

**❖ AIM**

Create employee management system using sql and PL/SQL(use the concept of cursor, trigger, function, package and exception handling)

Note: create atleast three modules.

**❖ EXPLANATION OF PROJECT**

In this project I have created a table that contains the data of 7 employees . Attributes of this employee table are employee number , name , salary.

**❖ MODULE IN PROJECT**

- 1) Procedure
- 2) Cursor
- 3) Exception handling

## ❖ DESCRIPTION OF EACH MODULE

### 1) Procedure:

A Procedure is a subprogram unit that consists of a group of PL/SQL statements. Each procedure in Oracle has its own unique name by which it can be referred. This subprogram unit is stored as a database object. Below are the characteristics of this subprogram unit.

#### Syntax:

```
CREATE OR REPLACE PROCEDURE <procedure_name>
```

```
( <parameter> IN/OUT <datatype>)
```

```
[IS / AS]
```

```
BEGIN
```

```
<execution part>
```

```
EXCEPTION
```

```
<exception handling part>
```

```
END;
```

### 2) Cursor:

A **cursor** is a pointer to this context area. PL/SQL controls the context area through a cursor. 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**.

There are two types of cursors –

- Implicit Cursor

Implicit cursors are automatically created by Oracle whenever an SQL statement is executed, when there is no explicit cursor for the statement. Programmers cannot control the implicit cursors and the information in it.

- Explicit Cursor

Explicit cursors are programmer-defined cursors for gaining more control over the **context area**. An explicit cursor should be defined in the declaration section of the PL/SQL Block. It is created on a SELECT Statement which returns more than one row.

#### Syntax:

```
DECLARE cursor_name CURSOR
```

```
FOR select_statement;  
OPEN cursor_name;  
FETCH NEXT FROM cursor INTO variable_list;  
WHILE @@FETCH_STATUS = 0  
BEGIN  
    FETCH NEXT FROM cursor_name;  
END;  
CLOSE cursor_name;  
DEALLOCATE cursor_name;
```

### 1) **Exception handling:**

Exception Handling in PL/SQL. An exception is an error which disrupts the normal flow of program instructions. PL/SQL provides us the exception block which raises the exception thus helping the programmer to find out the fault and resolve it.

#### **Syntax:**

```
DECLARE  
  
<declaration section>  
  
BEGIN  
  
<executable command(s)>  
  
EXCEPTION  
  
<exception handling goes here>  
  
WHEN exception1 THEN  
    <exception 1 handling statement>  
  
WHEN exception2 THEN  
    < exception 2 handling statement >  
  
WHEN exception3 THEN  
    < exception 3 handling statement >
```

.....

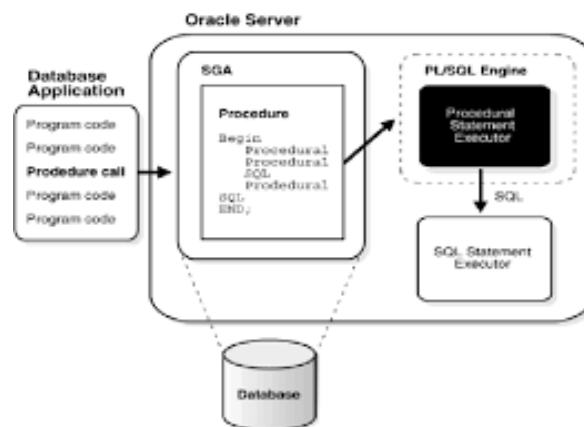
WHEN others THEN

< other exception handling statement >

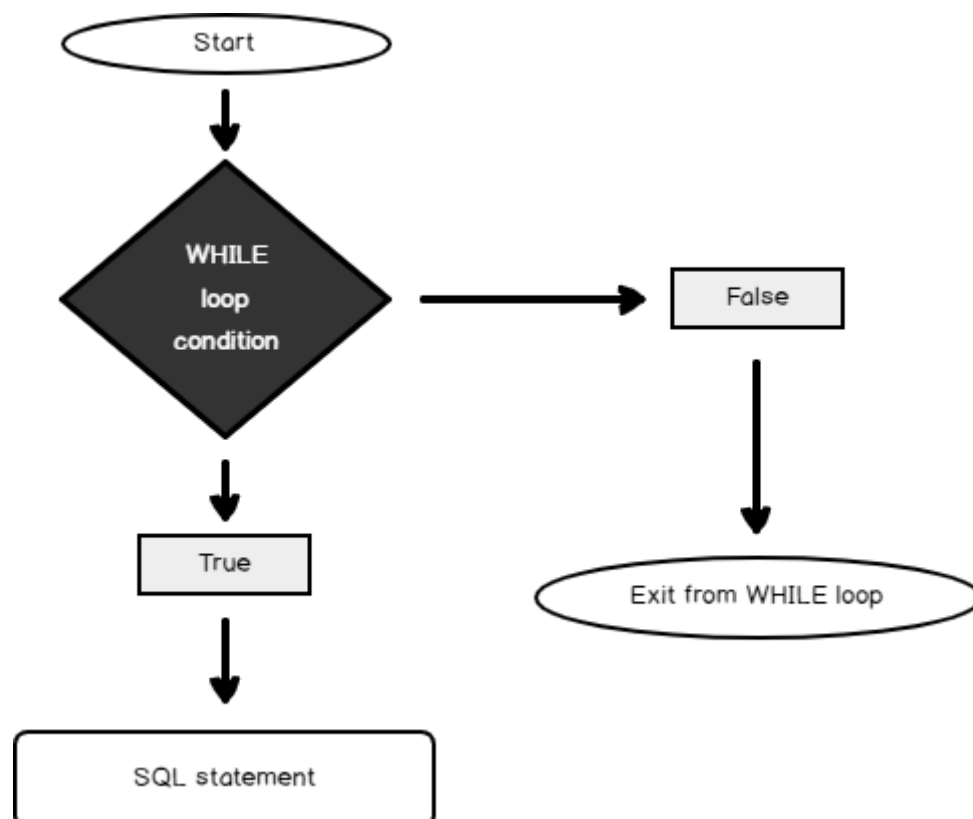
END;

### ❖ Execution flow in the form of flow chart & Explanation of flow

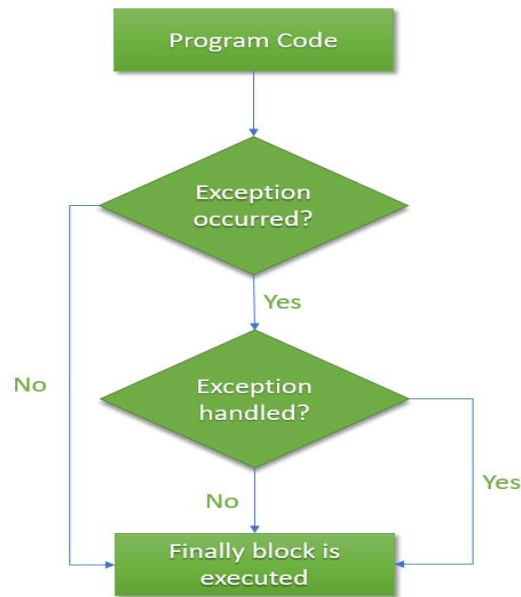
#### 1) Procedure: -



#### 2) Cursor: -



### 3) Exception handling: -



### ❖ MAIN CODE

#### 1) Procedure:

##### Aim: -

Create procedure for employee interview.

##### Program code: -

```
CREATE OR REPLACE PROCEDURE employee_interview(b_name in varchar2)
```

```
IS
```

```
BEGIN
```

```
dbms_output.put_line('Employee Name: ' || b_name || ' is approved.');
```

```
END;
```

```
/
```

```
BEGIN
```

```
Employee_interview('kashyap');
```

```
END;
```

/

### 3) Cursor: -

#### Aim: -

To display all the employee details whose salary is greater than 2,500.

#### Program code: -

```
set serveroutput on;

declare

cursor c2 (e_sal number)is

select emp_name,emp_no from employee

where emp_sal>e_sal;

Begin

for i in c2(2500)

loop

dbms_output.put_line(i.emp_name || ' ' || i.emp_sal);

end loop;

end;
```

/

### 3)Exception handling: -

#### Aim:

Generate an exception for invalid input when entered employee number not found from the table.

#### Program code:

```
set serveroutput on;

declare

v_id number:=&v_id;
```

```
ex_invalid exception;
```

```
Begin
```

```
if v_id>107 then
```

```
raise ex_invalid;
```

```
end if;
```

```
Exception
```

```
when ex_invalid then
```

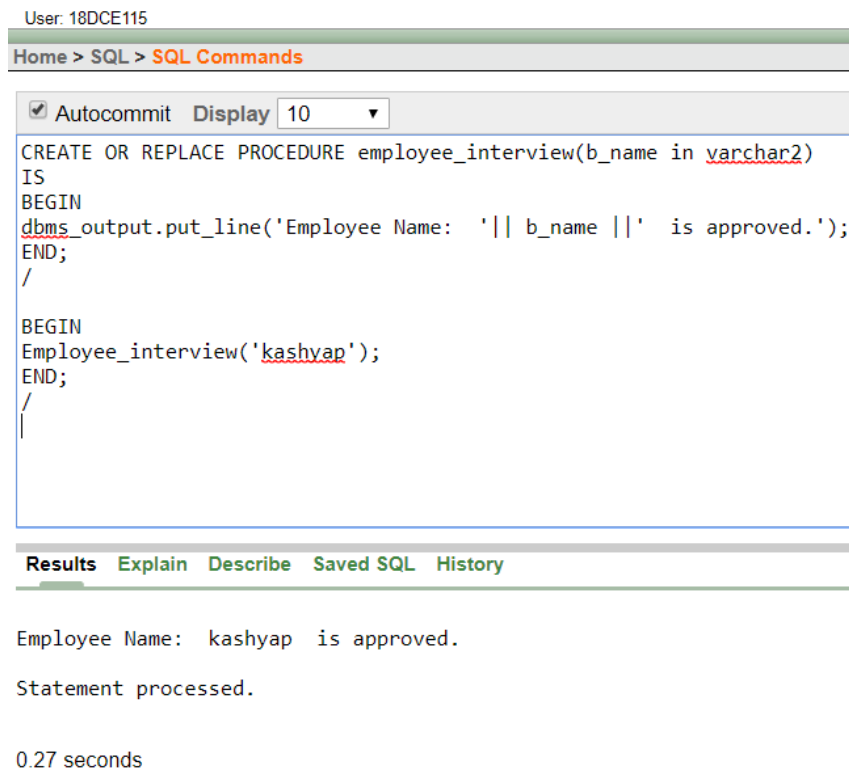
```
dbms_output.put_line('No such employee');
```

```
end;
```

```
/
```

## ❖ Result

### 1) Procedure: -



The screenshot shows a web-based SQL interface. At the top, it says 'User: 18DCE115'. Below that is a breadcrumb 'Home > SQL > SQL Commands'. There is a toolbar with 'Autocommit' checked and a 'Display' dropdown set to '10'. The main text area contains the following SQL code:

```
CREATE OR REPLACE PROCEDURE employee_interview(b_name in varchar2)
IS
BEGIN
  dbms_output.put_line('Employee Name: ' || b_name || ' is approved. ');
END;
/

BEGIN
  Employee_interview('kashyap');
END;
/
|
```

Below the code area is a tabbed interface with 'Results' selected. The results pane shows the output of the procedure execution:

```
Employee Name: kashyap is approved.
Statement processed.

0.27 seconds
```

**2) Cursor: -**

```
SQL> @demo.sql
Aman    3000
Anamika  2975

PL/SQL procedure successfully completed.
```

**3) Exception handling:**

```
SQL> @demo.sql
Enter value for v_id: 103
old   2: v_id number:=&v_id;
new   2: v_id number:=103;

PL/SQL procedure successfully completed.

SQL> @demo.sql
Enter value for v_id: 108
old   2: v_id number:=&v_id;
new   2: v_id number:=108;
No such employee

PL/SQL procedure successfully completed.
```

**❖ Conclusion**

Here in this mini project activity I have cleared my concepts of all these 3 topics that is procedure, Cursor and Exception handling.