

# CURSOR FUNCTION PROCEDURE

G. M. Shahariar

Lecturer

Department of CSE

Ahsanullah University of Science & Technology



# Question

Write a PL/SQL code to implement the following –

- › Prompt to user: “Enter last 3 digits of your ID = “
- › Nested Anonymous Blocks (BEGIN – END blocks)
- › Take Input in the outer block
- › Check Even/Odd in the inner block
- › Print “Even”/”Odd” in the outer block

- Run 1.sql
- Run 2.sql

What does the error say?

How can we get multiple rows inside BEGIN – END block?

# PL/SQL CURSOR

$\pi$

A cursor is a pointer that points to a result of a query. **A cursor holds the rows (one or more) returned by a SQL statement.** The set of rows the cursor holds is referred to as the **active set**.

You can name a cursor so that it could be referred to in a program to **fetch and process the rows returned by the SQL statement, one at a time.**

# Remember FOR LOOP Syntax ?

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    NUM number := 5;
```

```
BEGIN
```

```
    FOR i IN 1..NUM LOOP
```

```
        DBMS_OUTPUT.PUT_LINE(i);
```

```
    END LOOP;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
  A money.id%TYPE;
```

```
  B money.taka%TYPE;
```

```
BEGIN
```

```
  FOR R IN (SELECT id, taka from money) LOOP
```

```
    A := R.id;
```

```
    B := R.taka;
```

```
    DBMS_OUTPUT.PUT_LINE(A||' '||B);
```

```
  END LOOP;
```

```
END;
```

```
/
```

## PL/SQL Procedure

Like a PL/SQL function, a **PL/SQL procedure** is a named block that does a specific task. PL/SQL procedure allows you to encapsulate complex business logic and reuse it in both database layer and application layer.

- › Run procedure.sql
- › Open main.sql
- › You can explicitly run a procedure in SQLPLUS CMD:  
**EXEC procedure-name(value);**



## PL/SQL Function

**PL/SQL function** is a named block that returns a value. A PL/SQL function is also known as a subroutine or a subprogram.

- › Run function.sql
- › Open main.sql
- › You can explicitly run a function in SQLPLUS CMD:  
`select function-name(value) from dual;`

For both function & procedure, each parameter has one of three modes: IN, OUT and IN OUT.

An **IN parameter** is a read-only parameter. If the function tries to change the value of the IN parameters, the compiler will issue an error message. You can pass a constant, literal, initialized variable, or expression to the function as the IN parameter.

An **OUT parameter** is a write-only parameter. The OUT parameters are used to return values back to the calling program. An OUT parameter is initialized to a default value of its type when the function begins regardless of its original value before being passed to the function.

An **IN OUT parameter** is read and write parameter. It means the function reads the value from an IN OUT parameter, change its value and return it back to the calling program. The RETURN clause in the function header specifies the data type of returned value.

## Built-in Functions

- › <https://www.guru99.com/subprograms-procedures-functions-pl-sql.html>

$\pi$

No OFFLINE

Next Day ONLINE