



ORACLE PL/SQL



MÓDULO 06

TRATAMIENTO DE ERRORES

GUSTAVO CORONEL
desarrollasoftware.com



ORACLE PL/SQL



Contenido

TIPOS DE ERRORES.....	3
EXCEPCIONES	3
ESQUEMA GENERAL.....	3
EXCEPCIONES PREDEFINIDAS.....	4
INSTRUCCIÓN RAISE	5
EXCEPCIONES DE USUARIO	6
GENERACIÓN DE MENSAJES DE ERROR	7
CURSOS VIRTUALES	9
JAVA ORIENTADO A OBJETOS.....	¡ERROR! MARCADOR NO DEFINIDO.
PROGRAMACIÓN DE BASE DE DATOS CON JAVA JDBC	¡ERROR! MARCADOR NO DEFINIDO.



ORACLE PL/SQL



TIPOS DE ERRORES

TIPO DE ERROR	QUIÉN INFORMA	CÓMO ES TRATADO
DE COMPILACIÓN	Compilador PL/SQL	Interactivamente: el compilador informa de los errores y el programador debe corregirlos.
DE EJECUCIÓN	Motor de Ejecución PL/SQL	Programáticamente: las excepciones son generadas e interceptadas por las rutinas de tratamiento de excepciones.

EXCEPCIONES

Es el mecanismo de tratamiento de errores en tiempo de ejecución. Tenemos dos tipos de excepciones:

- Definidas por el usuario
- Predefinidas

ESQUEMA GENERAL

Formato:

```
BEGIN
-----
-----
-----
EXCEPTION
  WHEN nombre_excepción THEN
    secuencia_de_instrucciones;
  WHEN nombre_excepción THEN
    secuencia_de_instrucciones;
  [ WHEN OTHERS THEN
    secuencia_de_instrucciones; ]
END;
```



ORACLE PL/SQL



EXCEPCIONES PREDEFINIDAS

Oracle ha definido diversas excepciones que corresponden con los errores Oracle más comunes.

EXCEPCIÓN	DESCRIPCIÓN
INVALID_CURSOR	Ocurre cuando se hace referencia a un cursor que esta cerrado.
CURSOR_ALREADY_OPEN	Ocurre cuando se trata de abrir un cursor que ya esta abierto.
NO_DATA_FOUND	Ocurre cuando una sentencia SELECT no retorna ninguna fila.
TOO_MANY_ROWS	Ocurre cuando una sentencia SELECT retorna mas de una fila.
VALUE_ERROR	Ocurre cuando hay conflicto de tipos de datos.

Script 1

Desarrollar un procedimiento para consultar el salario de un empleado.

```
create or replace procedure FindEmp( Cod Emp.EmpNo%Type )
is
    Salario Emp.Sal%Type;
Begin
    Select Sal Into Salario
    From Emp
    Where EmpNo = Cod;
    DBMS_Output.Put_Line( 'Salario: ' || Salario );
Exception
    When No_Data_Found Then
        DBMS_Output.Put_Line( 'Código no existe.' );
End;
```

Ejecución:

```
SQL> exec FindEmp( 9999 );
Código no existe.

PL/SQL procedure successfully completed.
```



INSTRUCCIÓN RAISE

Permite generar una excepción.

Script 2

```
create or replace procedure UpdateSalEmp
( Codigo Emp.EmpNo%Type, Salario Emp.Sal%Type )
is
    Cont Number;
Begin
    Select Count(*) Into Cont
    From Emp
    Where EmpNo = Codigo;
    If (Cont=0) Then
        Raise No_Data_Found;
    End If;
    Update Emp
    Set Sal = Salario
    Where EmpNo = Codigo;
    Commit;
    DBMS_Output.Put_Line( 'Proceso OK' );
Exception
    When No_Data_Found Then
        DBMS_Output.Put_Line( 'Código no existe.' );
End;
```

Ejecución:

```
SQL> exec UpdateSalEmp( 9999, 5000 );
Código no existe.

PL/SQL procedure successfully completed.
```



EXCEPCIONES DE USUARIO

Script 3

Desarrollar una segunda versión del procedimiento UpdateSalEmp, pero con una excepción de usuario.

```
create or replace procedure UpdateSalEmp2
( Codigo Emp.EmpNo%Type, Salario Emp.Sal%Type )
is
    Cont Number;
    Excep1 Exception;
Begin
    Select Count(*) Into Cont
    From Emp
    Where EmpNo = Codigo;
    If (Cont=0) Then
        Raise Excep1;
    End If;
    Update Emp
    Set Sal = Salario
    Where EmpNo = Codigo;
    Commit;
    DBMS_Output.Put_Line( 'Proceso OK' );
Exception
    When Excep1 Then
        DBMS_Output.Put_Line( 'Código no existe.' );
End;
```

Ejecución:

```
SQL> exec UpdateSalEmp2( 9999, 5000 );
Código no existe.

PL/SQL procedure successfully completed.
```



GENERACIÓN DE MENSAJES DE ERROR

Los mensajes de error que se generan con `RAISE_APPLICATION_ERROR` deben estar entre -20,999 y -20,000.

Script 4

Desarrollar una tercera versión del procedimiento `UpdateSalEmp`, pero esta vez genere un mensaje de error.

```
create or replace procedure UpdateSalEmp3
( Codigo Emp.EmpNo%Type, Salario Emp.Sal%Type )
is
    Cont Number;
Begin
    Select Count(*) Into Cont
    From Emp
    Where EmpNo = Codigo;
    If (Cont=0) Then
        Raise_Application_Error( -20000, 'No existe empleado.' );
    End If;
    Update Emp
    Set Sal = Salario
    Where EmpNo = Codigo;
    Commit;
    DBMS_Output.Put_Line( 'Proceso OK' );
End;
```

Ejecución:

```
SQL> exec UpdateSalEmp3( 9999, 5000 );
BEGIN UpdateSalEmp3( 9999, 5000 ); END;

*
ERROR at line 1:
ORA-20000: No existe empleado.
ORA-06512: at "SCOTT.UPDATESALEMP3", line 10
ORA-06512: at line 1
```



ORACLE PL/SQL



Script 5

Otra versión del mismo procedimiento.

```
create or replace procedure UpdateSalEmp4
( Codigo Emp.EmpNo%Type, Salario Emp.Sal%Type )
is
    Cont Number;
Begin
    Select Count(*) Into Cont
    From Emp
    Where EmpNo = Codigo;
    If (Cont=0) Then
        Raise_Application_Error( -20000, 'No existe empleado.' );
    End If;
    Update Emp
    Set Sal = Salario
    Where EmpNo = Codigo;
    Commit;
    DBMS_Output.Put_Line( 'Proceso OK' );
Exception
    When Others Then
        dbms_output.put_line ( 'Error Nro. ORA' || to_char(sqlcode) );
        dbms_output.put_line ( sqlerrm );
End;
```

Ejecución:

```
SQL> exec UpdateSalEmp4( 9999, 5000 );
Error Nro. ORA-20000
ORA-20000: No existe empleado.

PL/SQL procedure successfully completed.
```




ORACLE PL/SQL



CURSOS VIRTUALES

En estos enlaces se publican cupones de descuento:

- <https://github.com/gcoronelc/UDEMY>
- <https://www.facebook.com/groups/bolsa.sistemas>
- <https://www.facebook.com/groups/universidadjava>
- <https://www.facebook.com/groups/desarrollasoftware>
- <https://chat.whatsapp.com/H6BBebNDZHEAqAk6gTU5ZS>

JAVA ORIENTADO A OBJETOS



JAVA ORIENTADO A OBJETOS

Eric Gustavo Coronel Castillo
www.desarrollasoftware.com
INSTRUCTOR

<https://bit.ly/2B3ixUW>

PROGRAMACIÓN DE BASE DE DATOS CON JAVA JDBC



PROGRAMACIÓN DE BASE DE DATOS ORACLE CON JAVA JDBC

Eric Gustavo Coronel Castillo
www.desarrollasoftware.com
INSTRUCTOR

<https://bit.ly/31apy0O>