



Relaciones

Sesión Conceptual 1





Inicio

{desafío}
latam_



15 minutos

- Conocer el patrón DAO y Singleton para la conexión.
- Conocer la IDE de BD Dbeaver.
- Conectar aplicaciones Java con base de datos.

Objetivo



Desarrollo

{desafío}
latam_



150 minutos

/* Java empresarial y las bases de datos */

Conceptos básicos sobre bases de datos

Conjunto de datos
almacenados,
organizados y
ordenados
sistemáticamente.

Representa
información de una
persona, cosa u
evento.

Organizada en
tablas,
columnas y
relaciones.

Instalación base de datos Oracle 18c express Edition

- Conoceremos dos buenos IDEs de base de datos, el DBEAVER y el SQL DEVELOPER.
- **Instalación en Windows.**
 - No debe existir ningún resto de instalación de alguna versión de Oracle.



Oracle Database Express Edition (XE) Release 18.4.0.0.0 (18c)

Thank you for accepting the OTN License Agreement.

● Oracle Database 18c Express Edition for Linux x64 [Download](#)
(2,574,155,124 bytes - October 19, 2018)
[Sha256sum: 308c044444342b9a3a8d332c58b12c540edf933dc8162d8eda3225e662433f1b]

● Oracle Database 18c Express Edition for Windows x64 [Download](#)
(2,054,204,100 bytes - February 20, 2019)
[Sha256sum: 7C1A85D05F6FCC5BE70E7BA4017BEEC70C8D096E685031695B0486C40C114F1A]

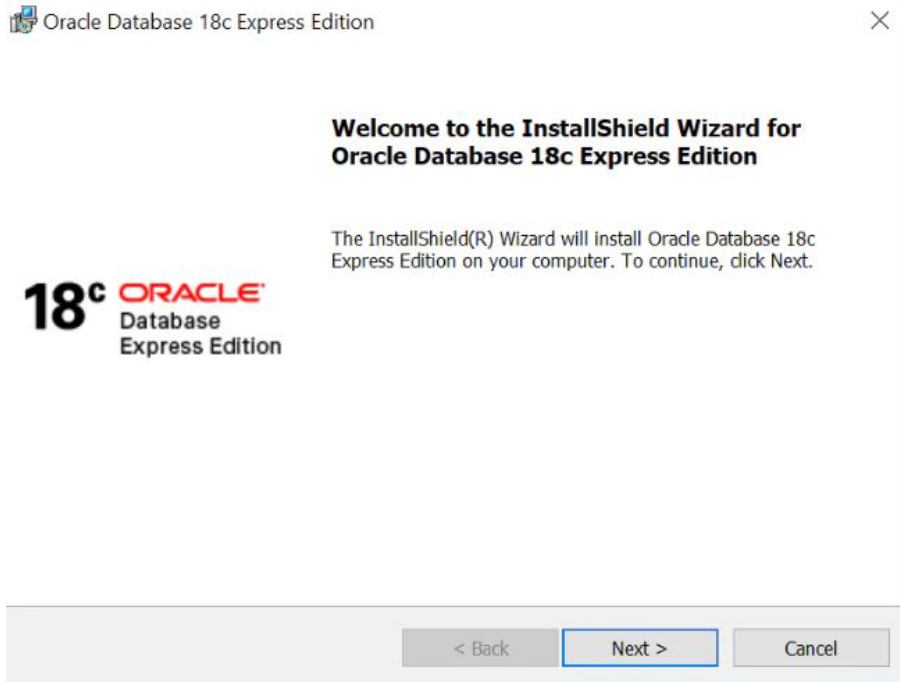
● Oracle Database Preinstall RPM for RHEL and CentOS

- [Release 7](#)
- [Release 6](#)

● Getting Started:

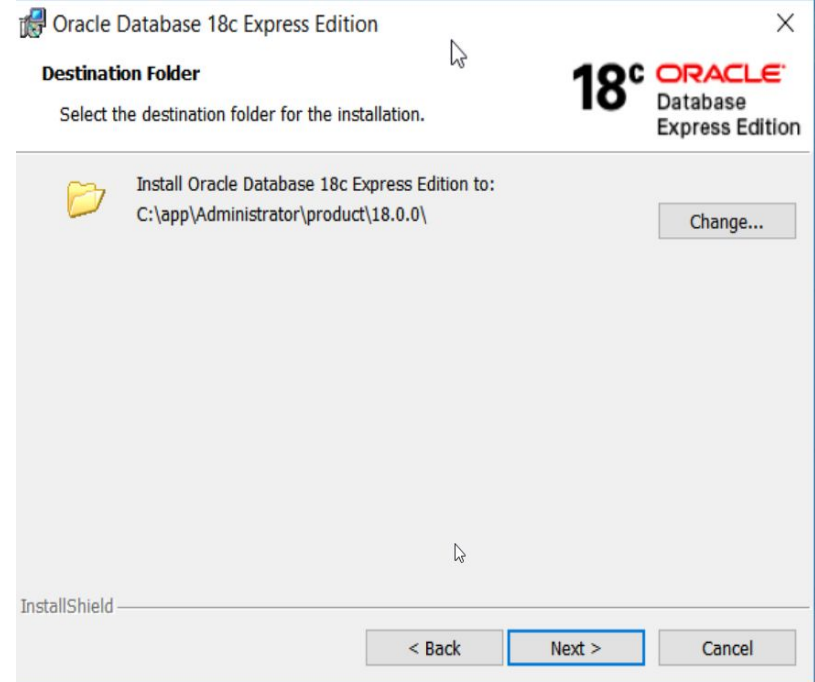
- [Quick Start](#)
- [Frequently Asked Questions](#)

Instalador



{desafío}
latam_

Eligiendo directorio



Los usuarios por defecto son:

- SYS
- SYSTEM o PDBADMIN (PDB es la sigla de pluggable database).

La contraseña ingresada debe tener:

- Al menos 12 caracteres de longitud
- Contener al menos 1 carácter en mayúscula
- 1 carácter en minúscula
- 1 dígito [0-9].

Oracle Database 18c Express Edition

Oracle Database Information
Specify the database password.

18^c ORACLE[®]
Database
Express Edition

This password will be used for SYS, SYSTEM and PDBADMIN accounts.

Enter Database Password

Confirm Database Password

InstallShield

< Back Next > Cancel

**Agregando
contraseña a la
base de datos**

Oracle Database Installed Successfully.

The InstallShield Wizard has successfully installed Oracle Database 18c Express Edition. Click Finish to exit the wizard.

Oracle Database Express Edition Connection Information:

Multitenant container database: localhost:1521

Pluggable database: localhost:1521/XEPDB1

EM Express URL: <https://localhost:5500/em>

< Back

Finish

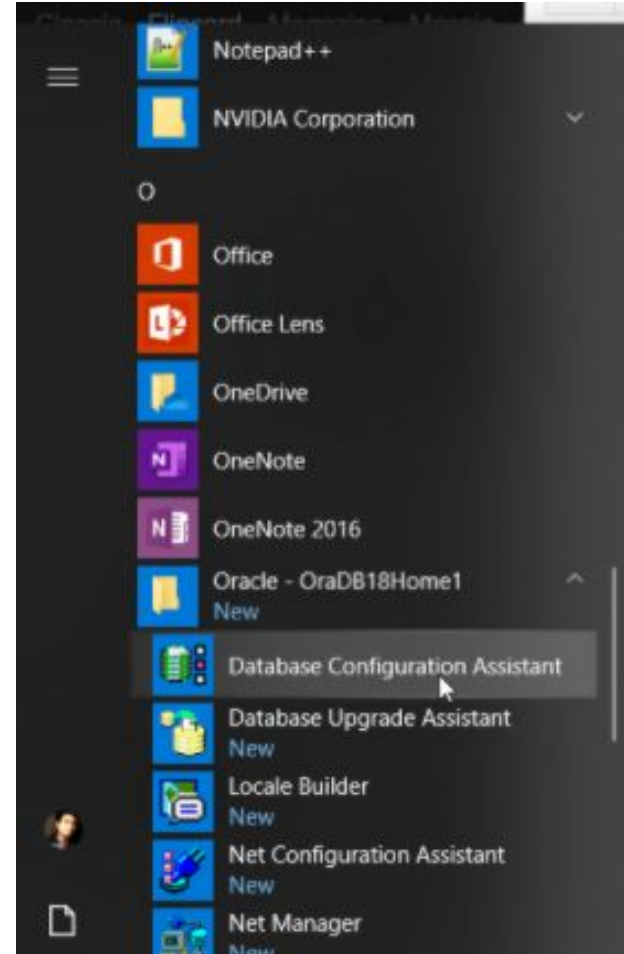
Cancel

18^c ORACLE[®]
Database
Express Edition

Resultado de
la instalación

Creación de base de datos en Oracle

- Se utiliza un asistente de creación de base de datos conectable.
- El asistente fue instalado en la máquina junto a la instalación anterior.



Oracle Database Express Edition (En Español) - Step 1 of 14

Select Database Operation

18c ORACLE Database

Select the operation that you want to perform.

- ☒ Create a database
- ☐ Configure an existing database
- ☐ Delete database
- ☐ Manage templates
- ☐ Manage Pluggable databases
- ☐ Oracle RAC database instance management

Database Operation

Creation Mode

Deployment Type

Database Identification

Storage Option

Fast Recovery Option

Database Options

Configuration Options

Management Options

User Credentials

Creation Option

Summary

Progress Page

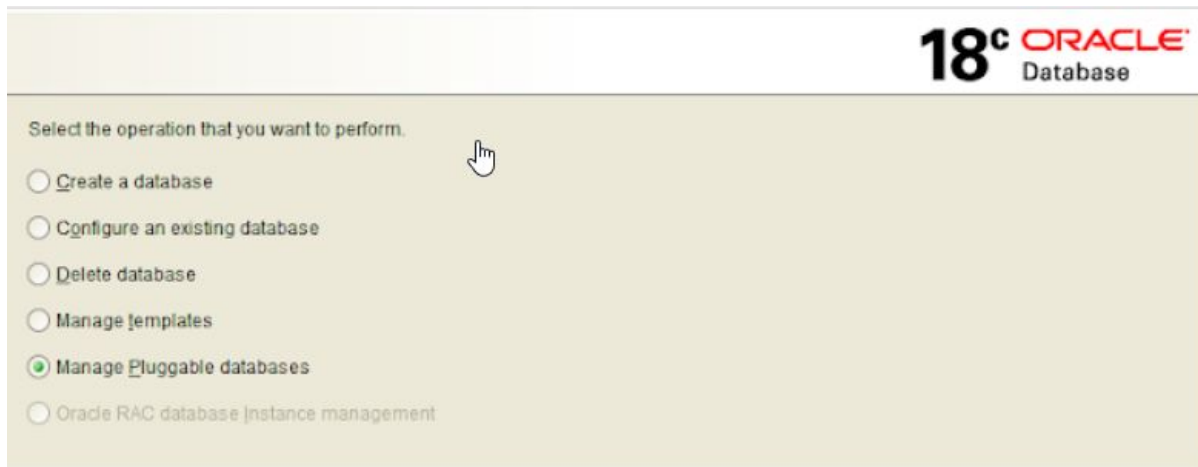
Finish

Help

< Back Next > Finish Cancel

Asistente de
creación

Seleccionar Manage Pluggable database.




18^c ORACLE[®]
Database

Select the operation that you want to perform.

- ☐ Create a database
- ☐ Configure an existing database
- ☐ Delete database
- ☐ Manage templates
- ☒ Manage Pluggable databases
- ☐ Oracle RAC database instance management

Seleccionar Create a Pluggable database.



es 18^c ORACLE[®]
Database

Select the operation that you want to perform in a Container database

- ☒ Create a Pluggable database
- ☐ Delete a Pluggable database
- ☐ Unplug a Pluggable database
- ☐ Configure a Pluggable database

Select a Container database within which Pluggable database needs to be created.

	Database	Local instance	Type
<input checked="" type="radio"/>	XE	XE	Single Instance

DBCA will connect to the database using OS based authentication. Database credentials may be needed if OS based authentication is disabled. Specify the credentials, if needed.

User name:

Password:

< Back

Next >

Finish

Cancel

@
Seleccionando
la base de datos
XE

Seleccionar PSB\$SEED.

18^c ORACLE[®]
Database

☒ Create a new Pluggable database from another PDB

Select Pluggable database:

☐ Create Pluggable database from an unplugged PDB

☐ Create as clone

☐ Create from PDB archive

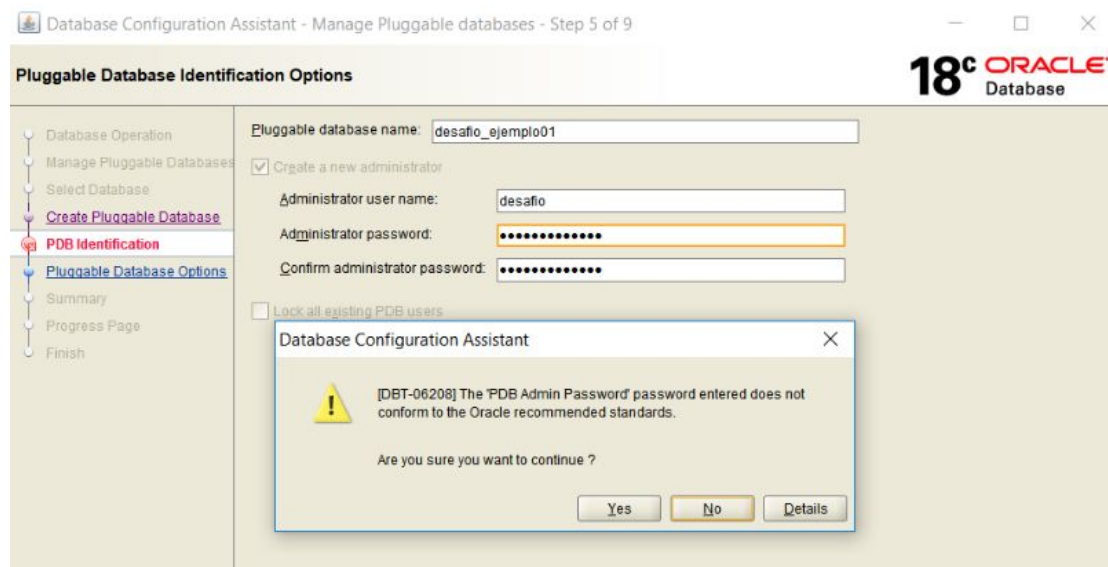
Pluggable database archive:

☒ Create using PDB file set

Pluggable database metadata file:

Pluggable database datafile backup:

- Nombre de base de datos: desafio_ejemplo01
- UserName administrador: desafio
- Password: desafio_latam



Indica que no cumple con los requisitos.

Agregar nombre a la base de datos.

PDB storage options

Specify the location for PDB datafiles.

Storage type: File System

Database location: C:\APP\ROBERTO\PRODUCT\118.0.0\ORADATA\XE\{PDB_NAME}

Browse...

☒ Create default user & tablespace

Progress

53%

Completing Pluggable Database Creation : In Progress

Status

➔ PDB Creation	In Progress
✓ • Prepare for db operation	Succeeded
✓ • Creating Pluggable Database	Succeeded
➔ • Completing Pluggable Database Creation	In Progress
• Executing Post Configuration Actions	Pending

Details

Revert All

Revert

Retry

Skip

DBCA Log Location:

C:\app\ROBERTO\product\18.0.0\cfgtoollogs\dbca\XE\EPDB2\trace_log_2019-02-23_09-38-55AM

Database Alert Log Location:

C:\APP\ROBERTO\PRODUCT\18.0.0\diag\rdbms\xe\XE\trace>alert_XE.log

Creando la
base de datos

Instalación en Linux (CentOS, Oracle Linux)

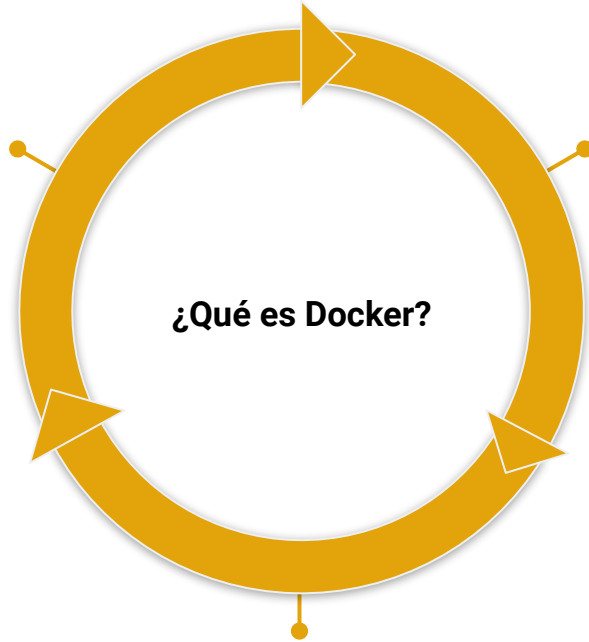
- **CentOS 8:**

- <https://tipstecnologicos.es/linux-instalacion-oracle-18c-xe-en-centos-8-mediante-conso-la/>

- **Oracle Linux:**

- <http://oracleparatodos1.blogspot.com/2018/05/instala-oracle-18c-sobre-linux-6-y-crea.html>
- <https://oraxedatabase.blogspot.com/2018/10/como-instalar-oracle-database-18c.html>
- <https://apasoft-training.es/blog/instalar-oracle-18c-en-linux>

Permiten
empaquetar una
aplicación con
todas las partes
que necesita



Contenedor virtual.

Facilita la creación,
implementación y
ejecución de
aplicaciones

{desafío}
latam_

Instalación
en Mac
(Docker, VM)

DBEAVER

Descripción de la herramienta:

Administración
de base de
datos libre.

Muchas
funcionalidades
y un muy buen
funcionamiento.

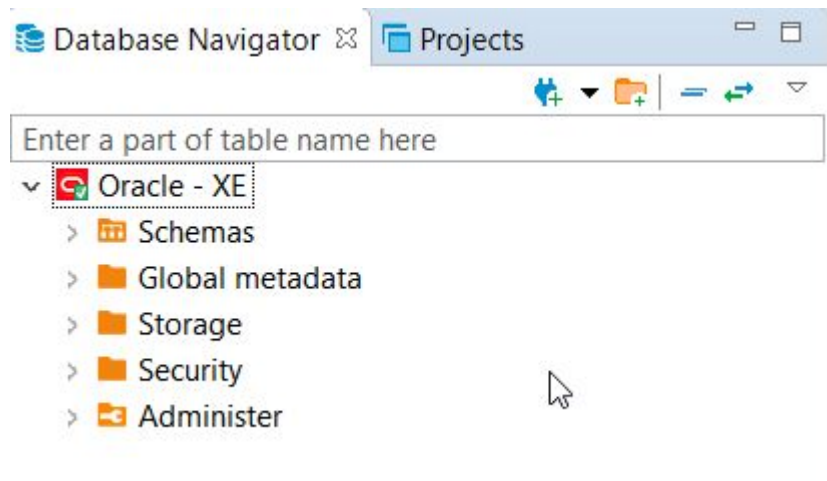
Puede utilizarse
con Oracle,
mysql, postgres,
bases de datos
no relacionales,
etc.

Facilidad de uso
y su bajo
consumo de
recursos.

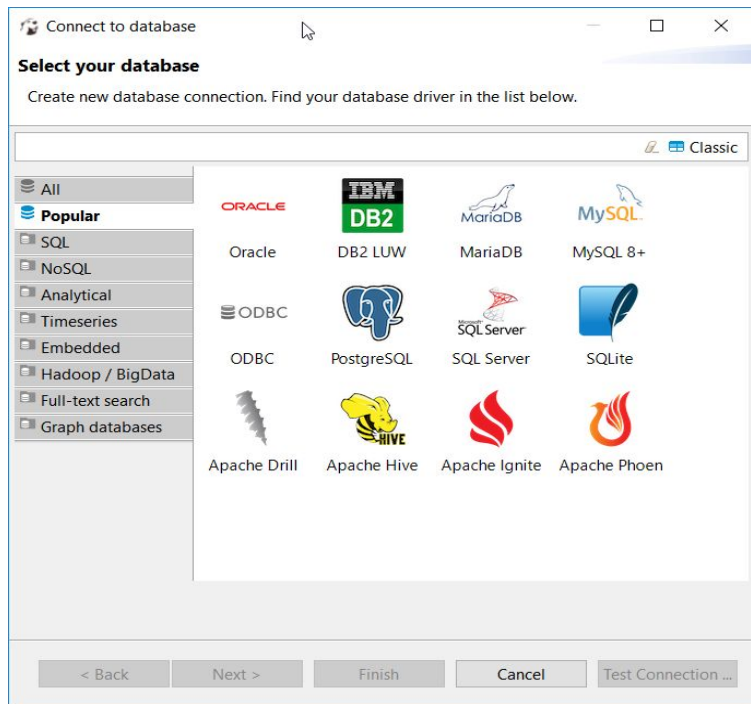
Conociendo el IDE para conectar a la Base de datos

Similar a eclipse y es muy intuitiva.

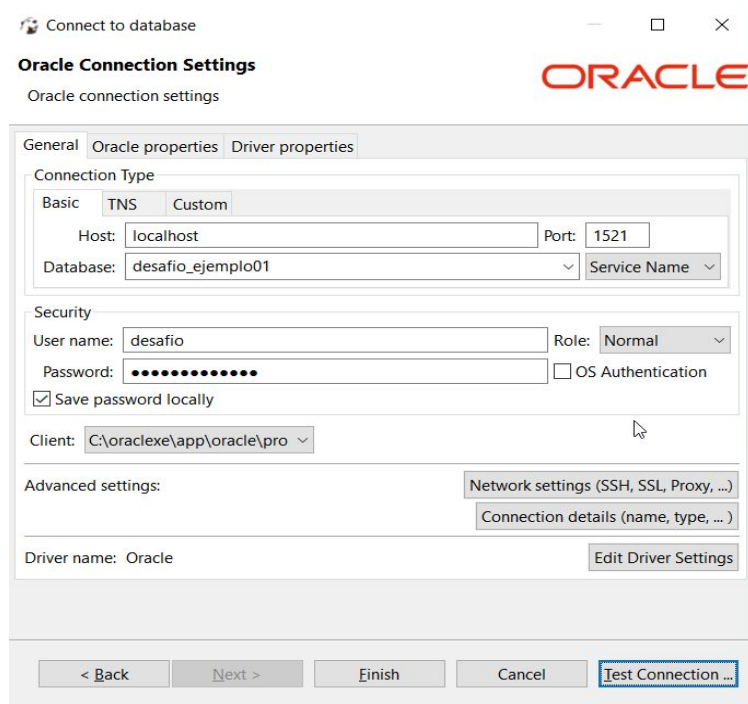
Interfaz.

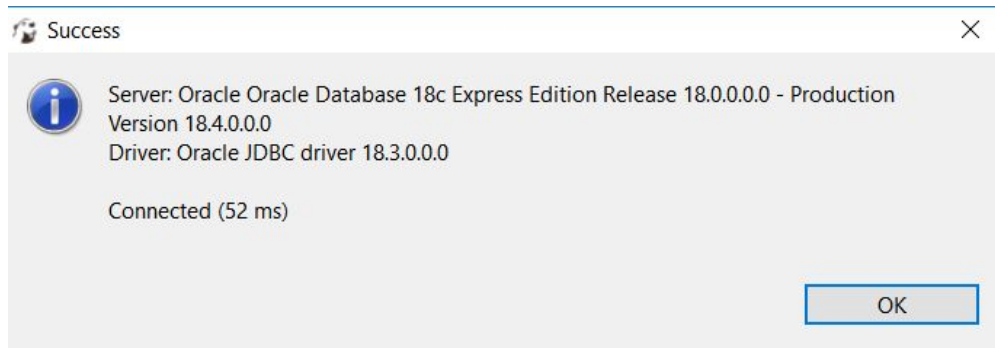


Seleccionar motor de base de datos a utilizar



Probando la conexión





**Resultado al
conectarse a la
base de datos**

SQL Developer

Descripción de la herramienta:





Descargar la opción Windows 32-bit/64-bit

SQL Developer 19.2 Downloads

Version 19.2.0.206.2117 August 1, 2019

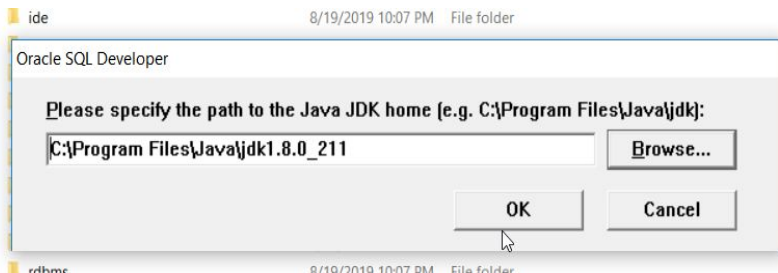
- [Release Notes](#)
- [Bugs Fixed](#)
- [Documentation](#)

Platform	Download	Notes
Windows 64-bit with JDK 8 included	 Download (490 MB)	<ul style="list-style-type: none">• MD5: aaaa291a468b73fc5c374f9213d7456d• SHA1: f250948419d3336998fb666f8ce5dcd6203431ab• Installation Notes
Windows 32-bit/64-bit	 Download (410 MB)	<ul style="list-style-type: none">• MD5: b6633f8488d6ccece6a345d59a984d63• SHA1: 086c800487af8dcb2599036a3b56b76f8fc0394d• Installation Notes• JDK 8 or 11 required

Activate Windows
Go to Settings to activate Windows.



Seleccionar la ruta del JDK



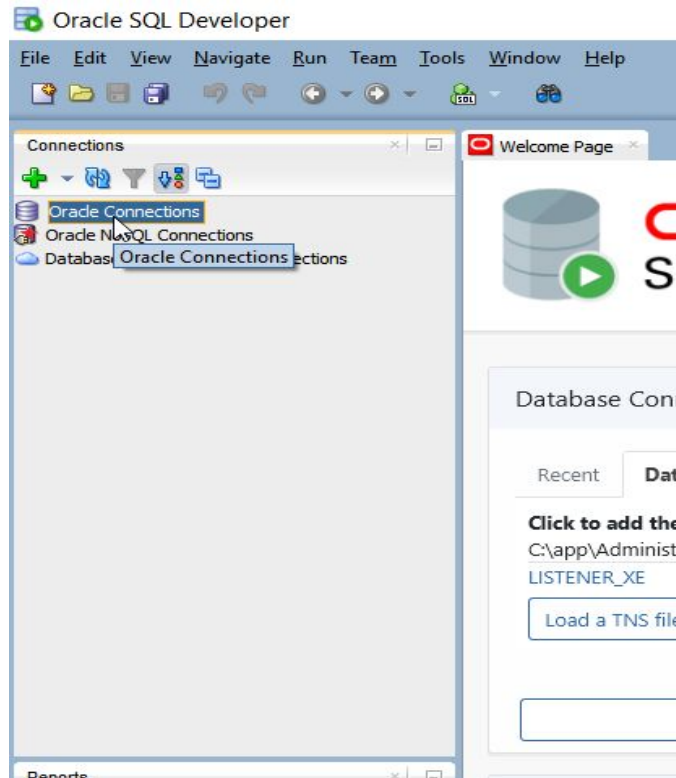
Instalando el entorno de trabajo



Conociendo el IDE para conectar a la Base de datos

Conectar el sql developer con la base de datos correspondiente.

Creando una conexión.



Agregando las credenciales.

New / Select Database Connection

Connection Name	Connection Details
desafio_ejemplo01	sys@//localhost:...

Name: desafio_ejemplo01

Database Type: Oracle

User Info

Authentication Type: Default

Username: sys

Password:

Role: SYSDBA

Save Password: ☒

Connection Type: Basic

Details

Hostname: localhost

Port: 1521

☐ SID

☒ Service name: desafio_ejemplo01

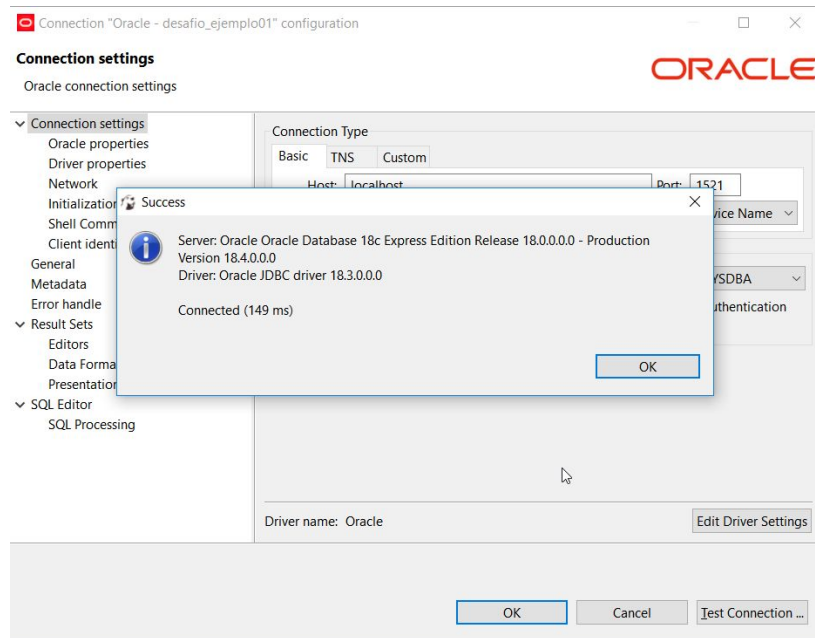
Status :

Help Save Clear Test Connect Cancel

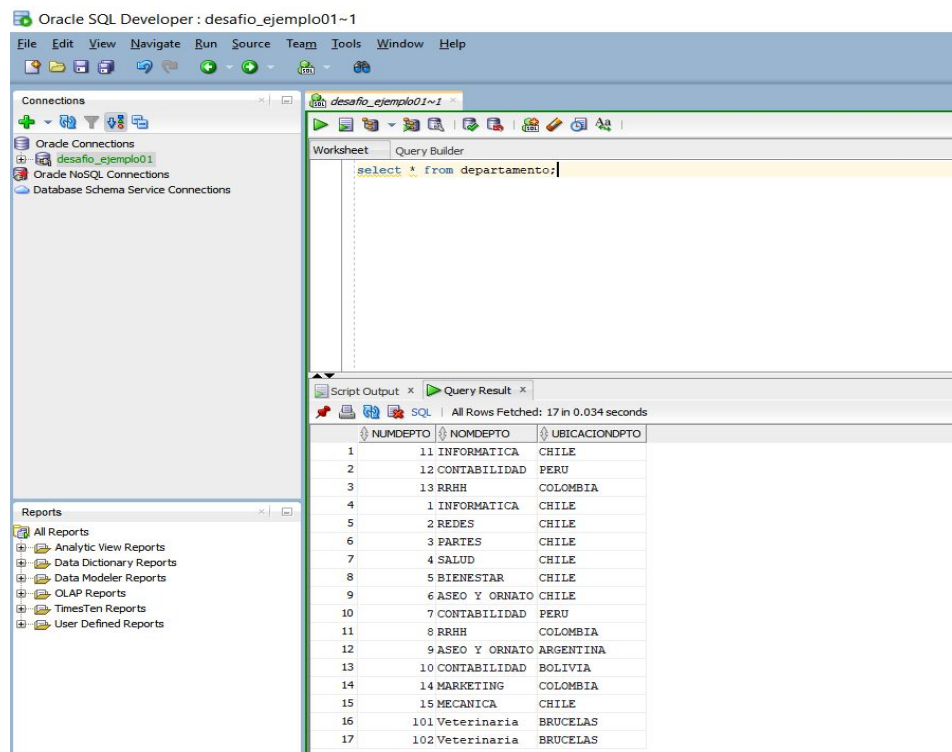
Los datos de conexión son:

Name	A gusto personal, puede ser cualquier nombre
Username	sys
Password	admin
Role	SYSDBA
Hostname	localhost
Port	1521
Service name	desafio_ejemplo01 (es el nombre de la base de datos)

Probando la conexión



Realizando una búsqueda



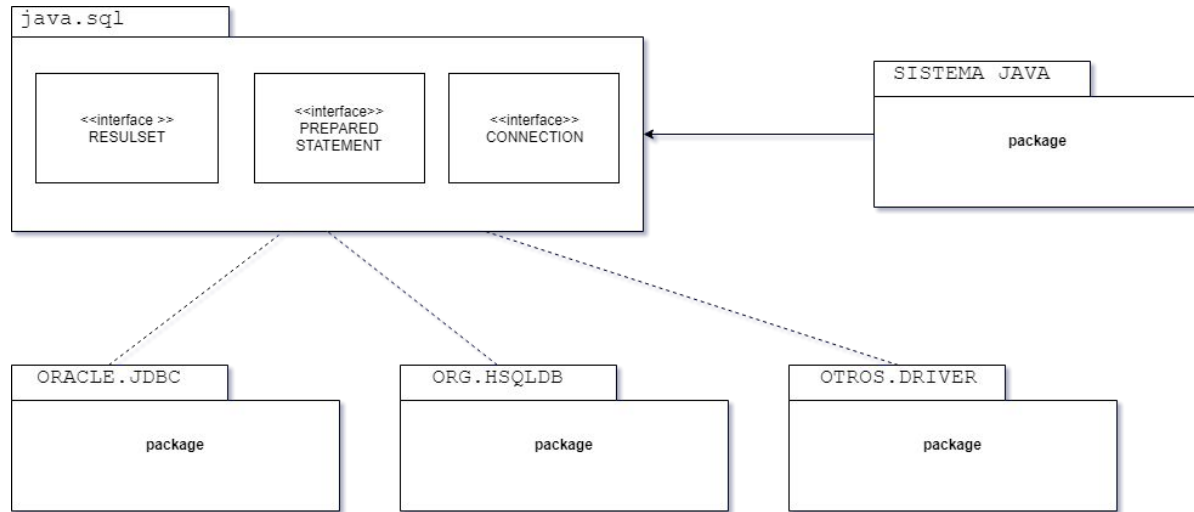
Conectar programas java con base de datos con JDBC

- Definen cómo se ejecuta la conexión.
- La forma de ejecutar las sentencias sql.
- Es necesario tener una clases que implementen todas estas interfaces, se le llama drivers.

Driver de base de datos

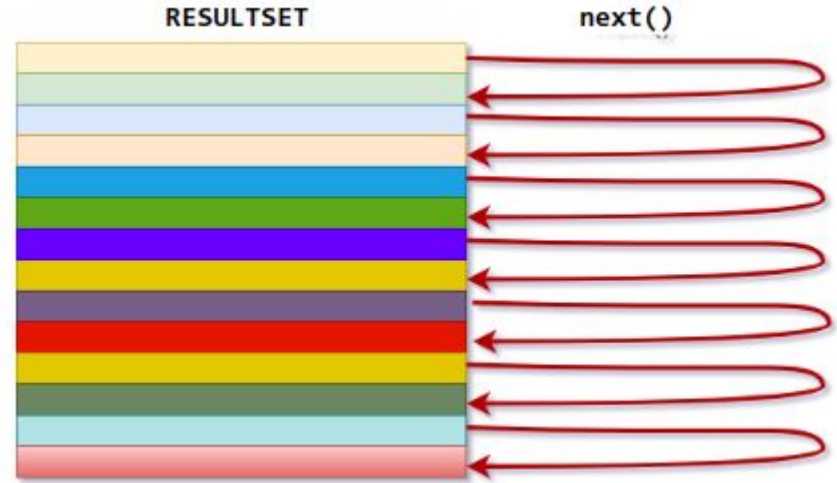
Intermediario entre el programa java y el motor de base de datos en cuestión.

Diagrama JDBC.



Componentes del api JDBC

- **ResultSet:**
 - Proporciona los métodos necesarios para poder obtener una representación de los datos de una tabla, previa consulta sql.



- Un objeto resultset no es actualizable y el cursor solo se mueve hacia adelante.
- Con un resultset podemos recorrer todo un set de datos obtenidos de una tabla gracias a una serie de métodos que proveen distintas funcionalidades.

Ejemplo:

Si la query retorna un valor de tipo String

```
resultset.getString("nombre");
```

Documentación para ResultSet.

int	<code>getConcurrency()</code> Retrieves the concurrency mode of this ResultSet object.
String	<code>getCursorName()</code> Retrieves the name of the SQL cursor used by this ResultSet object.
Date	<code>getDate(int columnIndex)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Date object in the Java programming language.
Date	<code>getDate(int columnIndex, Calendar cal)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Date object in the Java programming language.
Date	<code>getDate(String columnLabel)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Date object in the Java programming language.
Date	<code>getDate(String columnLabel, Calendar cal)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a java.sql.Date object in the Java programming language.
double	<code>getDouble(int columnIndex)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a double in the Java programming language.
double	<code>getDouble(String columnLabel)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a double in the Java programming language.
int	<code>getFetchDirection()</code> Retrieves the fetch direction for this ResultSet object.
int	<code>getFetchSize()</code> Retrieves the fetch size for this ResultSet object.
float	<code>getFloat(int columnIndex)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a float in the Java programming language.
float	<code>getFloat(String columnLabel)</code> Retrieves the value of the designated column in the current row of this ResultSet object as a float in the Java programming language.
int	<code>getHoldability()</code> Retrieves the holdability of this ResultSet object.
int	<code>getInt(int columnIndex)</code> Retrieves the value of the designated column in the current row of this ResultSet object as an int in the Java programming language.

<https://docs.oracle.com/javase/8/docs/api/java/sql/ResultSet.html>

PreparedStatement

Interface que representa una sentencia sql precompilada.

```
PreparedStatement pstmt = con.prepareStatement("UPDATE EMPLOYEES  
                                           SET SALARY = ? WHERE ID = ?");  
pstmt.setBigDecimal(1, 153833.00)  
pstmt.setInt(2, 110592)
```

Connection

Una conexión (sesión) con una base de datos específica.

Las instrucciones SQL se ejecutan y los resultados se devuelven dentro del contexto de una conexión.

La base de datos de un objeto puede proporcionar información que describe sus tablas, su gramática SQL admitida, etc.

Drivers de conexión con Oracle

Oracle JDBC Thin Driver
Implementación para la mayoría de requerimientos de conexión.
compuesto en su totalidad de código java y es conocido como un driver tipo IV.
Soporta desde la versión 5 de java y es independiente de la plataforma.
Proporciona conexión directa a la base de datos.
Soporta el protocolo TCP/IP y puede ser ejecutado en cualquier máquina que tenga instalada la java virtual machine.

Oracle JDBC OCI Driver
Conocido como driver de tipo II.
Necesita de la instalación de un cliente oracle en la máquina en la cual se implemente.
Se utiliza para implementaciones empresariales.

Descarga del driver Oracle JDBC

1. Para el caso de la tecnología Oracle debe dirigirse a la página oficial de descarga:
<https://www.oracle.com/technetwork/database/features/jdbc/jdbc-ucp-122-3110062.html>
2. Descargar el driver ojdbc8.jar.
3. Guardar en una ubicación conocida porque se ocupará más adelante.

/* Manipular información de una base de datos */

Creación del modelo de datos

- El negocio indica que quiere registrar los departamentos de la empresa las cuales están repartidas en el mundo y a la vez mantener a los empleados y su departamento.

Creamos dos tablas:

```
CREATE TABLE DEPARTAMENTO (NUMDEPTO NUMBER  
PRIMARY KEY NOT NULL, NOMDEPTO VARCHAR(100),  
UBICACIONDPTO VARCHAR(100));
```

```
CREATE TABLE EMPLEADO (NUMEMPLEADO NUMBER PRIMARY  
KEY, NOMBRE VARCHAR(100), NUMDEPTO INTEGER);
```

```
ALTER TABLE EMPLEADO  
ADD CONSTRAINT DEP_EMPLEADO  
FOREIGN KEY (NUMDEPTO)  
REFERENCES DEPARTAMENTO (NUMDEPTO);
```

```
INSERT INTO DEPARTAMENTO VALUES(11,  
'INFORMATICA', 'CHILE');
```

```
INSERT INTO EMPLEADO VALUES(0101, 'Matias  
Zarate', 11);
```

Creando tablas departamento y empleado.

```
*<Oracle - XE> Script
--creacion de tablas para el ejemplo 01 jdbc desafio latam

CREATE TABLE DEPARTAMENTO (NUMDEPTO NUMBER PRIMARY KEY NOT NULL, NOMDEPTO VARCHAR(100), UBICACIONDPTO VARCHAR(100));

CREATE TABLE EMPLEADO (NUMEMPLEADO NUMBER PRIMARY KEY, NOMBRE VARCHAR(100), NUMDEPTO INTEGER);

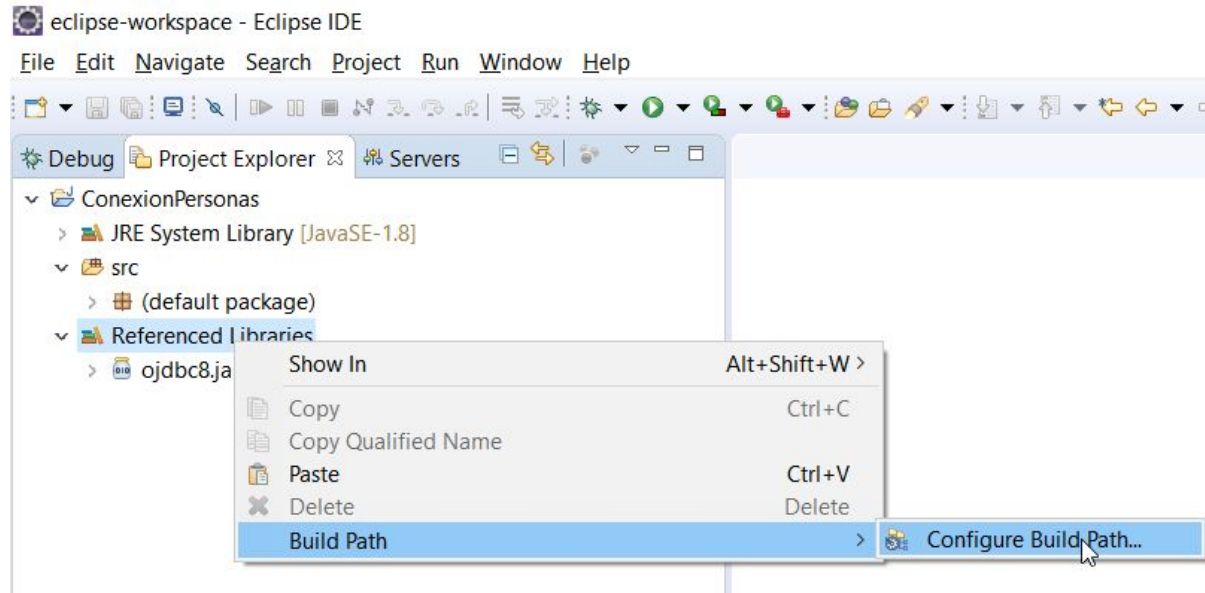
ALTER TABLE EMPLEADO
ADD CONSTRAINT DEP_EMPLEADO
FOREIGN KEY (NUMDEPTO)
REFERENCES DEPARTAMENTO (NUMDEPTO);

INSERT INTO DEPARTAMENTO VALUES(11,'INFORMATICA','CHILE');

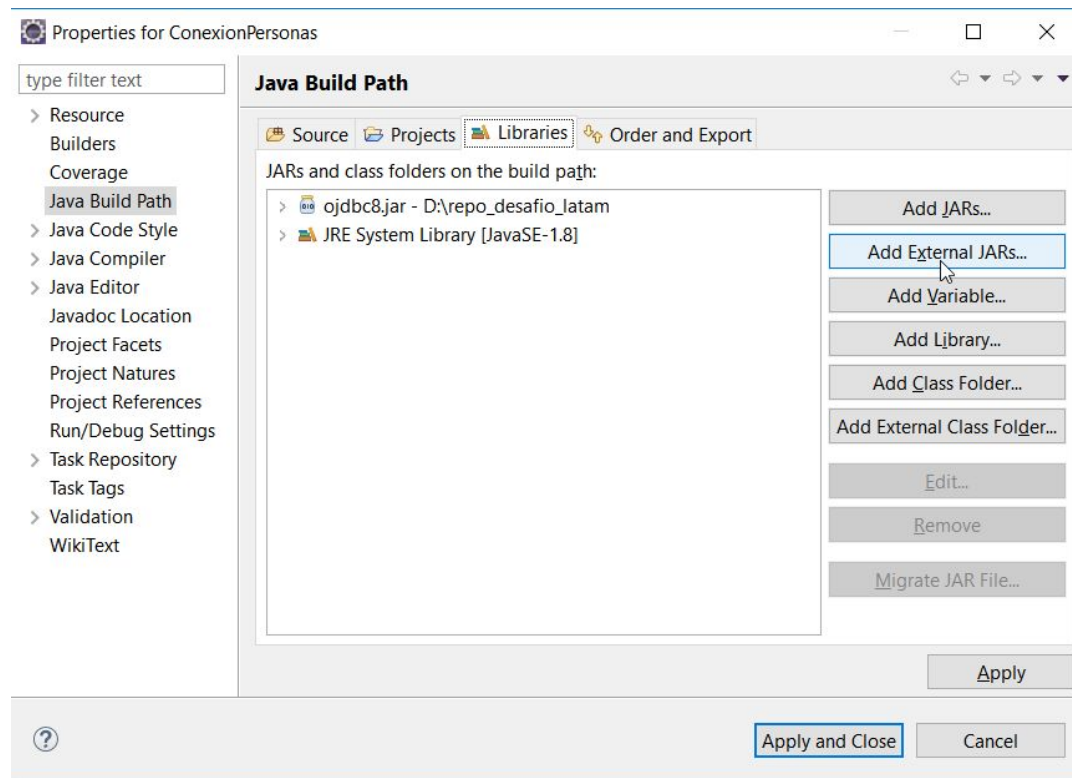
INSERT INTO EMPLEADO VALUES (0101,'Matias Zarate', 11);
```

Importar Driver JDBC a Eclipse

- Abrimos el Eclipse y generamos un nuevo proyecto java normal.



Agregando .jar



Clases de JDBC que nos permite conectar a las bases de datos

DriverManager	Administrar la carga del driver de conexión a base de datos.
Connection	Disponibiliza las clases utilizadas para generar y administrar la conexión a base de datos.
PreparedStatement	Ofrece clases para generar las sentencias con sus valores a utilizar.
ResultSet	Otorga los datos rescatados desde Oracle.

Analizando código entre líneas 7 a la 13

```
7 | String usr = "sys as sysdba";  
8   String pwd = "admin";  
9   String driver = "oracle.jdbc.driver.OracleDriver";  
10  String url = "jdbc:oracle:thin:@//localhost:1521/desafio_ejemplo01";  
11  Connection conn = null;  
12  PreparedStatement pstmt = null;  
13  ResultSet rs = null;
```



- **usr:** Corresponde al nombre de usuario del esquema de base de datos.
- **pwd:** Configuración de la password de oracle.
- **driver:** La ruta del driver que se está utilizando.
- **url:** String de conexión necesario para poder acceder a los recursos.

- Declaración de las instancias Connection, Prepared Statement y Resultset.

```
11      Connection conn = null;
12      PreparedStatement pstmt = null;
13      ResultSet rs = null;
```

- Analizando código entre las líneas 14 a la 22.
Iniciamos el driver de conexión mediante Class.forName(driver).

```
14      try {
15          //::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
16          // 1 PARTE LEVANTAR EL DRIVER
17          //::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
18          // levantamos el driver
19          Class.forName(driver);
20          // establecemos la conexión
21          conn = DriverManager.getConnection(url,usr,pwd);
```

- Líneas 23 en adelante.
Es posible generar la sentencia sql que queramos.
- Línea 30.
Recibir los datos que retorna la consulta.

```
22 //::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
23 // 2 PARTE GENERACION DE QUERY
24 //::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
25 //definimos la query
26 String sql = "";
27 sql+="INSERT INTO DEPARTAMENTO (NUMDEPTO,NOMDEPTO,UBICACIONDPTO)";
28 sql+="VALUES(?,?,?)";
29 //preparamos sentencias que ejecutaremos
30 pstmt = conn.prepareStatement(sql);
31 //adjuntamos los valores a los parametros
32 pstmt.setInt(1,12);
33 pstmt.setString(2, "CONTABILIDAD");
34 pstmt.setString(3, "MEXICO");
35 int resultado = pstmt.executeUpdate();
36 //si la variable resultado es mayor a 0 el insert fue correcto
37 if(resultado > 0) {
38     System.out.println("Fila correctamente insertada !");
39 }else {
40     System.out.println("Ocurrio un error insertando el departamento");
41 }
```

- Resultado en la consola.

```
Console Problems Debug Shell
<terminated> DemoPersonas [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (Jul 17, 2019, 6:42:03 PM)
DEPARTAMENTOS:::
11
INFORMATICA
CHILE
EMPLEADOS:::
101
Matias Zarate
11
```

- Bloque finally.

```
44         }finally {
45             //cerramos todos los recursos en orden inverso al que fueron declarados
46             try {
47                 if(rs != null) rs.close();
48                 if(pstm!=null) pstm.close();
49                 if(conn!=null) conn.close();
50             } catch (Exception e) {
51                 e.printStackTrace();
52             }
53         }
```

Sentencias DML

Ejecutar un insert.

Sentencias de modificación de datos.

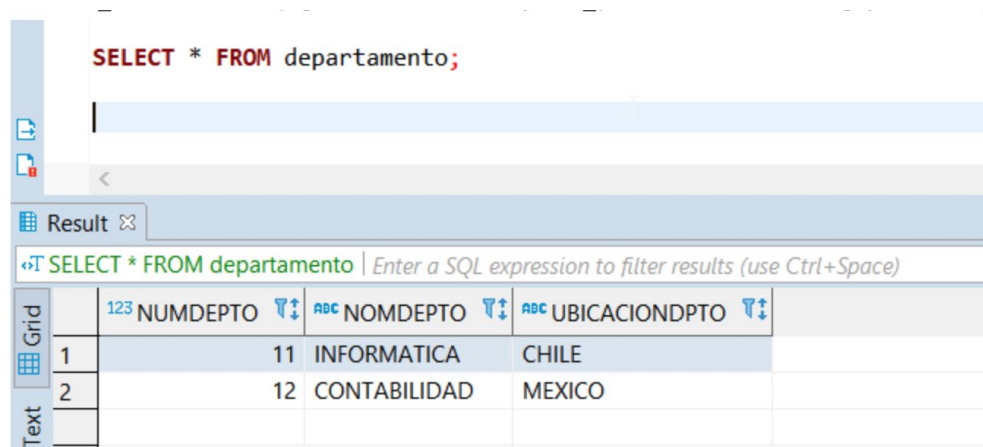
Procedimiento para generar un insert a la tabla de departamentos.

```
22 //::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
23 // 2 PARTE GENERACION DE QUERY
24 //::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
25 //definimos la query
26 String sql = "";
27 sql+="INSERT INTO DEPARTAMENTO (NUMDEPTO,NOMDEPTO,UBICACIONDPTO)";
28 sql+="VALUES(?,?,?)";
29 //preparamos sentencias que ejecutaremos
30 pstmt = conn.prepareStatement(sql);
31 //adjuntamos los valores a los parametros
32 pstmt.setInt(1,12);
33 pstmt.setString(2, "CONTABILIDAD");
34 pstmt.setString(3, "MEXICO");
35 int resultado = pstmt.executeUpdate();
36 //si la variable resultado es mayor a 0 el insert fue correcto
37 if(resultado > 0) {
38     System.out.println("Fila correctamente insertada !");
39 }else {
40     System.out.println("Ocurrio un error insertando el departamento");
41 }
```

Línea 28.

Cuenta con la sentencia VALUES y una serie de simbolos de interrogacion.

Nuevos registros.



The screenshot shows a SQL IDE interface. At the top, a query editor contains the text `SELECT * FROM departamento;`. Below the editor, a tab labeled "Result" is active, displaying the results of the query in a table format. The table has three columns: "NUMDEPTO", "NOMDEPTO", and "UBICACIONDPTO". The results show two rows of data.

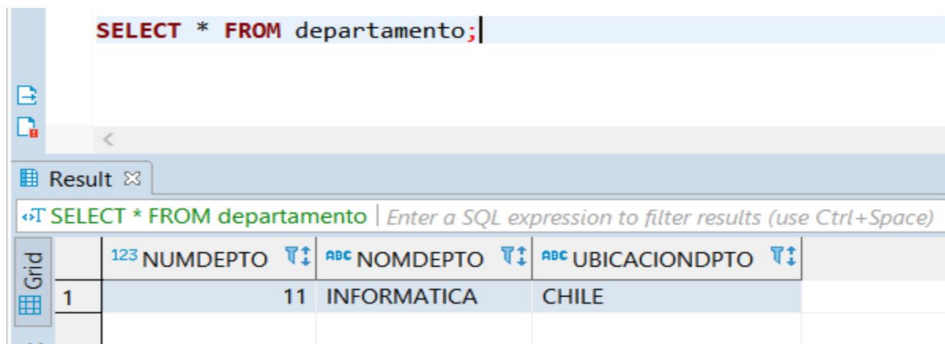
	NUMDEPTO	NOMDEPTO	UBICACIONDPTO
1	11	INFORMATICA	CHILE
2	12	CONTABILIDAD	MEXICO

Ejecutar un delete.

Eliminaremos el mismo registro que se insertó en la clase anterior.

```
25 //definimos la query
26 String sql = "";
27 sql+="DELETE FROM DEPARTAMENTO WHERE NUMDEPTO = ?";
28 //preparamos sentencias que ejecutaremos
29 pstmt = conn.prepareStatement(sql);
30 //adjuntamos los valores a los parametros
31 pstmt.setInt(1,12);
32 int resultado = pstmt.executeUpdate();
33 //si la variable resultado es mayor a 0 el insert fue correcto
34 if(resultado == 1) {
35     System.out.println("Fila correctamente eliminada !");
36 }else {
37     System.out.println("Ocurrio un error eliminando el departamento");
38 }
```

Elemento eliminado.



The screenshot shows a Java IDE with a SQL query editor and a results window. The query is `SELECT * FROM departamento;`. The results window shows a table with the following data:

	NUMDEPTO	NOMDEPTO	UBICACIONDPTO
1	11	INFORMATICA	CHILE

Ejecutar un delete masivo en cascada

Hay ocasiones en que se necesitan eliminar datos de tablas que tienen relación entre sí.

Ejemplo:

El departamento de recursos humanos va a desaparecer y junto a él todos sus empleados

Añadir la sentencia ON DELETE CASCADE la declaración del foreign key:

```
ALTER TABLE EMPLEADO  
ADD CONSTRAINT DEP_EMPLEADO  
FOREIGN KEY (NUMDEPTO)  
REFERENCES DEPARTAMENTO (NUMDEPTO) ON DELETE CASCADE;
```

Generar la sentencia de delete en contra de la tabla principal EMPLEADO.

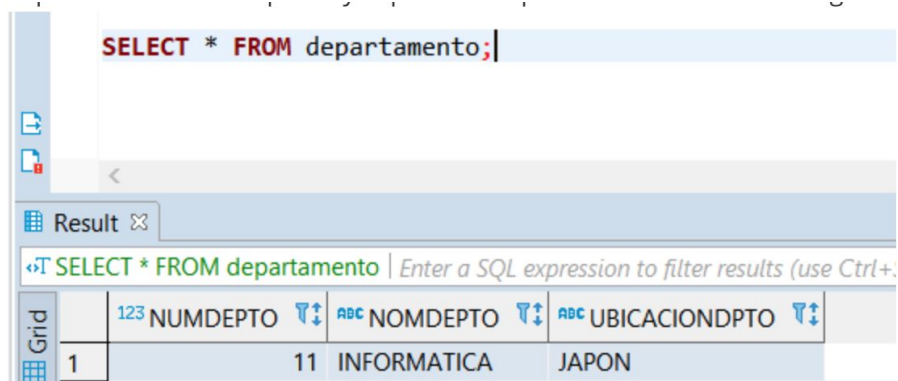
```
DELETE FROM EMPLEADO  
A WHERE A.NUMDEPTO = 11;
```

ON DELETE CASCADE

Ejecuta el borrado masivo dependiendo de la relación a la clave foránea especificada.

```
27 //definimos la query  
28 String sql = "";  
29 sql+="UPDATE DEPARTAMENTO SET UBICACIONDPTO = ? ";  
30 //preparamos sentencias que ejecutaremos  
31 pstmt = conn.prepareStatement(sql);  
32 //adjuntamos los valores a los parametros  
33 pstmt.setString(1,"JAPON");  
34 int resultado = pstmt.executeUpdate();  
35 System.out.println(resultado + "filas actualizadas");
```

Resultado de la ejecución del update.



The screenshot shows a database client window. At the top, a SQL query is entered in a text field: `SELECT * FROM departamento;`. Below the query field, there is a tab labeled "Result" with a close icon. Under the "Result" tab, the same query is displayed: `SELECT * FROM departamento`, followed by a hint: *Enter a SQL expression to filter results (use Ctrl+;)*. Below the query, a table grid displays the results. The grid has a "Grid" tab on the left. The table has four columns: "NUMDEPTO", "NOMDEPTO", "UBICACIONDPTO", and an empty column. The first row of data shows the values 11, INFORMATICA, and JAPON.

	123 NUMDEPTO	ABC NOMDEPTO	ABC UBICACIONDPTO	
1	11	INFORMATICA	JAPON	

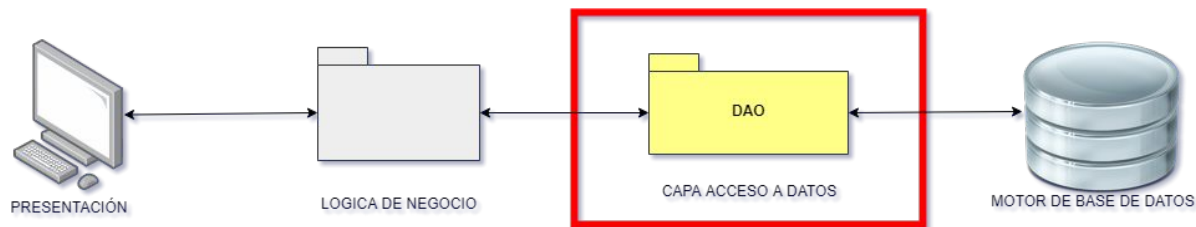
/* Patrón Data Access Object (DAO) */

Conocer el patrón DAO

Se encarga de abstraer y encapsular el acceso a las bases de datos.

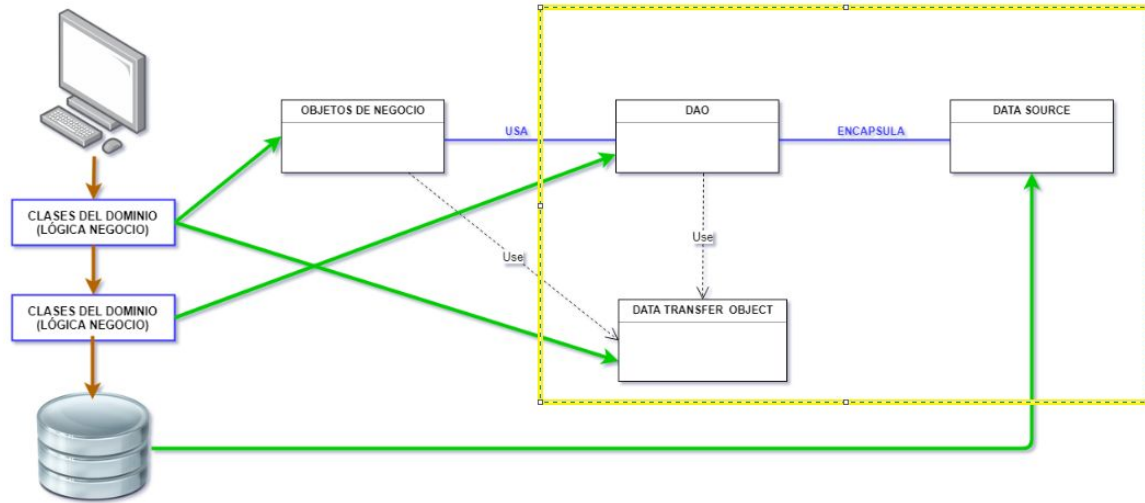
Administra la conexión y la forma en que se obtienen y manipulan los elementos.

Implementa el mecanismo requerido de acceso para trabajar con la fuente de datos.



- Sólo se tocará la capa DAO en caso de:
 - Modificaciones a las querys
 - Cambio de motor de BD
 - Cambio de driver de conexión
 - Cambios de implementación JDBC
 - Otros.

Estructura del patrón DAO



Objetos de negocio	Necesita acceder a la base de datos para obtener los datos de un empleado.
DAO	El DAO abstrae a los objetos de negocio la implementación de acceso a los datos.
Data Source	Representa la implementación de la conexión al motor de base de datos.
Data Transfer Object	Envía los datos a los objetos de negocio.

Analogía entre carretilla y DTO.



Implementación del patrón DAO en una aplicación web

JSP para
la vista

Un servlet
para procesar
las peticiones

Patrón DAO
para el
acceso a
datos

JDBC para
establecer la
conexión

Base de
datos de
ejemplo

Base de datos

Relación Departamento - Empleado.



Aplicación Web

- Para crear la estructura del proyecto, crear dentro de la carpeta src los siguientes packages.
 - com.desafiolatam.dao
 - com.desafiolatam.modelo
 - com.desafiolatam.procesaConexion
 - com.desafiolatam.servlet

Se crean las clases que representan las entidades del negocio.


```
1 package com.desafiolatam.modelo;
2
3 public class Empleado {
4     private int numEmpleado;
5     private String nombreEmpleado;
6     private int numDepto;
7
8     public Empleado(int numEmpleado, String nombreEmpleado, int numDepto) {
9         super();
10        this.numEmpleado = numEmpleado;
11        this.nombreEmpleado = nombreEmpleado;
12        this.numDepto = numDepto;
13    }
14
15    //getters y setters
```



Empleado

Departamento

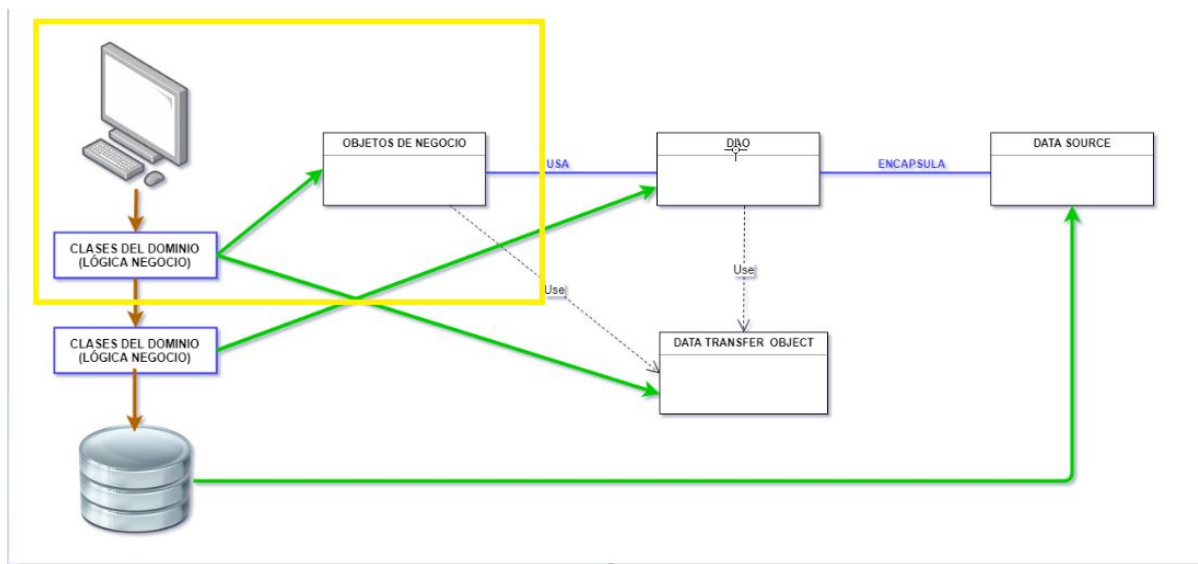
```
1 package com.desafiolatam.modelo;
2
3 public class Departamento {
4     private int numDepto;
5     private String nombreDepto;
6     private String ubicacionDepto;
7
8     public Departamento(int numDepto, String nombreDepto, String ubicacionDepto) {
9         super();
10        this.numDepto = numDepto;
11        this.nombreDepto = nombreDepto;
12        this.ubicacionDepto = ubicacionDepto;
13    }
14
15    //getters y setters
```

Paquete modelo

▼  com.desafiolatam.modelo

- >  Departamento.java
- >  Empleado.java

- El siguiente paso es crear las clases respectivas que representen a las tablas DEPARTAMENTO y EMPLEADO.
- Los modelos son las clases principales que se comunicaran con el patrón DAO.



- ¿Que es una Interfaz?
 - Clase abstracta donde puede estar compuesta por uno o más métodos especificados, pero no implementados.

Características de una interfaz
Herencia
Polimorfismo
Modularización del código.
Hacen más simple el mantenimiento del código
Facilitan el uso del código

Ejemplo de una Interfaz.

```
public interface Impresora{
    public abstract print();
    public abstract scan();
    public abstract copy();
}

public class Canon extends Impresora{ //clase donde se haria
la implementación de cada método }

public class Epson extends Impresora{ //clase donde se haria
la implementación de cada método }

public class Samsung extends Impresora{ //clase donde se
haría la implementación de cada método }
```


Métodos comunes en una implementación DAO	
create()	Crea un registro en la base de datos.
update(id)	Modifica un registro en la base de datos.
delete(id)	Elimina un registro en la base de datos.
getAll()	Trae todos los registros de una tabla de la base de datos.
getById(id)	Trae un registro específico de una tabla de la base de datos.

Objetos DAO

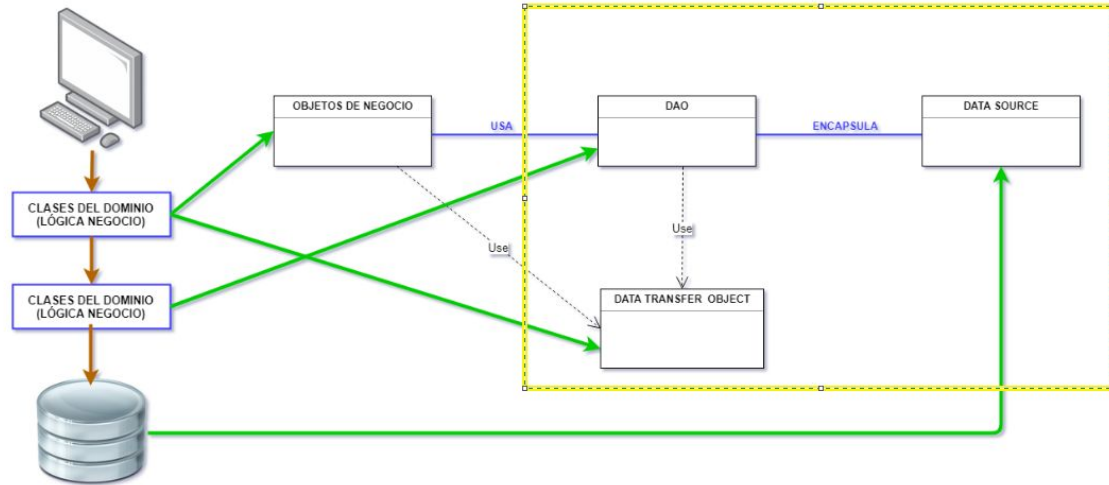
Creando Interfaz Departamento DAO

- ▼ com.desafiolatam.dao
 - > DepartamentoDao.java
 - > DepartamentoDaoImp.java
 - > EmpleadoDao.java
 - > EmpleadoDaoImp.java

```
1 package com.desafiolatam.dao;  
2  
3 import java.util.List;  
4  
5  
6  
7 public interface DepartamentoDao {  
8     public List<Departamento> obtieneDepartamento();  
9 }  
10
```

Diagrama incorporando DATASOURCE.

Se encarga de inicializar la conexión mediante JDBC.



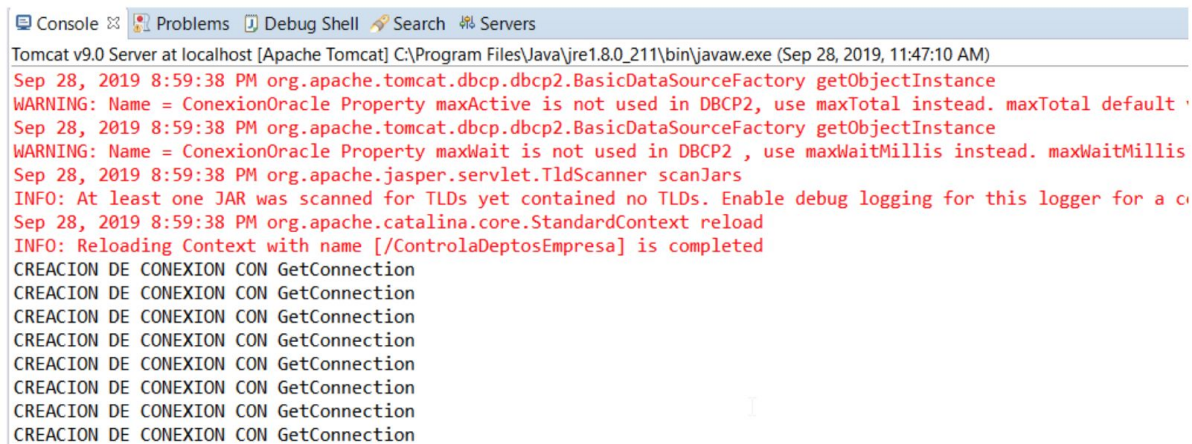
El patrón Singleton

Permite generar una y solo una instancia de alguna clase.

Creando una conexión.

```
20 protected Connection generaConexion() {
21     String usr = "sys as sysdba";
22     String pwd = "admin";
23     String driver = "oracle.jdbc.driver.OracleDriver";
24     String url = "jdbc:oracle:thin:@//localhost:1521/desafio_ejemplo01";
25
26     try {
27         Class.forName(driver);
28         conn = DriverManager.getConnection(url,usr,pwd);
29     } catch (Exception ex) {
30         ex.printStackTrace();
31     }
32     return conn;
33 }
```

Cada vez que se envíe un requerimiento que tenga que comunicarse con la base de datos, se abrirá una conexión.

A screenshot of a Java IDE's console window. The title bar shows tabs for 'Console', 'Problems', 'Debug Shell', 'Search', and 'Servers'. The console text shows the following logs:

```
Tomcat v9.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (Sep 28, 2019, 11:47:10 AM)
Sep 28, 2019 8:59:38 PM org.apache.tomcat.dbcp.dbcp2.BasicDataSourceFactory getObjectInstance
WARNING: Name = ConexioOracle Property maxActive is not used in DBCP2, use maxTotal instead. maxTotal default
Sep 28, 2019 8:59:38 PM org.apache.tomcat.dbcp.dbcp2.BasicDataSourceFactory getObjectInstance
WARNING: Name = ConexioOracle Property maxWait is not used in DBCP2 , use maxWaitMillis instead. maxWaitMillis
Sep 28, 2019 8:59:38 PM org.apache.jasper.servlet.TldScanner scanJars
INFO: At least one JAR was scanned for TLDs yet contained no TLDs. Enable debug logging for this logger for a c
Sep 28, 2019 8:59:38 PM org.apache.catalina.core.StandardContext reload
INFO: Reloading Context with name [/ControlaDeptosEmpresa] is completed
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
CREACION DE CONEXION CON GetConnection
```

Resultado a la ejecución del código.

¿Qué es el patrón Singleton?

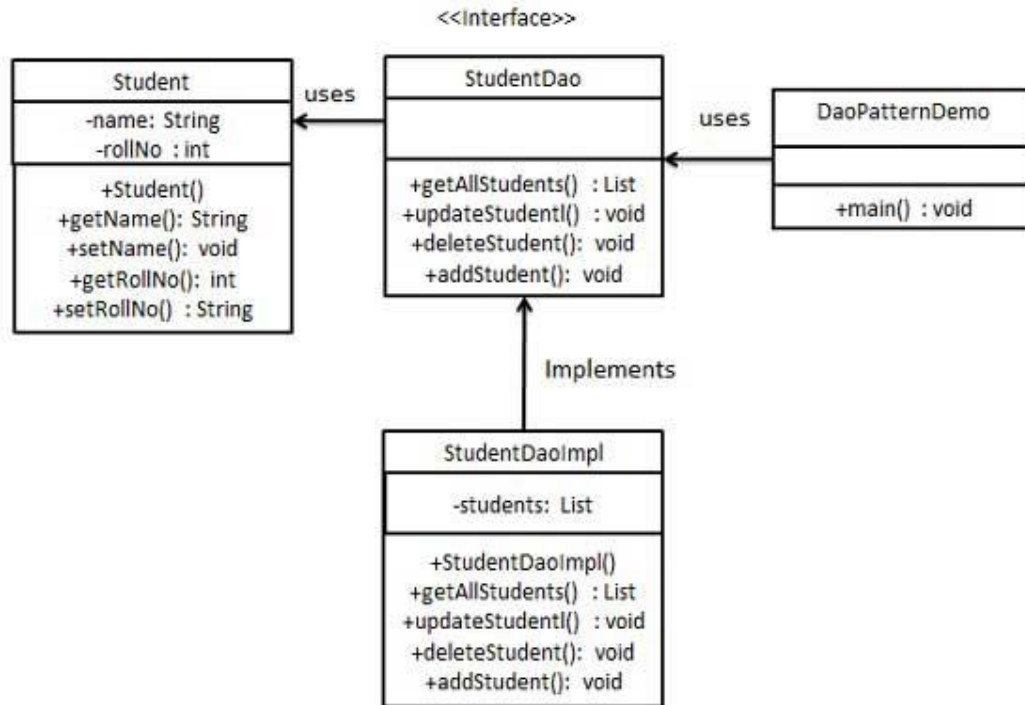
Impide que se generen nuevas instancias de clases si es que existe una.

Para este proyecto tenemos que:

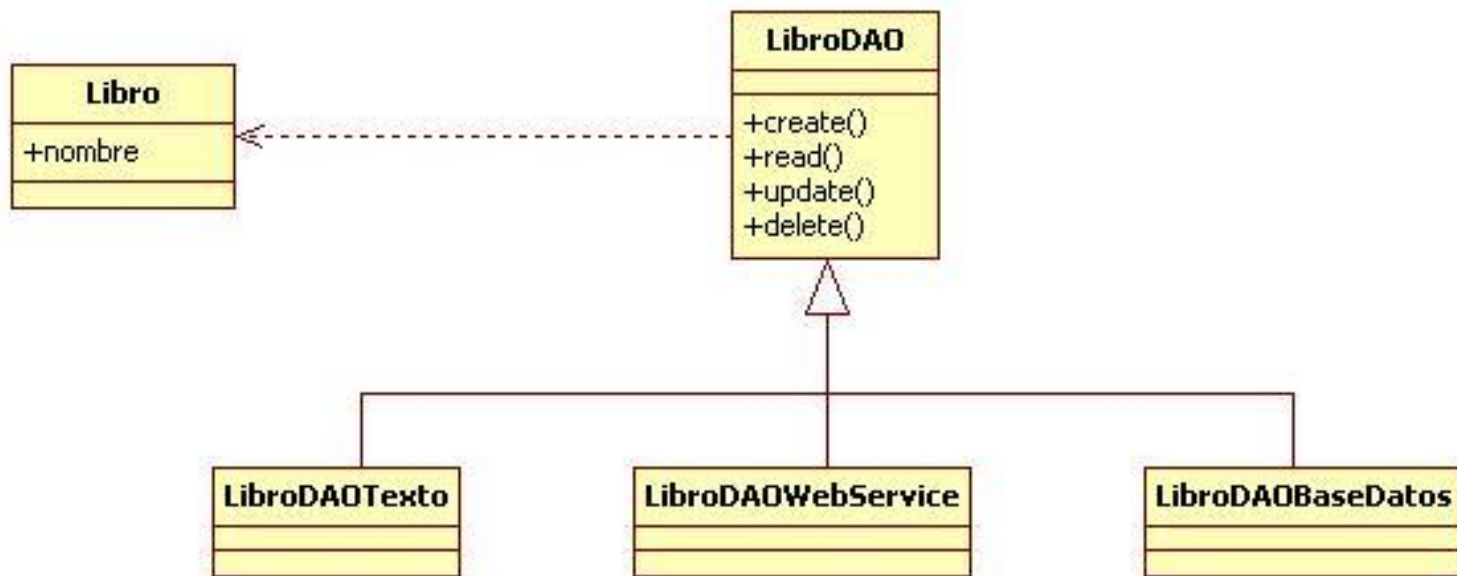
1. Declarar la instancia de Connection como static.
2. Verificar si el objeto de tipo Connection es nulo o no.

```
35 protected Connection generaPoolConexion() {
36     Context initContext;
37     if(conn == null) {
38         try {
39             initContext = new InitialContext();
40             DataSource ds = (DataSource) initContext.lookup("java:/comp/env/jdbc/ConexionOracle");
41             try {
42
43                 conn = ds.getConnection();
44
45                 System.out.println("CREACION DE CONEXION CON GetConnection");
46             } catch (SQLException e) {
47                 e.printStackTrace();
48             }
49         } catch (NamingException e) {
50             e.printStackTrace();
51         }
52         return conn;
53     } else {
54         return conn;
55     }
56 }
57
58 }
```

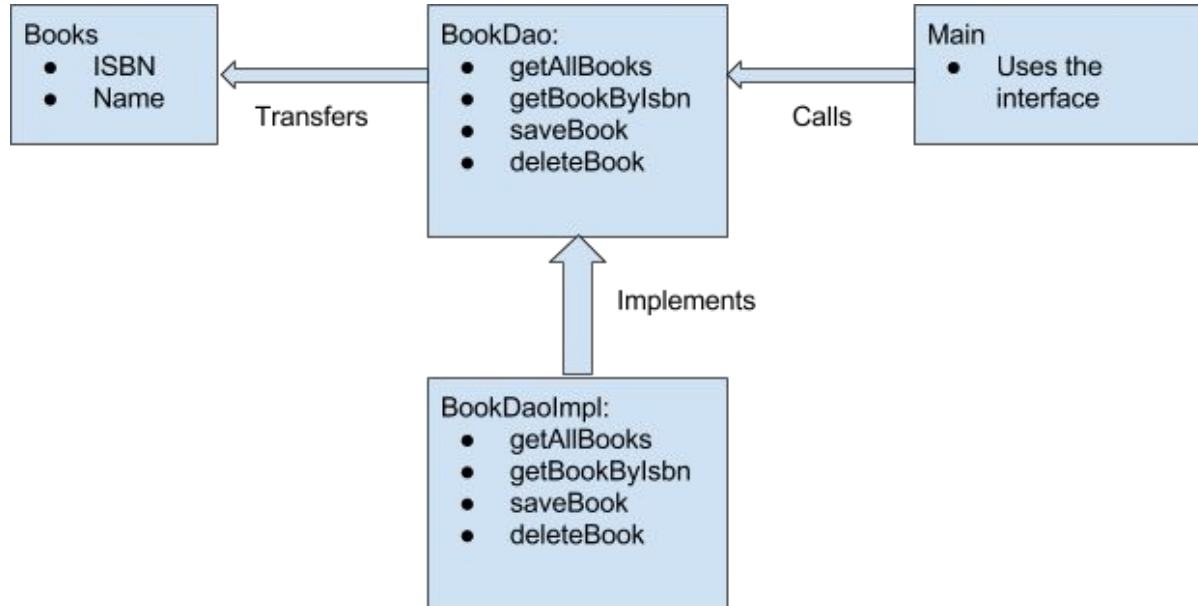
Ejemplos Patrón DAO



Ejemplo Patrón DAO 2



Ejemplo Patrón DAO 3



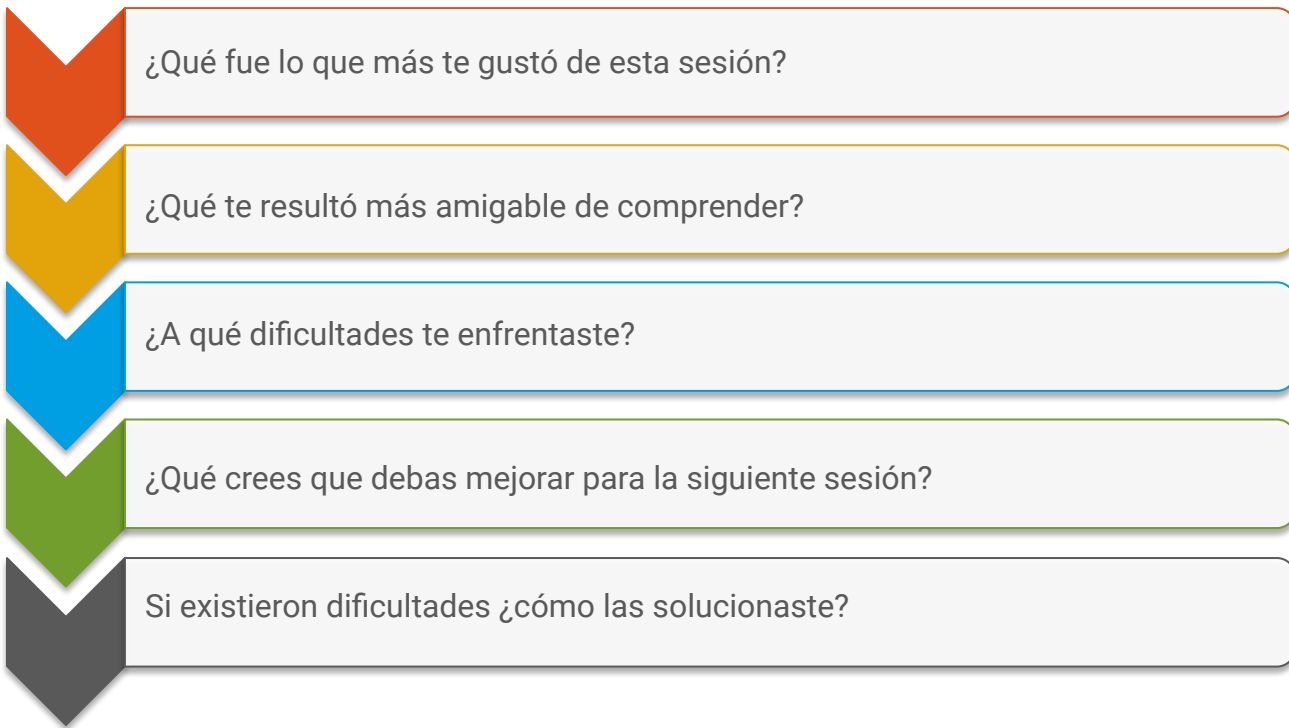


Cierre

{desafío}
latam_



15 minutos



¿Qué fue lo que más te gustó de esta sesión?

¿Qué te resultó más amigable de comprender?

¿A qué dificultades te enfrentaste?

¿Qué crees que debas mejorar para la siguiente sesión?

Si existieron dificultades ¿cómo las solucionaste?



*Academia de
talentos digitales*

www.desafiolatam.com



/DesafioLatam



/DesafioLatam



/DesafioLatam



/DesafioLatam