

Conexión Base de Datos

Lenguajes de Programación 2



PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ


Dr. Freddy Paz

Procedimiento Almacenado

- Es un programa dentro de la base de datos que ejecuta una acción o conjunto de acciones específicas.
- Un procedimiento tiene un nombre, un conjunto de parámetros y un bloque de código.



Ejemplo Creación (MySQL)



```
DELIMITER $$  
CREATE PROCEDURE insertarEmpleado(  
    IN _dni VARCHAR(8),  
    IN _nombres VARCHAR(100),  
    IN _apellido_paterno VARCHAR(100),  
    IN _apellido_materno VARCHAR(100))  
  
BEGIN  
    INSERT INTO empleado (dni, nombres, apellido_paterno,  
    apellido_materno) VALUES (_dni, _nombres, _apellido_paterno,  
    _apellido_materno);  
END
```

CallableStatement

- Es la interfaz utilizada para ejecutar procedimientos SQL almacenados. Los parámetros de entrada se establecen utilizando el método *set*.
Un objeto de tipo *CallableStatement* puede retornar un objeto de tipo *ResultSet* o un *boolean*. Los parámetros de salida deben registrarse antes de la ejecución del procedimiento almacenado y ser recuperados después de la ejecución.

CallableStatement



```
import java.sql.CallableStatement;
```

```
Class.forName("com.mysql.jdbc.Driver");
```

```
Connection con =
```

```
DriverManager.getConnection("jdbc:mysql://50.62.209.73/prueba", "prueba",  
"lp2");
```

```
CallableStatement cStmt = con.prepareCall("{call  
insertarEmpleado(?,?,?,?)}");
```

```
cStmt.setString(1, "29871543");
```

```
cStmt.setString(2, "Freddy Alberto");
```

```
cStmt.setString(3, "Paz");
```

```
cStmt.setString(4, "Espinoza");
```

```
cStmt.execute();
```

CallableStatement



```
import java.sql.CallableStatement;
```

```
Class.forName("com.mysql.jdbc.Driver");
```

```
    Connection con =
```

```
    DriverManager.getConnection("jdbc:mysql://50.62.209.73/prueba", "prueba",  
    "lp2");
```

```
    CallableStatement cStmt = con.prepareCall("{call  
insertarEmpleado(?,?,?,?)}");
```

```
    cStmt.setString("_dni", "12456111");
```

```
    cStmt.setString("_nombres", "Freddy Alberto");
```

```
    cStmt.setString("_apellido_paterno", "Paz");
```

```
    cStmt.setString("_apellido_materno", "Espinoza");
```

```
    cStmt.execute();
```

Ejemplo Creación Tabla



```
CREATE TABLE empleado2 (  
    id INT AUTO_INCREMENT,  
    nombres VARCHAR(50),  
    apellido_paterno VARCHAR(50),  
    apellido_materno VARCHAR(50),  
    PRIMARY KEY (id)  
);
```

Ejemplo PA con parámetro salida



DELIMITER \$\$

CREATE PROCEDURE INSERTAR_EMPLEADO_2(

OUT _id INT,

IN _nombres VARCHAR(100),

IN _apellido_paterno VARCHAR(100),

IN _apellido_materno VARCHAR(100))

BEGIN

**INSERT INTO empleado2 (nombres, apellido_paterno,
apellido_materno) VALUES (_nombres, _apellido_paterno,
_apellido_materno);**

SET _id = last_insert_id();

END

Ejecutar Procedimiento (MySQL)



SET @ID = 0;

**CALL INSERTAR_EMPLEADO_2(@ID, 'Juan', 'Perez',
'Sandoval');**

SELECT @ID;

Ejemplo PA con parámetro salida – JAVA

```
Class.forName("com.mysql.jdbc.Driver");  
Connection con =  
DriverManager.getConnection("jdbc:mysql://50.62.209.73/prueba",  
"prueba", "lp2");  
CallableStatement cStmt = con.prepareCall("{call  
INSERTAR_EMPLEADO_2(?,?,?,?)}");  
cStmt.registerOutParameter("_id", java.sql.Types.INTEGER);  
cStmt.setString("_apellido_paterno", "Paz");  
cStmt.setString("_nombres", "Freddy Alberto");  
cStmt.setString("_apellido_materno", "Espinoza");  
cStmt.execute();  
int id = cStmt.getInt("_id");  
System.out.println(id);
```

Ejemplo PA con parámetro salida – C#



```
MySqlConnection con = new MySqlConnection(cadena);
con.Open();
MySqlCommand comando = new MySqlCommand();
comando.Connection = con;
comando.CommandType = CommandType.StoredProcedure;
comando.CommandText = "INSERTAR_PROFESOR";
comando.Parameters.Add("_nombre", MySqlDbType.VarChar).Value = "Juan";
comando.Parameters.Add("_apellido", MySqlDbType.VarChar).Value = "Perez";
comando.Parameters.Add("_id_profesor", MySqlDbType.Int32).Direction = ParameterDirection.Output;
comando.ExecuteNonQuery();
int numero = Int32.Parse(comando.Parameters["_id_profesor"].Value.ToString());
System.Console.WriteLine(numero);
System.Console.Read();
con.Close();
```

PA listar Empleados



DELIMITER \$\$


CREATE PROCEDURE LISTAR_EMPLEADOS()

BEGIN

SELECT * FROM empleado;


END

Listar Empleados - Código



```
Class.forName("com.mysql.jdbc.Driver");
Connection con =
DriverManager.getConnection("jdbc:mysql://50.62.209.73/prueba",
"prueba", "lp2");
CallableStatement cStmt = con.prepareCall("{call
LISTAR_EMPLEADOS()}");
ResultSet rs = cStmt.executeQuery();
while (rs.next()){
    String dni = rs.getString("dni");
    System.out.println(dni);
}
```

PA buscar Empleados



```
DELIMITER //  
CREATE PROCEDURE BUSCAR_EMPLEADO(  
    IN variable VARCHAR(50)  
)  
BEGIN  
    SELECT * from empleado where nombres like  
Concat('%',variable,'%');  
END
```

Creación de Tabla (MSSQL)



CREATE TABLE Cliente(

IdCliente INT PRIMARY KEY IDENTITY(1,1),

Nombre VARCHAR(100),

Apellido VARCHAR(100)

)

Ejemplo Creación PA (MSSQL)



```
CREATE PROCEDURE INSERTAR_CLIENTE
```

```
    @Id INT OUTPUT,
```

```
    @Nombre VARCHAR(100),
```

```
    @Apellido VARCHAR(100)
```

```
AS
```

```
    INSERT INTO Cliente(Nombre,Apellido)
```

```
    VALUES (@Nombre,@Apellido)
```

```
    SET @Id = @@IDENTITY
```

```
GO
```


Ejecutar Procedimiento (MSSQL)



DECLARE @ID INT

EXEC INSERTAR_CLIENTE @ID OUTPUT, 'Juan','Perez'

SELECT @ID

Referencias



- D.J. Barnes y M. Kölling, Programación orientada a objetos con Java. Pearson Educación, 2007
- T. Budd, An introduction to Object-Oriented Programming (Third Edition). Pearson Education, 2001
- E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994
- B. Stroustrup, The C++ Programming Language (Third Edition) Addison-Wesley, 1997
- Agustín Froufe. Java 2. Manual de usuario y tutorial. Ed. Ra-Ma
- J. Sánchez, G. Huecas, B. Fernández y P. Moreno, Iniciación y referencia: Java 2. Osborne McGraw-Hill, 2001.
- B. Meyer, Object-Oriented Software Construction (Second Edition). Prentice Hall, 1997.