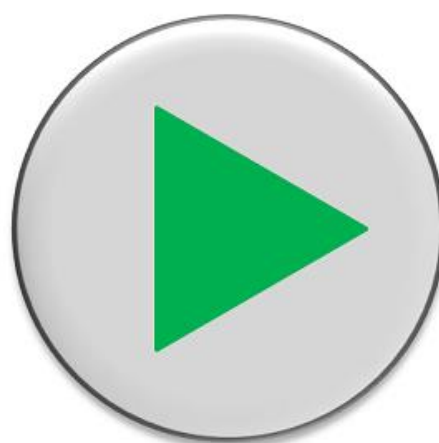




# ORACLE PL/SQL



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  |
| CUPONES .....                         | 9  |
| JAVA ORIENTADO A OBJETOS .....        | 9  |
| PROGRAMACIÓN CON JAVA JDBC.....       | 10 |
| PROGRAMACIÓN CON ORACLE PL/SQL.....   | 11 |



# 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

### CUPONES

En esta URL se publican cupones de descuento:

<https://github.com/gcoronelc/UDEMY>

## JAVA ORIENTADO A OBJETOS



### CURSO PROFESIONAL DE JAVA ORIENTADO A OBJETOS

---

**Eric Gustavo Coronel Castillo**

[www.desarrollasoftware.com](http://www.desarrollasoftware.com)

**I N S T R U C T O R**

En este curso aprenderás a crear software aplicando la Orientación a objetos, la programación en capas, el uso de patrones de software y swing.

Cada tema está desarrollado con ejemplos que demuestran los conceptos teóricos y finalizan con un proyecto aplicativo.

URL del Curso: <https://bit.ly/2B3ixUW>

Avance del curso: <https://bit.ly/2RYGXIt>

Cupones de descuento: <https://github.com/gcoronelc/UDEMY>



# ORACLE PL/SQL



## PROGRAMACIÓN CON JAVA JDBC



### PROGRAMACIÓN DE BASE DE DATOS ORACLE CON JAVA JDBC

---

**Eric Gustavo Coronel Castillo**

[www.desarrollasoftware.com](http://www.desarrollasoftware.com)

**I N S T R U C T O R**

En este curso aprenderás a programar bases de datos Oracle con JDBC utilizando los objetos Statement, PreparedStatement, CallableStatement y a programar transacciones correctamente teniendo en cuenta su rendimiento y concurrencia.

Al final del curso se integra todo lo desarrollado en una aplicación de escritorio.

URL del Curso: <https://bit.ly/31apy0O>

Avance del curso: <https://bit.ly/2vatZOT>

Cupones de descuento: <https://github.com/gcoronelc/UDEMY>

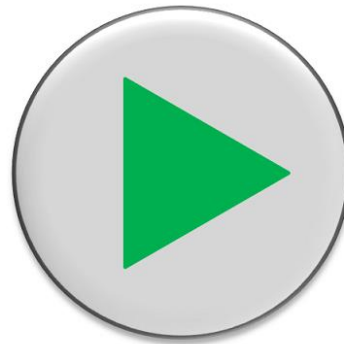


# ORACLE PL/SQL



## PROGRAMACIÓN CON ORACLE PL/SQL

# ORACLE PL/SQL



En este curso aprenderás a programar las bases de datos ORACLE con PL/SQL, de esta manera estarás aprovechando las ventajas que brinda este motor de base de datos y mejorarás el rendimiento de tus consultas, transacciones y la concurrencia.

Los procedimientos almacenados que desarrolles con PL/SQL se pueden ejecutarlos de Java, C#, PHP y otros lenguajes de programación.

URL del Curso: <https://bit.ly/2YZjfxT>

Avance del curso: <https://bit.ly/3bciqYb>

Cupones de descuento: <https://github.com/gcoronelc/UDEMY>