**EXCPETIONS IN Oracle**

An exception is an error raised during program execution either implicitly or explicitly by the program.

Exception is an runtime error.

Any abnormal condition that interrupts the normal flow of the program’s instruction at runtime is an exception.

**Exception handling** :- is a possibility to keep our program running even if appear runtime error resulting from code mistakes or hardware failure . we avoid it from exiting abruptly.

**Syntax**

DECLARE

Declaration section

BEGIN

Some statements;

EXCEPTION

WHEN exception\_one then

Do something

When exception\_two then

Do something

When others then

Do something

End;

/

Each block can have its own exception

DECLARE

Declaration section

BEGIN

Some statements;

Begin

Some statement;

Exception when exception\_one then

End;

EXCEPTION

WHEN exception\_two then

Do something

End;

**Types of Exception**

**System- Defined Exception** - defined and maintained implicitly by the oracle server.

Defined in oracle standard package.

**User defined exception** – unlike system defined exception, these exceptions are raised explicitly by pl/sql block

How to declare user defined exceptions.

1. Using variable of EXCEPTION type.

Declare a user defined exception by declaring a variable of exception datatype and raising it explicitly in your program using RAISE statement.

1. Using PRAGMA EXCEPTION\_INIT function

Using this we can map a non predefined error number with the variable of exception datatype.

1. Using RAISE\_APPLICATION\_ERROR

We can declare a user defined exception with own customized error number and message.

**Example 1**

1. Declare a variable of exception type
2. Raise the exception
3. Handle the exception

DECLARE

Var\_dividend number:=24;

Var\_divisor number:=0;

Var\_result number;

Ex\_divzero EXCEPTION;

BEGIN

IF var\_divisor =0 THEN

Raise ex\_divzero;

END IF;

Var\_result:=var\_dividend/var\_divisor;

DBMS\_OUTPUT.PUTLINE(‘Result =’||var\_Result);

EXCEPTION WHEN ex\_divzero THEN

DBMS\_OUTPUT.PUTLINE(‘ERROR divisor is zero);

END;

/

**Example for 2:**

Using pragma exception\_init we can associate an exception name with an oracle error number.

Pragma:- compiler directive which indicates that statement followed by pragma is compiler directive statements this means that statement will be processed at compile time but not at runtime.

Exception\_init :- helps in associating exception name with the oracle error number.

DECLARE

Age number:17;

Ex\_Age Exception;

Pragma exception\_init(ex\_Age,-20008);

Begin

If age <18 then

Raise\_application\_error(-20008,’Your age is less than 18’);

End if;

Dbms\_output.put\_line(‘you are eligible for voting’);

Exception when ex\_Age then

Dbms\_output.put\_line(sqlerrm);

End;

/

**Example for 3**: raise\_application\_error accepts two parameters first is number which is negative between -20001 to -20999 and second is string error message to be displayed.

ACCEPT var\_age NUMBER PROMPT ‘What is you age’;

DECLARE

Age:=&var\_Age;

Begin

If age <18 then

Raise\_application\_error(-20008,’Your age is less than 18’);

End if;

Dbms\_output.put\_line(‘you are eligible for voting’);

Exception when others then

Dbms\_output.put\_line(sqlerrm);

End;

/

**Predefined Exceptions** :- predefined exceptions are internally defined exceptions that have predefined names, which plsql declares globally in the package standard.

NO\_DATA\_FOUND ora-01403

Too\_many\_rows ora-1422

zero\_divide ora-1476

dup\_value\_on\_index ora-00001

invalid\_cursor ora-1001

cursor already\_open ora-06511

invalid\_number ora-01722

value\_Error ora-06502

case\_not found ora-06592

1. NO\_DATA\_FOUND

DECLARE

L\_ename VARCHAR2(30);

L\_empno NUMBER:=&emp;

BEGIN

SELECT ename

FROM emp

WHERE empno=l\_empno

AND rownum=1;

DBMS\_OUTPUT.PUT\_LINE(‘Employee name is ‘||l\_ename);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE(‘Employee with id’||l\_empno||’doesnot exists’);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(‘Some problem with code’);

END;

/

1. Too\_many\_rows

DECLARE

L\_ename VARCHAR2(30);

L\_job VARCHAR2(30):=’&job’;

BEGIN

SELECT ename

FROM emp

WHERE empno= L\_job

AND rownum=1;

DBMS\_OUTPUT.PUT\_LINE(‘Employee name with job is ‘|| L\_job );

EXCEPTION

WHEN Too\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE(‘More then one job is present for ‘|| L\_job );

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(‘Some problem with code’);

END;

/

1. Zero\_divide

DECLARE

L\_result NUMBER(6);

L\_num1 NUMBER(5):=&num1;

L\_num2 NUMBER(5):=&num2;

BEGIN

L\_result:=l\_num1/l\_num2;

DBMS\_OUTPUT.PUT\_LINE(‘Result of two numbers is ‘|| L\_result);

EXCEPTION

WHEN Zero\_divide THEN

DBMS\_OUTPUT.PUT\_LINE(‘denominator is zero’);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(‘Some problem with code’);

END;

/

13

0

Denominator is zero

1. **Dup\_val\_on\_index**

DECLARE

L\_empid:=&id;

L\_name:=’&name’;

L\_Sal:=&sal;

BEGIN

INSERT INTO emp1 VALUES(l\_empid,l\_name,l\_Sal);

Commit;

DBMS\_OUTPUT.PUT\_LINE(‘Employee name is ‘|| l\_name||’Employee id is’||l\_empid||’ employee salary is’||l\_Sal);

EXCEPTION

WHEN Dup\_val\_on\_indexTHEN

DBMS\_OUTPUT.PUT\_LINE(‘Employee with id ‘|| l\_empid||’ exists’);

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(‘Some problem with code’);

END;

/

o/p:-

1. **Invalid\_cursor**

DECLARE

Cursor c1 is

SELECT \* from emp1;

Rec c1%rowtype;

BEGIN

LOOP

FETCH c1 INTO rec;

EXIT WHEN c1%NOTFOUND;

Dbms\_output.put\_line(‘Employee id is’||rec.empid||’Employee name’||rec.empname ||’Employee salary is’||rec.sal);

END LOOP;

Close c1;

EXCEPTION

WHEN INVALID\_CURSOR THEN

Dbms\_ouput.put\_line(‘cursor is not open’);

END;

/

o/p

cursor is not open

1. **cursor already\_open ora-06511**

DECLARE

Cursor c1 is

SELECT \* from emp1;

Rec c1%rowtype;

BEGIN

Open c1;

LOOP

FETCH c1 INTO rec;

EXIT WHEN c1%NOTFOUND;

Dbms\_output.put\_line(‘Employee id is’||rec.empid||’Employee name’||rec.empname);

Open c1;

Dbms\_output.put\_line(’Employee salary||rec.sal);

END LOOP;

Close c1;

EXCEPTION

WHEN INVALID\_CURSOR THEN

Dbms\_ouput.put\_line(‘cursor is not open’);

WHEN cursor\_already\_open THEN

Dbms\_ouput.put\_line(‘cursor is already open’);

END;

/

o/p

cursor is already open

1. **invalid\_number**

set serveroutput on;

BEGIN

INSERT INTO EMP(empno,sal) VALUES(8888,’1234xx’);

Commit;

EXCEPTION WHEN INVALID\_NUMBER THEN

Dbms\_output.put\_line(‘cannot convert string to number’);

End;

/

cannot convert string to number

1. **value\_Error**

DECLARE

L\_Sal number;

L\_var varchar2(2);

BEGIN

L\_var:=’abc’;

Select sal

Into l\_Sal

From emp

Where empno=100;

Dbms\_ouput.put\_line(‘employees salary:”||l\_Sal);

EXCEPTION

WHEN value\_error THEN

Dbms\_ouput.put\_line(‘Some error in value assignment’);

End;

/

o/p

Some error in value assignment

1. **case\_not found**

DECLARE

L\_output varchar2(100);

BEGIN

case &i

WHEN 1 THEN l\_output:=’choice is 1’

WHEN 2 THEN l\_output:=’choice is 2’

Dbms\_ouput.put\_line(l\_output);

EXCEPTION

WHEN CASE\_NOT\_FOUND THEN

Dbms\_ouput.put\_line(‘case is not defind please check’);

End;

/