EJECUCIÓN DE PROCEDIMIENTOS Y GESTIÓN DE ERRORES

Javier García-Retamero Redondo

EJECUCIÓN DE PROCEDIMIENTOS

Declaración de llamadas

CONCEPTO

- Un procedimiento es un conjunto de sentencias SQL que se puede llamar por su nombre para llevar a cabo alguna tarea.
- Normalmente reciben parámetros para ser más versátiles.
- Si devuelven un valor se denominan Funciones.

EJECUCIÓN DE PROCEDIMIENTOS

- Declaración de llamadas a un procedimiento o función:

Declaración	Para llamar a
{call nombre_procedimiento}	Procedimiento almacenado sin parámetros
{?=call nombre_procedimiento}	Función almacenada que devuelve un valor y no recibe parámetros
{call nombre_procedimiento(?,?,...)}	Procedimiento almacenado que recibe parámetros
{?=call nombre_procedimiento(?,?,...)}	Función almacenada que devuelve un valor y recibe varios parámetros

EJECUCIÓN DE PROCEDIMIENTOS

Function nombre_dep(d NUMBER, local OUT VARCHAR2)
RETURN VARCHAR2

- **Preparamos el string para la llamada:**

```
String sql = "{?=call nombre_dep (?,?)}";
              1                    2 3
```

- **Creamos un objeto llamando al método prepareCall:**

```
CallableStatement llamada=conexión.prepareCall(sql);
```

- **Registramos el parámetro de salida de la función:**

```
llamada.registerOutParameter(1,Types.VARCHAR);
```

- **Registramos los parámetros de salida (OUT) de la función:**

```
llamada.registerOutParameter(3, Types.VARCHAR);
```

- **Le damos valor al parámetro de entrada:**

```
llamada.setInt(2,Integer.parseInt(dep));
```

- **Realizamos la llamada al procedimiento:**

```
llamada.executeUpdate();
```

EJECUCIÓN DE PROCEDIMIENTOS

CONSIDERACIONES

CONSIDERACIONES

- **El usuario debe tener permisos para ejecutar procedimientos y funciones:**

En Mysql: GRANT SELECT ON mysql.proc TO 'usuario'@'ip_servidor'
En Oracle: GRANT EXECUTE PROCEDURE TO usuario;

- **Si el procedimiento que queremos ejecutar no es propiedad del usuario entonces debemos cambiar la conexión:**

```
Connection conexion = DriverManager.getConnection(
        "jdbc:mysql://localhost/ejemplo? noAccessToProcedureBodies=True", "usuario",
        "contraseña");
```

- **Para obtener los valores devueltos por una función o procedimiento:**
  getXXX(número_parámetro) como hacíamos con los ResulSet

```
String salida1 = llamada.getString(1);
String salida2 = llamada.getString(3);
```

CONSIDERACIONES

- **Introducción de la fecha del sistema como parámetro de entrada:**

```
java.util.Date utilDate=new java.util.Date();
java.sql.Date sqlDate = new java.sql.Date(utilDate.getTime());
…..
sentencia.setDate(2,sqlDate);
```

GESTIÓN DE ERRORES

CONSIDERACIONES

CONSIDERACIONES

- **Con SQLException podemos acceder a cierta información:**

```
catch (SQLException e) {
        System.out.println ("Ha ocurrido un error:");
        System.out.println ("Mensaje: " +e.getMessage());
        System.out.println ("SQL Estado: " +e.getSQLState());
        System.out.println ("Código de error: " +e.getErrorCode());
}
```

Método	Función
getMessage()	Mensaje que describe el error.
getSQLState()	Estado definido por el estándar X/OPEN SQL
getErrorCode()	Muestra el código de error que se ha producido en la BBDD. Por ejemplo en Oracle correspondería con el número que aparece en los ORA-