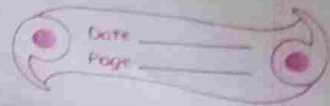


Assignment-3



Q-1 what is Exception? Explain type of exception with example.

- an exception is defined as a special condition that changes the program execution flow.
- the PL/SQL provides you with a flexible and powerful way to handle such exception.
- in the exception section, you can check what kind of exception has been occurred and handle it appropriately.

→ there are three types of exceptions:-

1. Predefined exception:

- system exception are automatically raised by oracle, when a program violates a RDBMS rule.
- there are some system exception which are raised frequently, so they are pre-defined and given a name in oracle which is known as named system exception.

- Example -

```
declare
  n1 number(3);
  n2 number(3);
  ans number(8,2);
begin
  n1 := &n01;
  n2 := &n02;
  ans := n1/n2;
  dbms_output.put_line(ans);
exception
  when zero_divide then
    dbms_output.put_line('divide zero error');
end;
```

2. undefined exception:

- it's an Oracle error that doesn't have a predefined exception.
- in this case you need to declare an exception and associate an Oracle number with it.
- those system exception for which Oracle does not provide a name is known as unnamed system exception. These exceptions do not occur frequently.

- Example -

```
declare
  error_or exception;
  pragma exception_init(error_or, -2045);
begin
  delete from employee where id=4;
exception
  when error_or then
    dbms_output.put_line('exception ( )');
end;
```

3. user defined exception:

- the user defined exception sometimes called programmer-defined exception is defined by you in a specific application.
- a user defined exception must be declared and then raised explicitly, using either a raise statement or the procedure dbms_standard.raise_application_error.
- you can map exception names with specific Oracle errors using the exception_init pragma.

- Example -

```
declare  
  myex exception;  
  i number;  
begin  
  for i in (select * from emp) loop  
    if i.eno=3 then  
      raise myex;  
    end if;  
  end loop;  
exception  
  when myex then  
    dbms_output.put_line('emp exists');  
end;
```

Q-2 Explain sub-program with its type

→ sub-program is an individual part of PL/SQL block which can be used to perform a specific task.

→ there are two types of subprograms:

1. PL/SQL procedure:

→ the PL/SQL stored procedure or simply a procedure is a PL/SQL block which perform one or more specific task. It is just like procedure in other programming language.

→ the procedure contains a header and a body

→ header contains the name of procedure and the parameter or variables passed to the procedure.

→ body contains a declaration section, execution section and exception section similar to a general PL/SQL block.

- Syntax -

```
create [or replace] procedure  
procedure_name [(parameter [in|out]  
type)]  
is  
    [declaration section]  
begin  
    executable_section  
[exception  
    exception_section]  
end;  
/
```

- Example -

```
create procedure p4 [id in number]  
is  
begin  
    insert into user values (id);  
end;  
/
```

- how to execute procedure?

```
execute [or exec] procedure_name  
procedure_name
```

2 PL/SQL function:

→ PL/SQL function is very similar to PL/SQL procedure.

→ main difference between procedure and function is, a function must always return a value, and on the other hand a procedure may or may not return a value.

→ header contains the name of function and the parameter or variables passed to the function.

→ body contains a declaration section.

- Syntax -

```
create [or replace] function fun_name  
[(parameter [in|out] type)]  
return return_datatype  
is  
    [declaration section]  
begin  
    <function body>  
end;  
/
```

- Example -

create or replace function add(n1 in
number, n2 in number)
return number

is

n3 number(8);

begin

n3 := n1 + n2;

end;

/

- execution function -

- @function_name

- declare

n3 number(8);

begin

n3 := add(10, 20);

dbms_output.put_line(n3);

end;

/

Q-3 what is trigger? Explain it's type.

→ trigger is PL/SQL block structure
which is fired when a DML statement
is executed on a database table.

→ trigger are stored programs, which are
automatically executed or fired when
some events occur.

→ trigger is triggered automatically
when an associated DML statement is
executed.

→ types of PL/SQL trigger:

1 row level trigger - an event is triggered
for each row updated, inserted or deleted.

2 statement level trigger - an event is
triggered for each SQL statement
executed.

• before trigger - it execute before the
triggering DML statement execute.

• after trigger - it execute after the
triggering DML statement execute.

- syntax -

```
create [or replace] trigger trigger_name  
{ before | after } { insert [or] update  
[or] delete } [of col_name]  
on table_name
```

```
[for each row [when (con.)]]
```

```
[referencing old as old new as n]  
declare
```

```
declare_section
```

```
begin
```

```
execution_section
```

```
exception
```

```
exception_section
```

```
end;
```

```
/
```

- Example -

```
create trigger sal_change before  
delete or insert or update on customer  
for each row
```

```
when (new.id > 0)
```

```
declare
```

```
sal_diff number;
```

```
begin
```

```
sal_diff := new.sal - old.sal;
```

```
d_o.p-1 (:old.sal);
```

```
d_o.p-1 (:new.sal);
```

```
d_o.p-1 (sal_diff);
```

```
end;
```

```
/
```