

CURSO DE JAVA CON JDBC

EJERCICIO

FUNCIONES CON CALLABLE STATEMENT DE JDBC



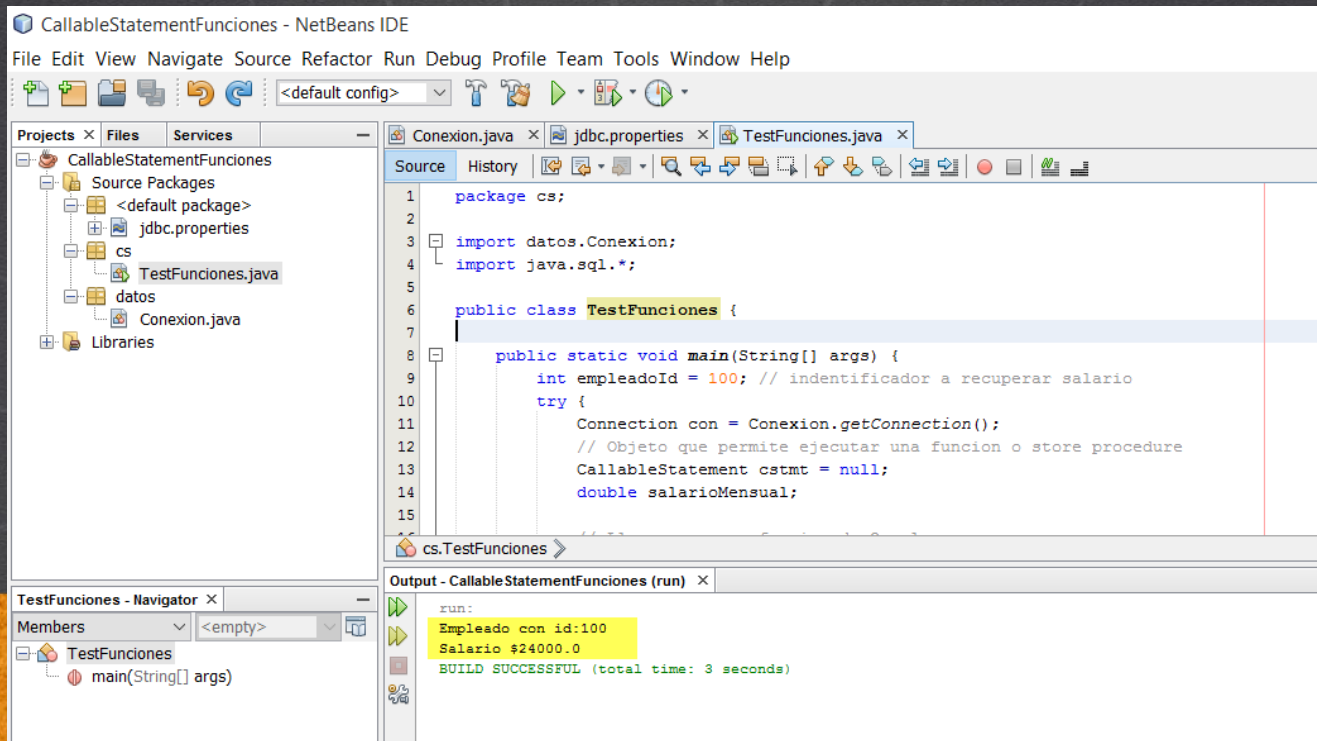
Experiencia y Conocimiento para tu vida

CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx

OBJETIVO DEL EJERCICIO

Vamos a crear una función de Oracle para poner en práctica el tema de Callable Statement con JDBC. Al finalizar veremos:



The screenshot displays the NetBeans IDE interface for a project named 'CallableStatementFunciones'. The 'TestFunciones.java' file is open in the editor, showing the following code:

```
1 package cs;
2
3 import datos.Conexion;
4 import java.sql.*;
5
6 public class TestFunciones {
7
8     public static void main(String[] args) {
9         int empleadoId = 100; // indentificador a recuperar salario
10        try {
11            Connection con = Conexion.getConnection();
12            // Objeto que permite ejecutar una funcion o store procedure
13            CallableStatement cstmt = null;
14            double salarioMensual;
15        }
16    }
17 }
```

The 'TestFunciones - Navigator' window shows the 'main(String[] args)' method. The 'Output - CallableStatementFunciones (run)' window displays the execution results:

```
run:
Empleado con id:100
Salario $24000.0
BUILD SUCCESSFUL (total time: 3 seconds)
```

PASO 1. COPIAMOS EL CÓDIGO DE LA FUNCIÓN DE ORACLE

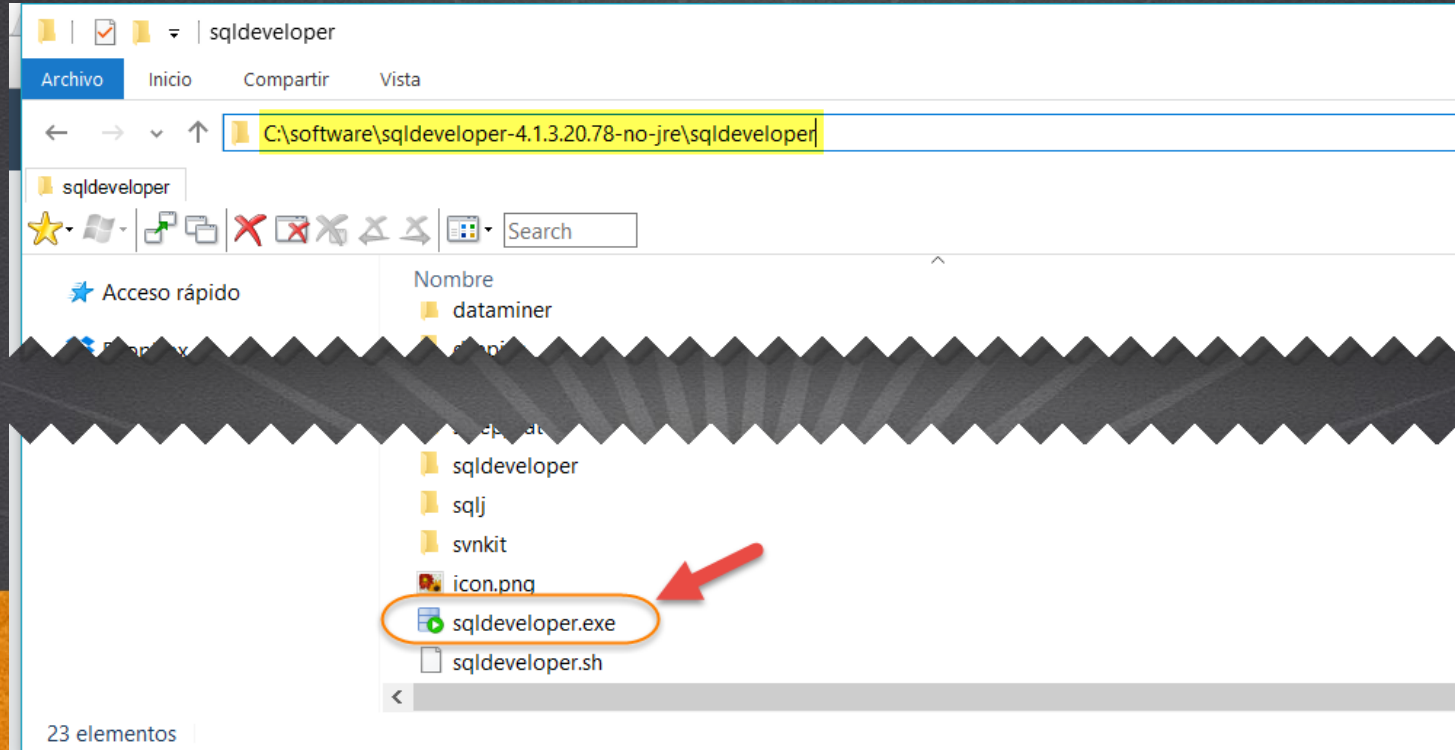
Archivo funcionEmployeeSalary.sql:

```
CREATE OR REPLACE FUNCTION "HR"."GET_EMPLOYEE_SALARY" ( p_emp_id IN employees.employee_id%TYPE )
RETURN employees.salary%TYPE
AS
    v_monthly_salary employees.salary%TYPE;
BEGIN
    --Ejecuta un select para obtener el salario actual para
    --el id_empleado proporcionado
    SELECT NVL(salary, -999)
    INTO v_monthly_salary
    FROM employees
    WHERE
        employee_id = p_emp_id;

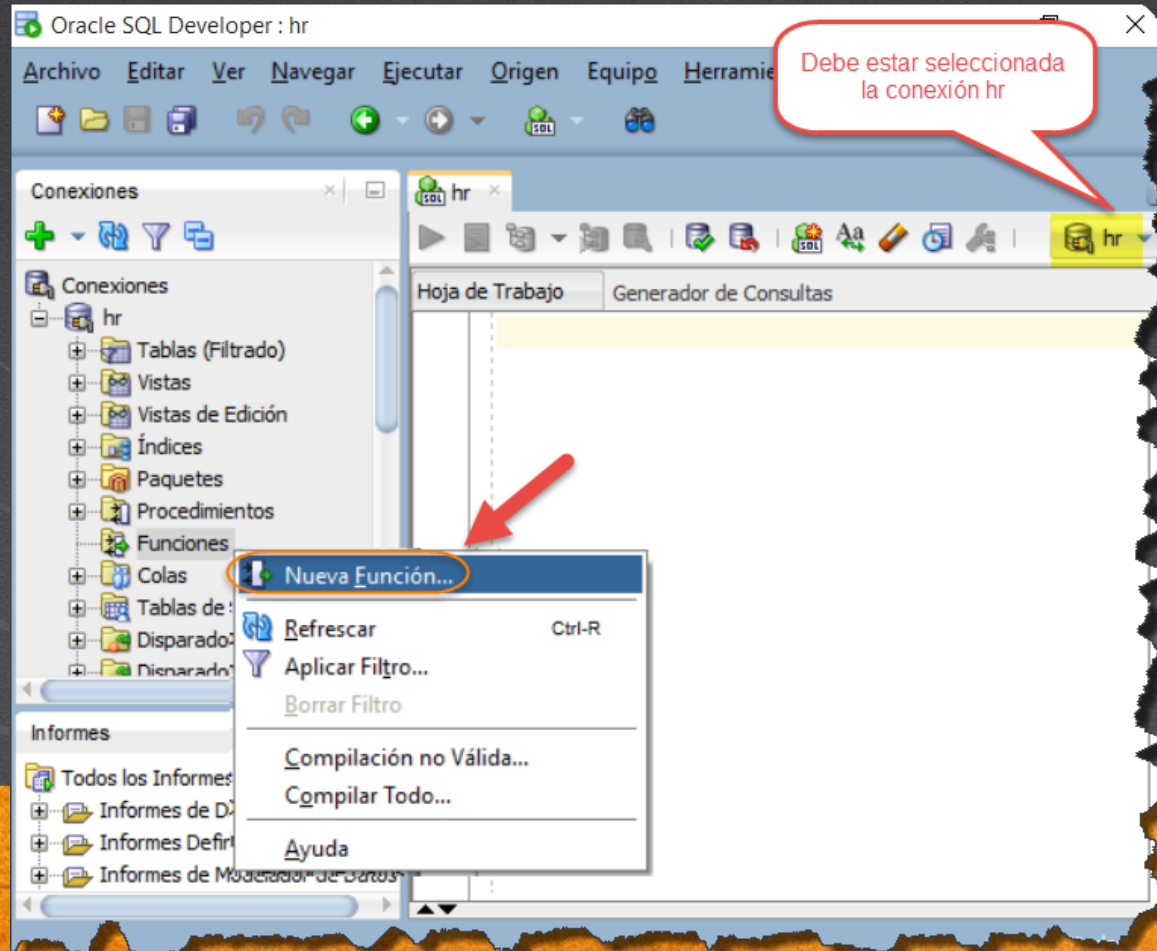
    RETURN v_monthly_salary;
END;
/
```


PASO 2. ABRIMOS SQL DEVELOPER

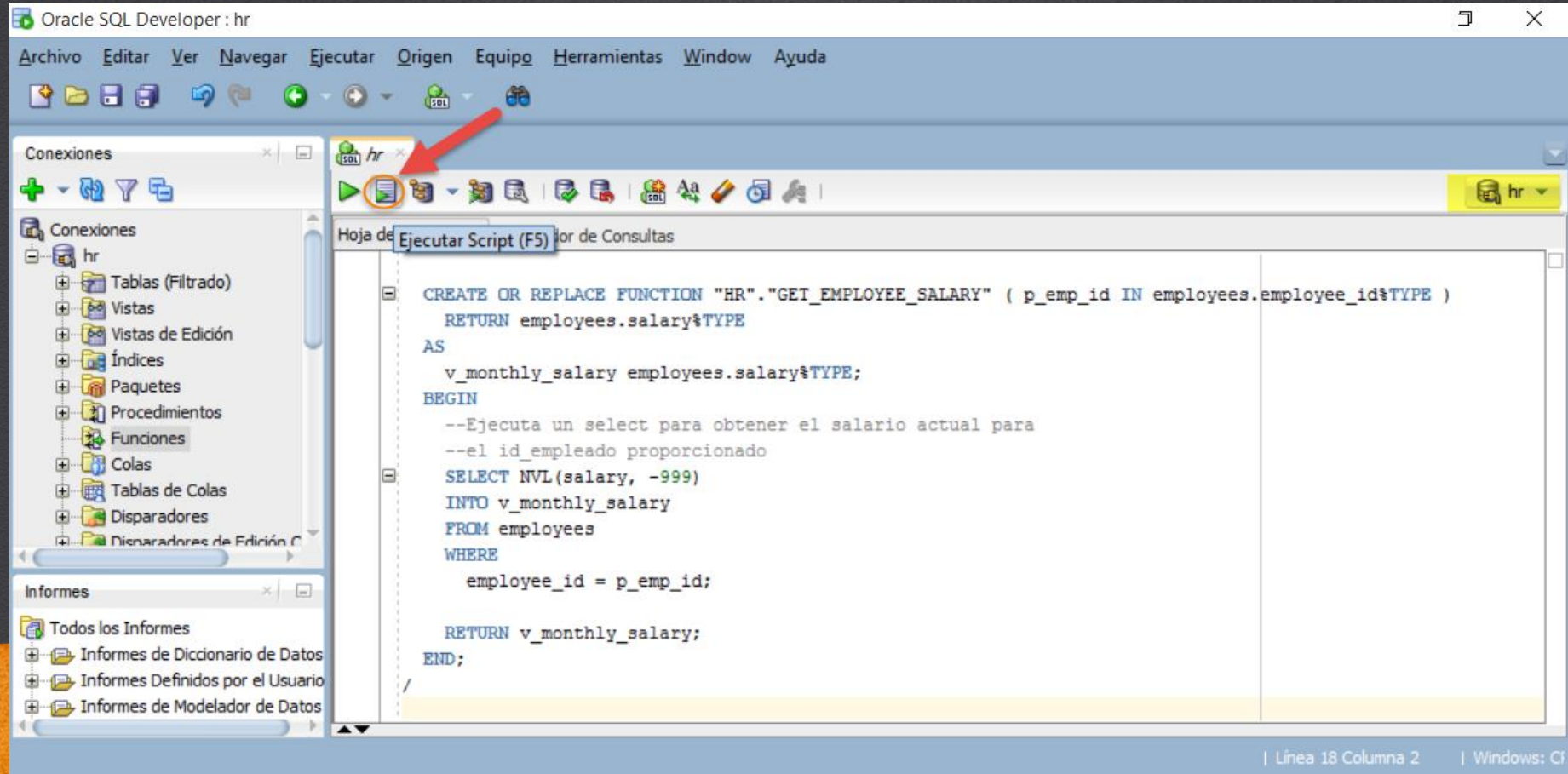
Podemos crear un acceso directo o abrir directamente el programa de SQL Developer:



PASO 3. PEGAMOS LA FUNCIÓN



PASO 4. EJECUTAMOS LA FUNCIÓN

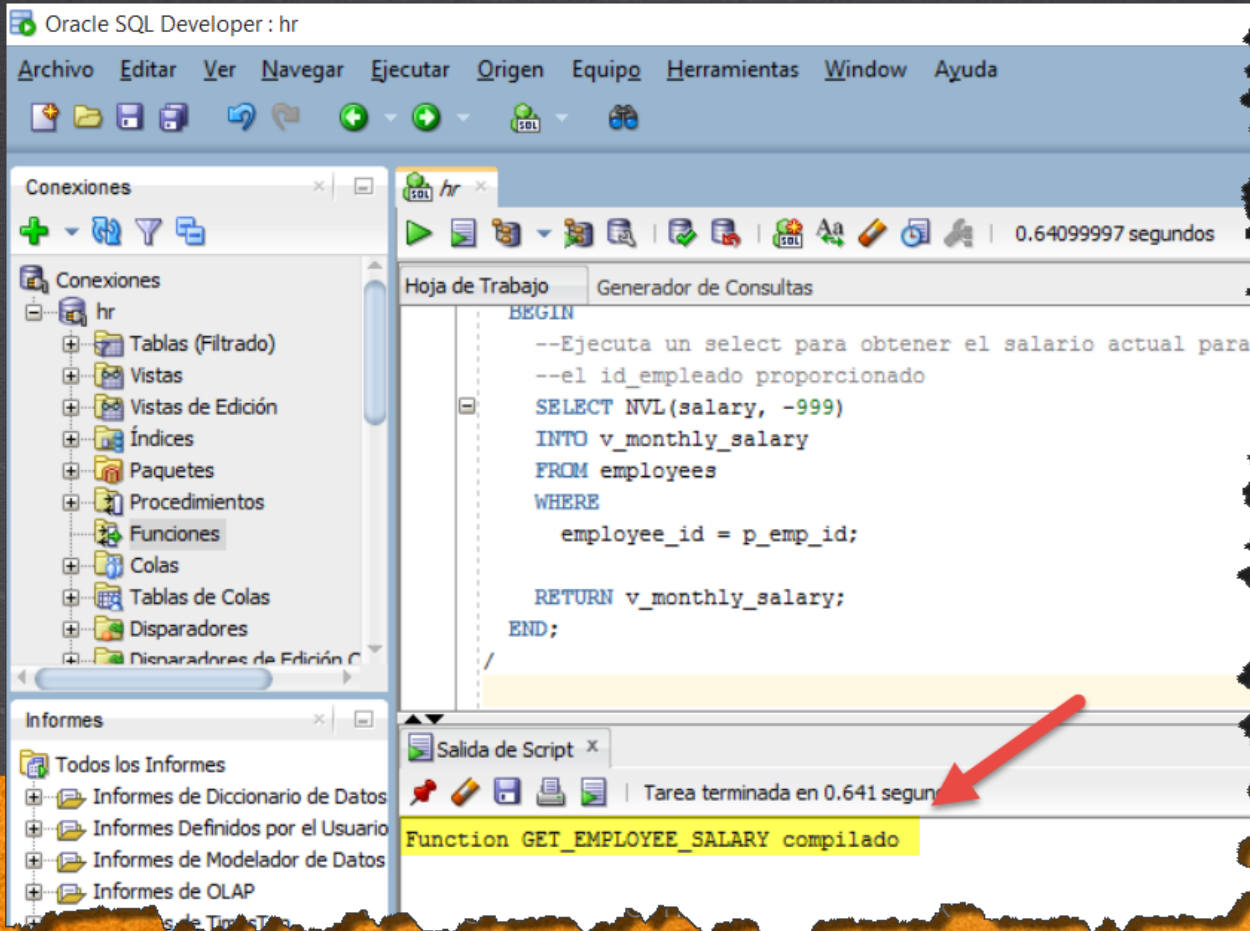


The screenshot displays the Oracle SQL Developer interface. The main window shows a script for creating or replacing a function named "GET_EMPLOYEE_SALARY". A red arrow points to the "Execute" button (a green play icon) in the toolbar. A tooltip "Ejecutar Script (F5)" is visible over this button. The script in the editor is as follows:

```
CREATE OR REPLACE FUNCTION "HR"."GET_EMPLOYEE_SALARY" ( p_emp_id IN employees.employee_id%TYPE )  
    RETURN employees.salary%TYPE  
AS  
    v_monthly_salary employees.salary%TYPE;  
BEGIN  
    --Ejecuta un select para obtener el salario actual para  
    --el id_empleado proporcionado  
    SELECT NVL(salary, -999)  
    INTO v_monthly_salary  
    FROM employees  
    WHERE  
        employee_id = p_emp_id;  
  
    RETURN v_monthly_salary;  
END;  
/
```

The status bar at the bottom right indicates "Línea 18 Columna 2" and "Windows: C".

PASO 4. EJECUTAMOS LA FUNCIÓN

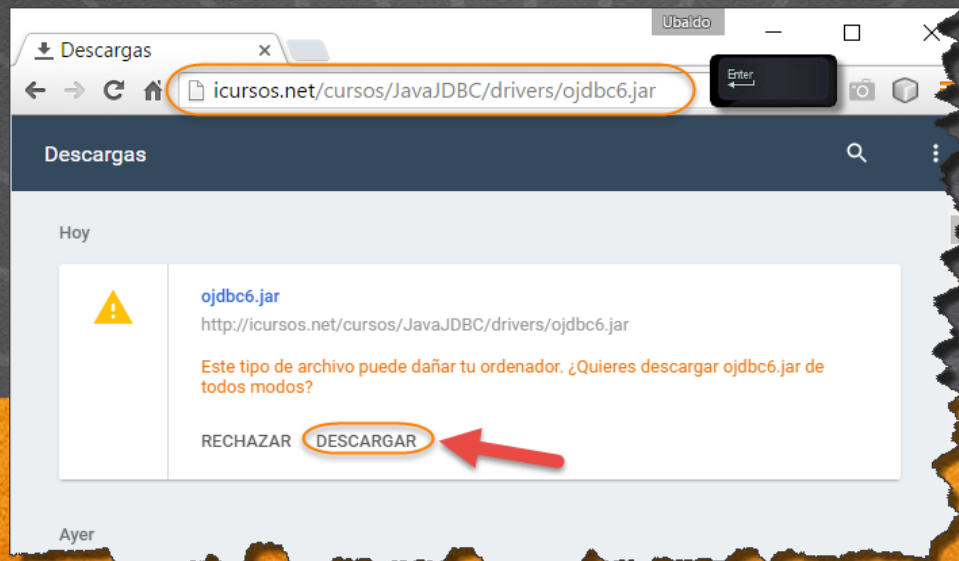


PASO 5. DESCARGAR EL DRIVER DE ORACLE

Descargar el driver de Oracle del link:

<http://icursos.net/cursos/JavaJDBC/drivers/ojdbc6.jar>

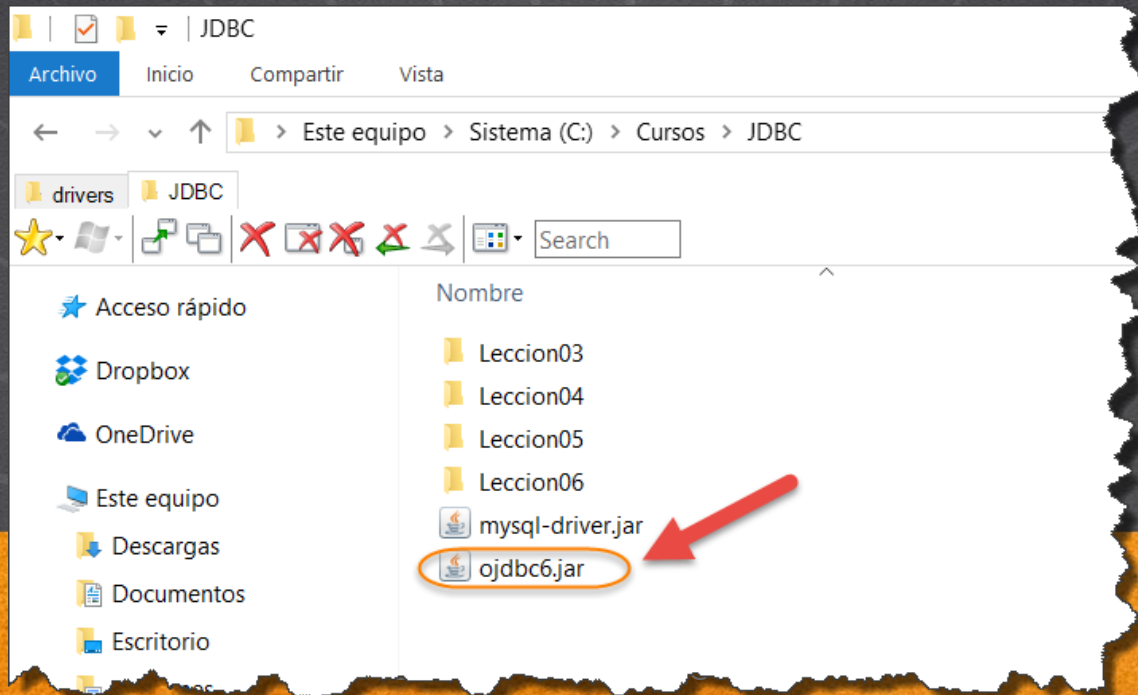
Aceptar la descarga en caso de que pregunte:



PASO 6. GUARDAR EL DRIVER DE ORACLE

Guardamos el driver de Oracle en alguna carpeta, por ejemplo:

C:\Cursos\JDBC



PASO 7. CREAMOS UN PROYECTO JAVA

New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name:

Project Location:

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

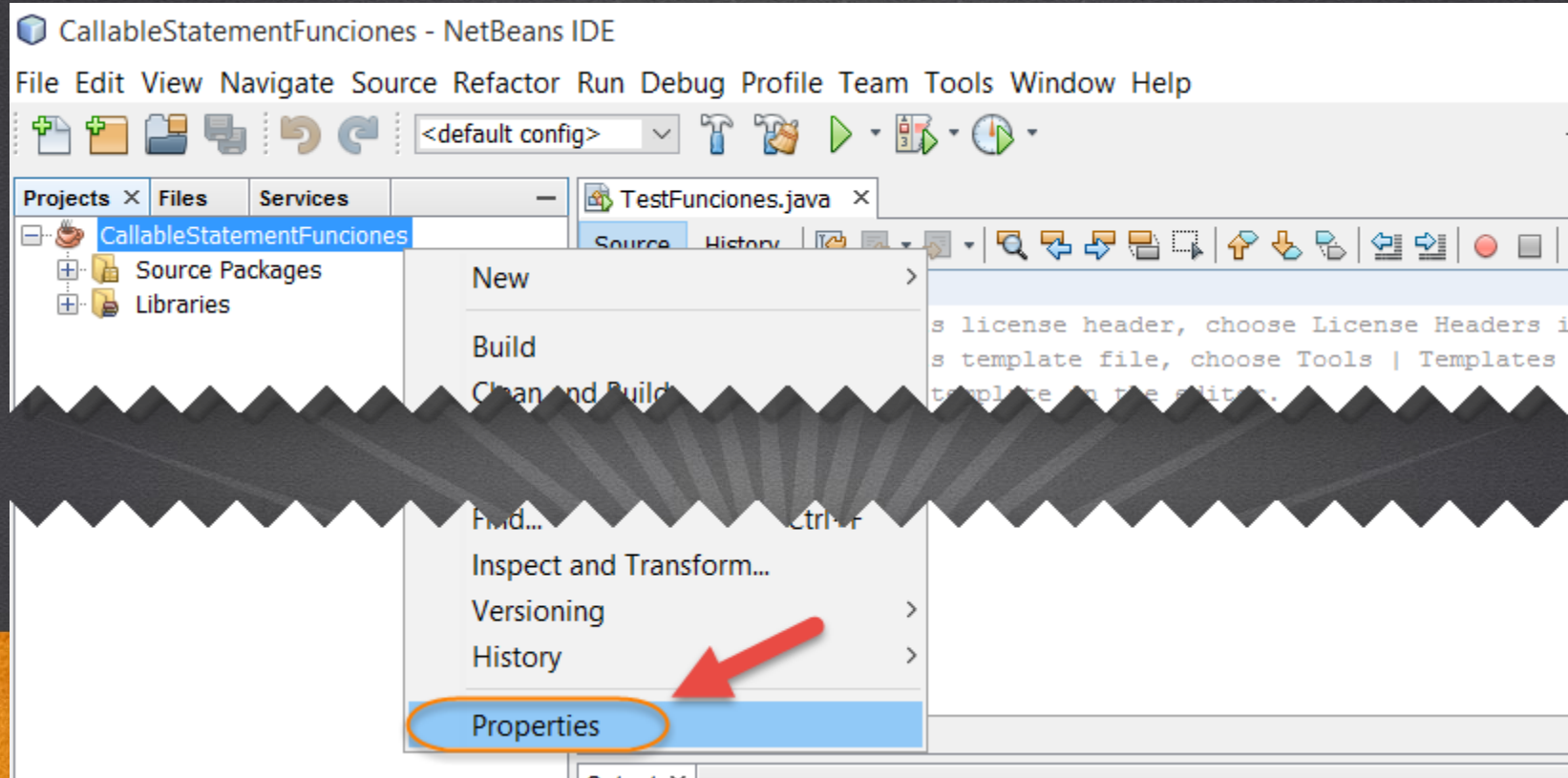
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☒ Create Main Class

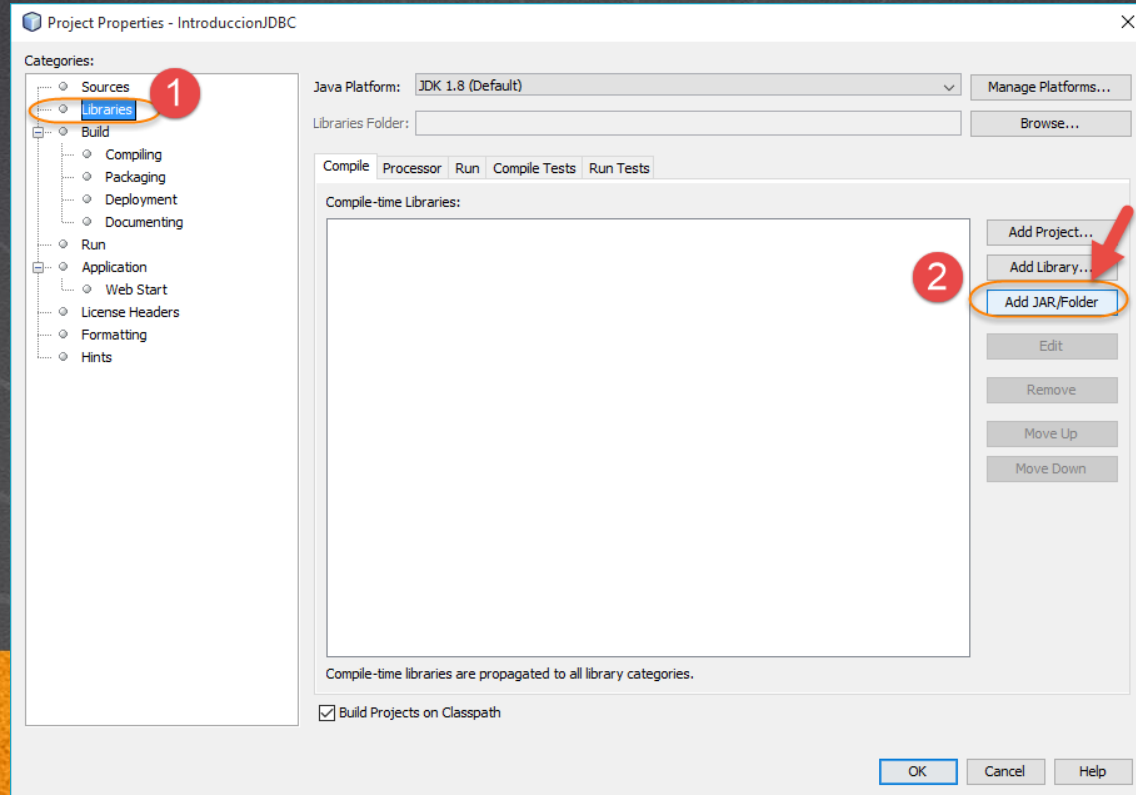
PASO 8. AGREGAR EL DRIVER AL CLASSPATH

Agregamos el driver al classpath de la aplicación como sigue:



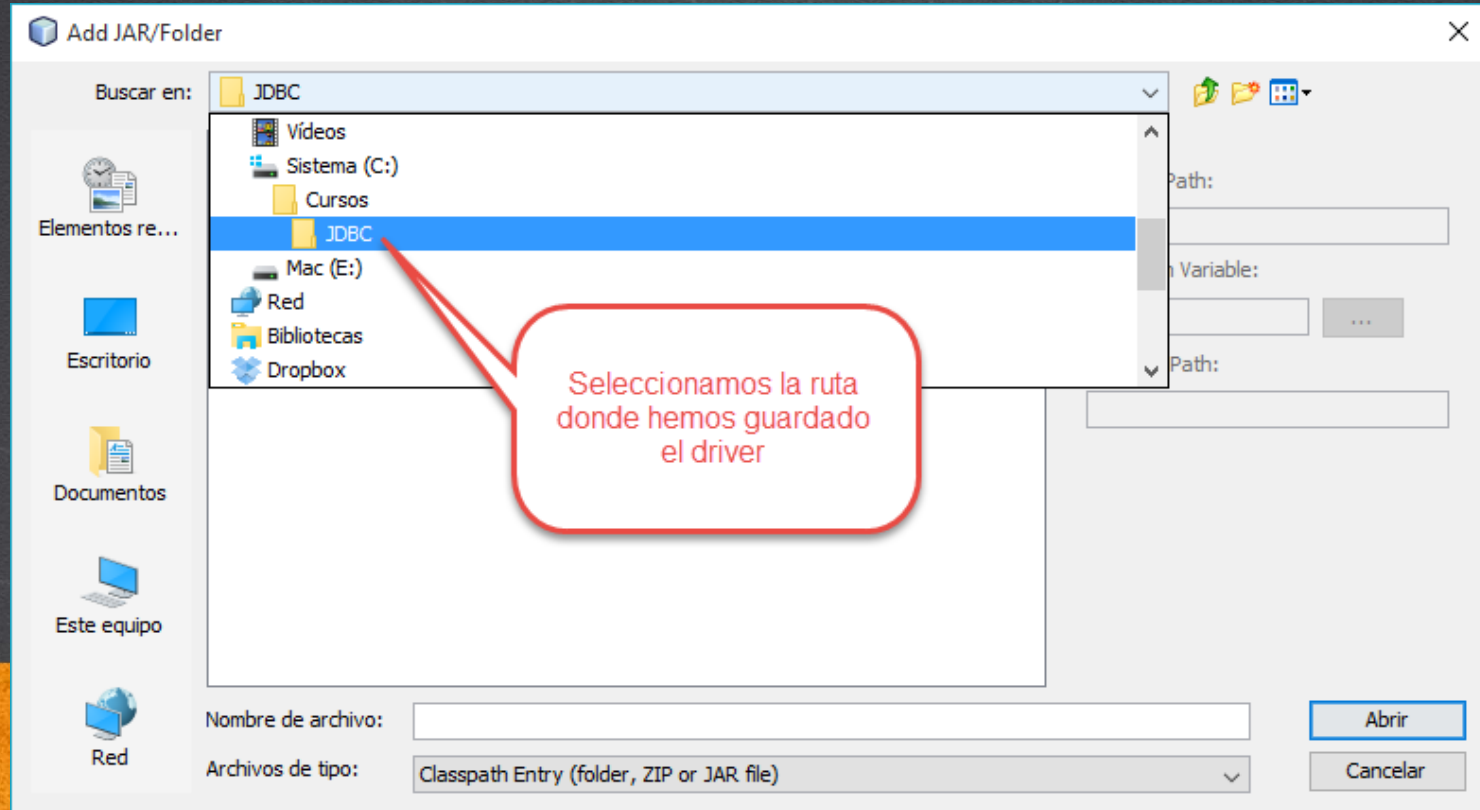
PASO 8. AGREGAR EL DRIVER AL CLASSPATH (CONT)

Agregamos el driver al classpath de la aplicación como sigue:



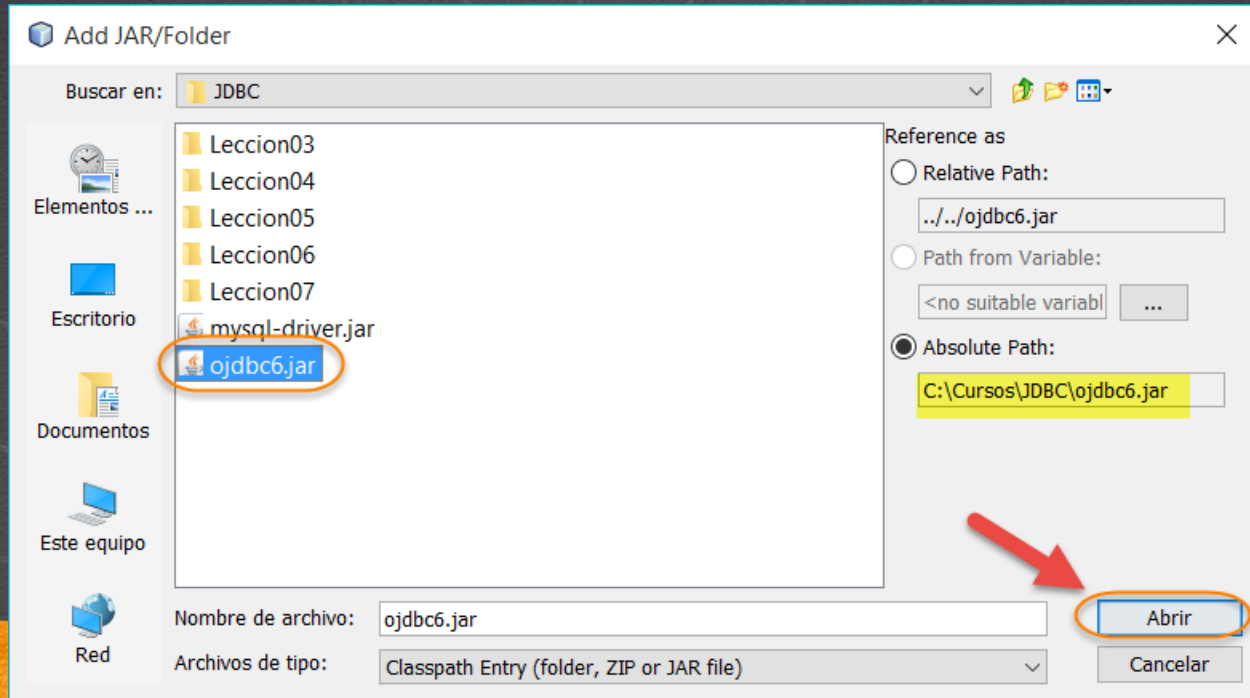
PASO 8. AGREGAR EL DRIVER AL CLASSPATH (CONT)

Agregamos el driver al classpath de la aplicación como sigue:



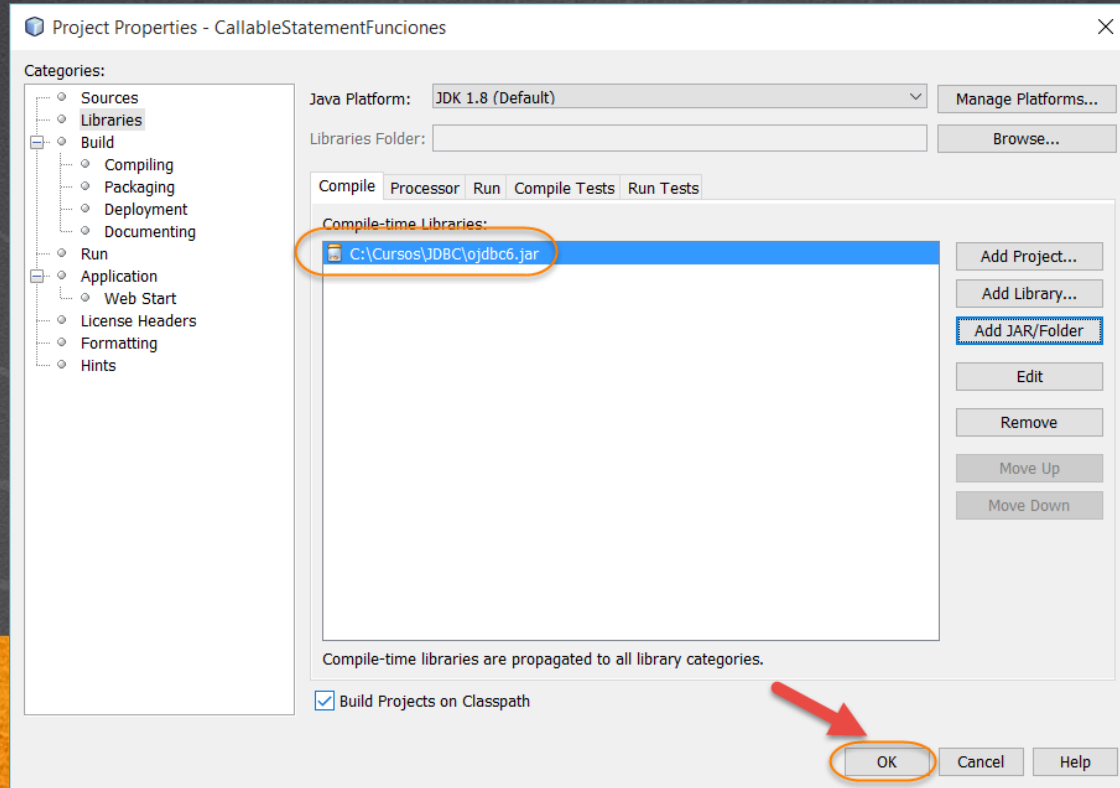
PASO 8. AGREGAR EL DRIVER AL CLASSPATH (CONT)

Agregamos el driver al classpath de la aplicación como sigue:



PASO 8. AGREGAR EL DRIVER AL CLASSPATH (CONT)

Agregamos el driver al classpath de la aplicación como sigue:



PASO 9. CREAMOS UNA CLASE

Agregamos una nueva clase:

New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

PASO 10. MODIFICAMOS EL CÓDIGO

Archivo Conexion.java:

```
package datos;

import java.sql.*;
import java.util.*;

public class Conexion {
    private static String JDBC_DRIVER;
    private static String JDBC_URL;
    private static String JDBC_USER;
    private static String JDBC_PASS;
    private static Driver driver = null;
    private static String JDBC_FILE_NAME= "jdbc";

    public static Properties loadProperties(String file){
        prop = new Properties();
        bundle = ResourceBundle.getBundle(file);
        e = bundle.getKeys();
        key = null;

        while(e.hasMoreElements()){
            = (String) e.nextElement();
            .put(key, bundle.getObject(key));
        }

        JDBC_DRIVER = prop.getProperty("driver");
        JDBC_URL = prop.getProperty("url");
        JDBC_USER = prop.getProperty("user");
        JDBC_PASS = prop.getProperty("pass");
        return prop;
    }
}
```


PASO 10. MODIFICAMOS EL CÓDIGO

Archivo Conexion.java:

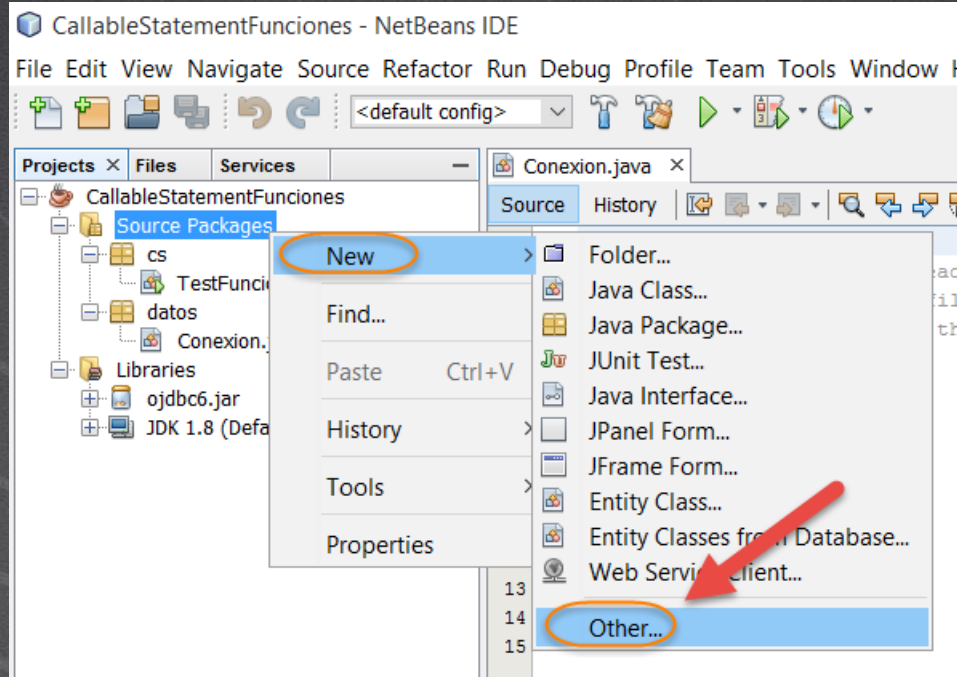
```
public static synchronized Connection getConnection()
    throws SQLException {
    if (driver == null) {
        try {
            //Cargamos las propiedades de conexion a la BD
            loadProperties(JDBC_FILE_NAME);
            //Se registra el driver
            Class jdbcDriverClass = Class.forName(JDBC_DRIVER);
            driver = (Driver) jdbcDriverClass.newInstance();
            DriverManager.registerDriver(driver);
        } catch (Exception e) {
            System.out.println("Fallo en cargar el driver JDBC");
            e.printStackTrace();
        }
    }
    return DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASS);
}

public static void close(ResultSet rs) {
    try {
        if (rs != null) {
            rs.close();
        }
    } catch (SQLException sqle) {
        sqle.printStackTrace();
    }
}
```

```
//Cierre del PreparedStatement
public static void close(PreparedStatement stmt) {
    try {
        if (stmt != null) {
            stmt.close();
        }
    } catch (SQLException sqle) {
        sqle.printStackTrace();
    }
}

//Cierre de la conexion
public static void close(Connection conn) {
    try {
        if (conn != null) {
            conn.close();
        }
    } catch (SQLException sqle) {
        sqle.printStackTrace();
    }
}
```

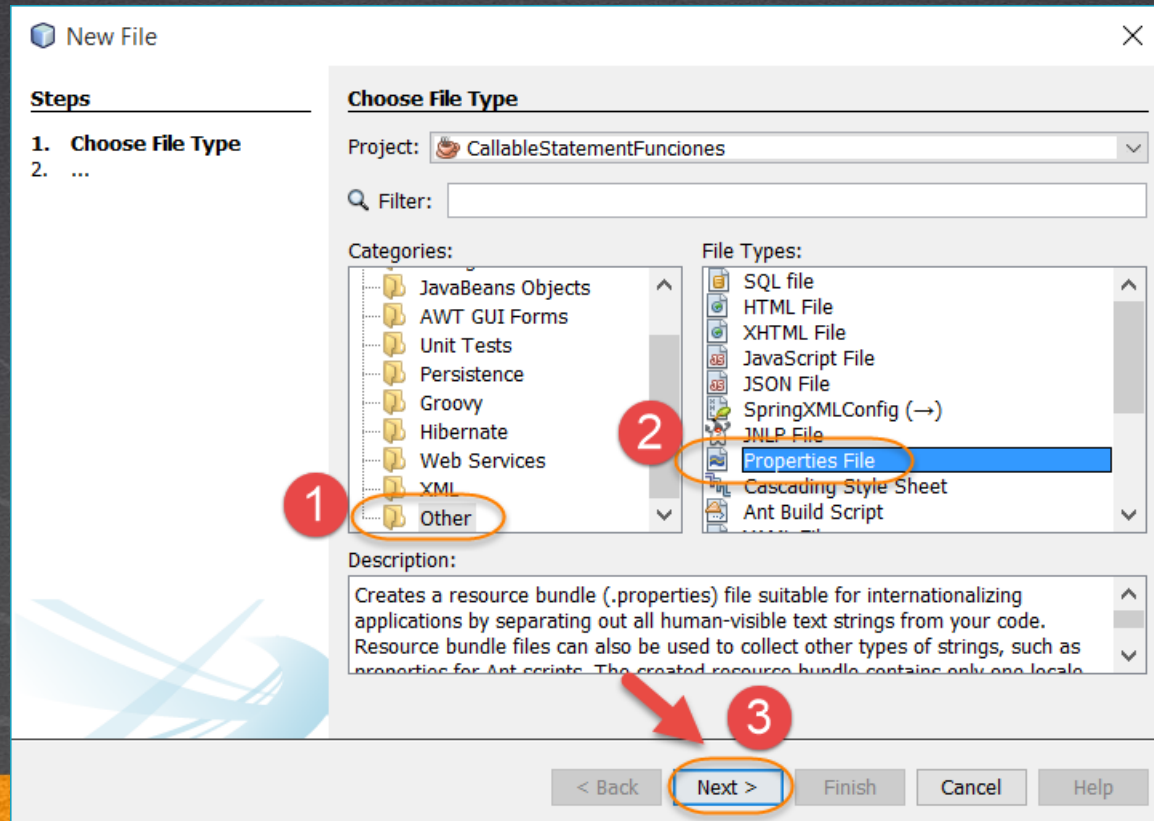
PASO 11. CREAMOS UN ARCHIVO DE PROPIEDADES



CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx

PASO 11. CREAMOS UN ARCHIVO DE PROPIEDADES



CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx

PASO 11. CREAMOS UN ARCHIVO DE PROPIEDADES

New Properties File

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

File Name:

Project:

Folder:

Created File:

< Back Next > **Finish** Cancel Help

CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx

PASO 12. MODIFICAMOS EL ARCHIVO

Archivo jdbc.properties:

#Archivo de propiedades que contiene los valores
#de la cadena de conexion

#Conexion a Oracle

```
driver = oracle.jdbc.driver.OracleDriver
url    = jdbc:oracle:thin:@localhost:1521:XE
user   = hr
pass   = hr
```

PASO 13. MODIFICAMOS EL CÓDIGO

Archivo TestFunciones.java:

```
package cs;

import datos.Conexion;
import java.sql.*;

public class TestFunciones {

    public static void main(String[] args) {
        int empleadoId = 100; // indentificador a recuperar salario
        try {
            Connection con = Conexion.getConnection();
            CallableStatement cstmt = null;
            double salarioMensual;

            cstmt = con.prepareCall("{ ? = call get_employee_salary(?) }");
            // Una funcion regresa un valor
            // por lo que lo registramos como el parametro 1
            cstmt.registerOutParameter(1, java.sql.Types.INTEGER);
            // registremos el segundo parametro
            cstmt.setInt(2, empleadoId);
            cstmt.execute();
            salarioMensual = cstmt.getDouble(1);
            cstmt.close();
            System.out.println("Empleado con id:" + empleadoId);
            System.out.println("Salario $" + salarioMensual);

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```


PASO 14. EJECUTAR EL CÓDIGO

CallableStatementFunciones - NetBeans IDE

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

<default config>

Projects Files Services

- CallableStatementFunciones
 - Source Packages
 - <default package>
 - jdbc.properties
 - cs
 - TestFunciones.java
 - datos
 - Conexion.java
 - Libraries

Conexion.java x jdbc.properties x TestFunciones.java x

Source History

```
1 package cs;
2
3 import datos.Conexion;
4 import java.sql.*;
5
6 public class TestFunciones {
7
8     public static void main(String[] args) {
9         int empleadoId = 100; // identificador a recuperar salario
10        try {
11            Connection con = Conexion.getConnection();
12            // Objeto que permite ejecutar una funcion o store procedure
13            CallableStatement cstmt = null;
14            double salarioMensual;
15
```

cs.TestFunciones

TestFunciones - Navigator

Members <empty>

- TestFunciones
 - main(String[] args)

Output - CallableStatementFunciones (run)

run:

Empleado con id:100
Salario \$24000.0

BUILD SUCCESSFUL (total time: 3 seconds)

PASO 15. VERIFICAMOS RESULTADO

Oracle SQL Developer : Tabla HR.EMPLOYEES@hr

Archivo Editar Ver Navegar Ejecutar Equipo Herramientas Window Ayuda

Conexiones

Conexiones

hr

Tablas (Filtrado)

- COUNTRIES
- DEPARTMENTS
- EMPLOYEES
- JOB_HISTORY
- JOBS
- LOCATIONS

Página Inicial hr EMPLOYEES

Columnas Datos Model Restricciones Permisos Estadísticas Disparadores Flashback Dependencias Detalles Particiones Índices SQL

Ordenar... Filtrar:

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT
1	100	Steven	King	SKING	515.123.4567	17/06/03	AD_PRES	24000	
2	101	Neena	Kochhar	NKOCHHAR	515.123.4568	21/09/05	AD_VP	17000	
3	102	Lex	De Haan	LDEHAAN	515.123.4569	13/01/01	AD_VP	17000	
4	103	Alexander	Hunold	AHUNOLD	590.423.4567	03/01/06	IT_PROG	9000	
5	104	Bruce	Ernst	BERNST	590.423.4568	21/05/07	IT_PROG	6000	
6	105	David	Turner	DTURNER	590.423.4569	06/06/06	IT_PROG	4000	

CONCLUSIÓN DEL EJERCICIO

- Con este ejercicio pusimos en práctica el concepto de Callable Statement, y en particular cómo ejecutar una función de Oracle.
- Para crear la función utilizamos el lenguaje de PL/SQL, el cual queda fuera del alcance de este curso. Sin embargo lo importante es cómo mandar a llamar cualquier función de Oracle, que es una de las tareas más comunes que nos encontraremos en nuestro día a día como programadores Java.



Experiencia y Conocimiento para tu vida

CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx

CURSO ONLINE

JAVA CON JDBC

Por: Ing. Ubaldo Acosta



Experiencia y Conocimiento para tu vida

CURSO DE JAVA CON JDBC

www.globalmentoring.com.mx