesh | 24 | Ghaziabad | 24000 |
| 4 | Chandan | 25 | Noida | 26000 |
| 5 | Alex | 21 | Paris | 28000 |
| 6 | Sunita | 20 | Delhi | 30000 |

After Creating Above Table Then Only Execute The below Code:

DECLARE

total\_rows number(4);

BEGIN

UPDATE customers

SET salary = salary + 5000;

IF sql%notfound THEN

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

ELSIF sql%found THEN

total\_rows := sql%rowcount;

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

END IF;

END;

/

**Output:**

6 customers updated

PL/SQL procedure successfully completed.

**Working with Triggers**

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;

/

Trigger Created.

>INSERT INTO CUSTOMERS (ID,NAME,AGE,ADDRESS,SALARY)

VALUES (7, 'Kriti', 22, 'HP', 7500.00 );

>UPDATE customers

SET salary = salary + 500

WHERE id = 2;

>Old salary: 1500

New salary: 2000

Salary difference: 500

**Working with Exceptions(User defined)**

**1.**

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;

/

**Output:**

No such customer!

PL/SQL procedure successfully completed.

**2.Raising Exeptions:**

ECLARE

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

ID must be greater than zero!

PL/SQL procedure successfully completed.