



TECNOLOGÍAS DE ACCESO A DATOS

4.1



NOVIEMBRE DE 2021
RODRÍGUEZ JÁCOME, DAVID
Desenvolvimento Web en Contorno Servidor

- Actividad 1: Hoja de cálculo.

En esta actividad se nos ha proporcionado la plantilla de una hoja de cálculo de Google Docs que debemos rellenar con datos de personas. Este ha sido el resultado:

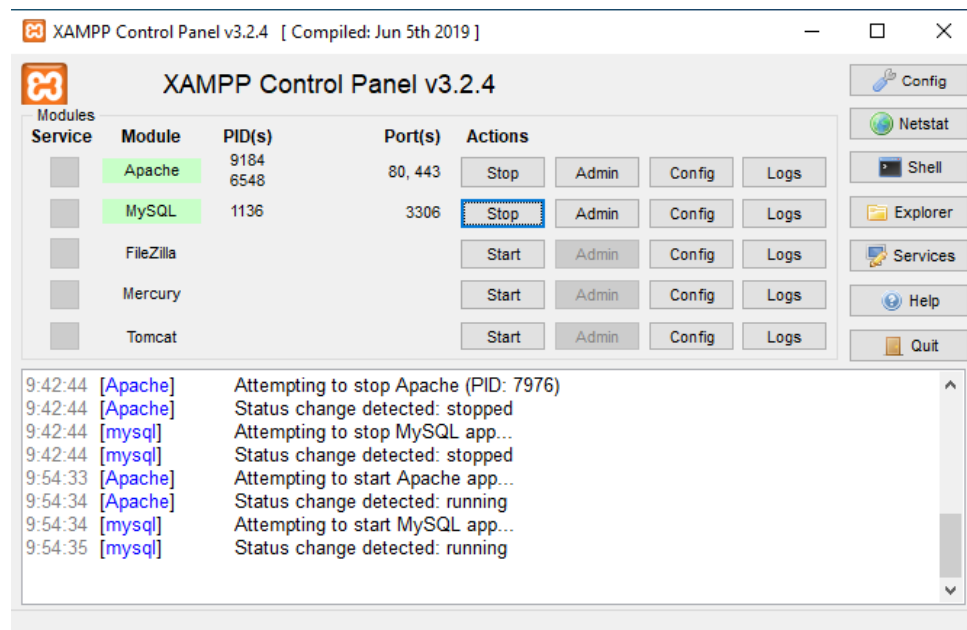
5	78956228W	Juan	Ríos	Pérez	Lugo	27678	659841772	34	David Rodríguez
6	78454312G	Laura	Vieitez	Novas	Ribeira	15960	665848412	21	David Rodríguez
7	74889653Y	Daniela	Pereira	Reiriz	Tui	36700	722841269	99	David Rodríguez

- Actividad 2: tecnologías de acceso a base de datos.

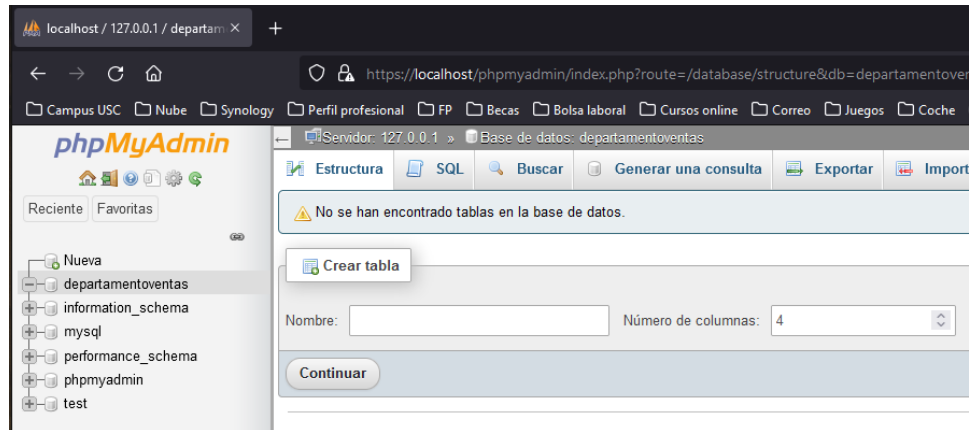
Escogeremos tres tecnologías para explicar diversos métodos de acceso a base de datos: PHPMyAdmin, MySQL WorkBench y JDBC (Java Database Connectivity).

- PHPMyAdmin:

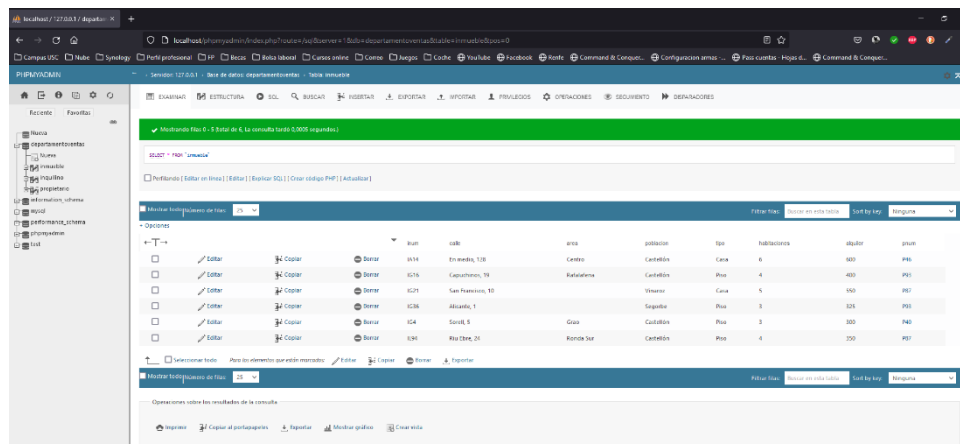
Con esta tecnología necesitaremos instalar en nuestro equipo XAMPP (hemos usado el SO Windows para ello), lo cual asumiremos que ya está instalado y configurado. Hecho esto, accedemos al panel de control de XAMPP y activaremos Apache y MySQL:



Seguidamente abriremos una ventana del navegador y escribiremos en la dirección URL “localhost/phpmyadmin” para acceder al gestor de bases de datos:



Una vez dentro, cargaremos un script que hemos creado para la actividad 3 para comprobar el acceso a la base de datos. Aquí vemos la base de datos cargada:



- MySQL Workbench.

Ahora tomando el ejemplo de Workbench y usando el mismo script de antes, iniciamos una sesión de trabajo en el servidor local de Workbench y copiamos el script en una query dentro de la misma sesión, y ejecutamos el script para ver el resultado:

The screenshot shows the MySQL Workbench interface. On the left, the 'Navigator' pane displays the 'departamentovenas' database structure, including tables like 'inmueble' and 'inquilino'. The 'Table: inmueble' is selected, showing its columns: 'inum' (PK, varchar(5)), 'calle' (varchar(50)), 'area' (varchar(30)), 'poblacion' (varchar(30)), 'tipo' (varchar(10)), 'habitaciones' (tinyint), 'alquiler' (int), and 'pnum' (varchar(5)).

The main editor shows a SQL script with the following queries:

```
1 • drop DATABASE if EXISTS departamentovenas;
2 • CREATE DATABASE IF NOT EXISTS departamentovenas
3     DEFAULT CHARACTER SET UTF8
4     DEFAULT COLLATE UTF8_SPANISH_CI;
5
6 • USE departamentovenas;
7
8 • CREATE TABLE if NOT EXISTS propietario (
9     pnum varchar(5) not null,
10    nombre varchar(30) not null,
11    apellido varchar(50) not null,
12    direccion varchar(50) not null,
13    telefono int(9) NOT NULL
14 ) engine INNODB;
15
16 • create table if not EXISTS inmueble (
17     inum varchar(5) not null,
```

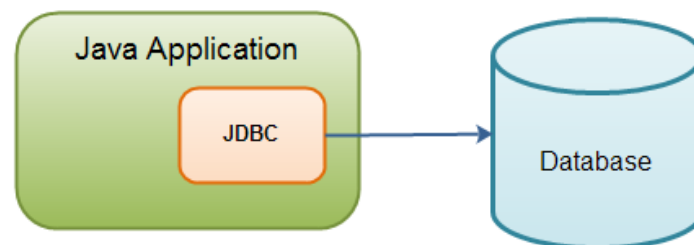
The 'Output' pane at the bottom shows the execution results of the script, listing 24 actions (inserts) and their timestamps. The actions are grouped by table: 'inmueble' (lines 19-21) and 'inquilino' (lines 22-24).

#	Time	Action
19	19:27:16	insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IG21", "San Francisco, 10", "", "Vinaroz", "Casa", 5, 550, "P87")
20	19:27:16	insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IG16", "Capuchinos, 19", "Rafalafena", "Castellón", "Piso", 4, 400, "P93")
21	19:27:16	insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q76", "Juan", "Felp", "Barceló 47, Castellón", 964282540, "Piso", 375)
22	19:27:16	insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q56", "Ana", "Grangel", "San Rafael 45, Almazora", 964551110, "Piso", 300)
23	19:27:16	insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q74", "Elena", "Abaso", "Navarra 76, Castellón", 964205560, "Casa", 700)
24	19:27:16	insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q62", "Alicia", "Mon", "Alloza 45, Castellón", 964229580, "Piso", 550)

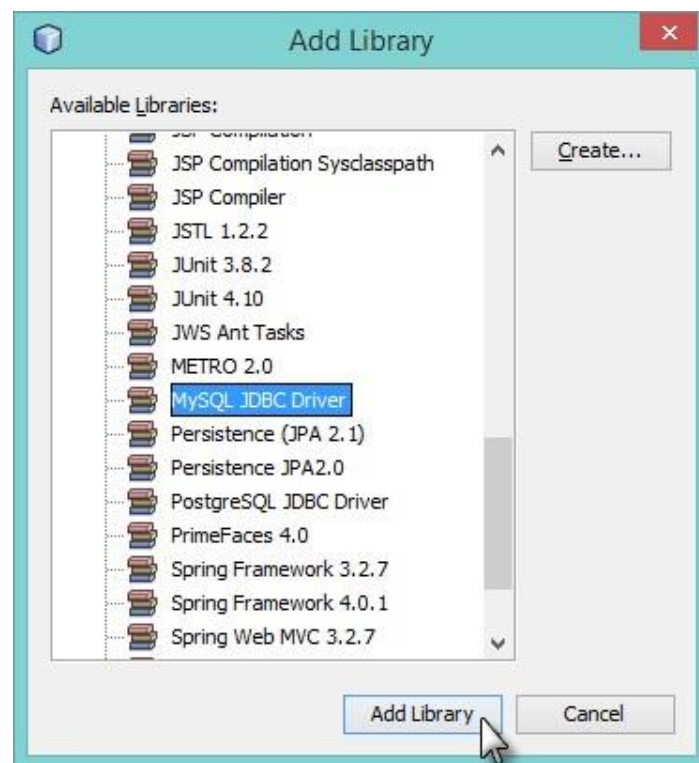
- JDBC.

Ahora veremos el caso de Java. Cabe aclarar que este proceso requeriría la codificación completa de una aplicación Java para poder realizar el acceso, por eso mismo y por cuestiones de tiempo límite que no permiten la construcción de dicha aplicación, sólo se mostrará brevemente su forma de acceso.

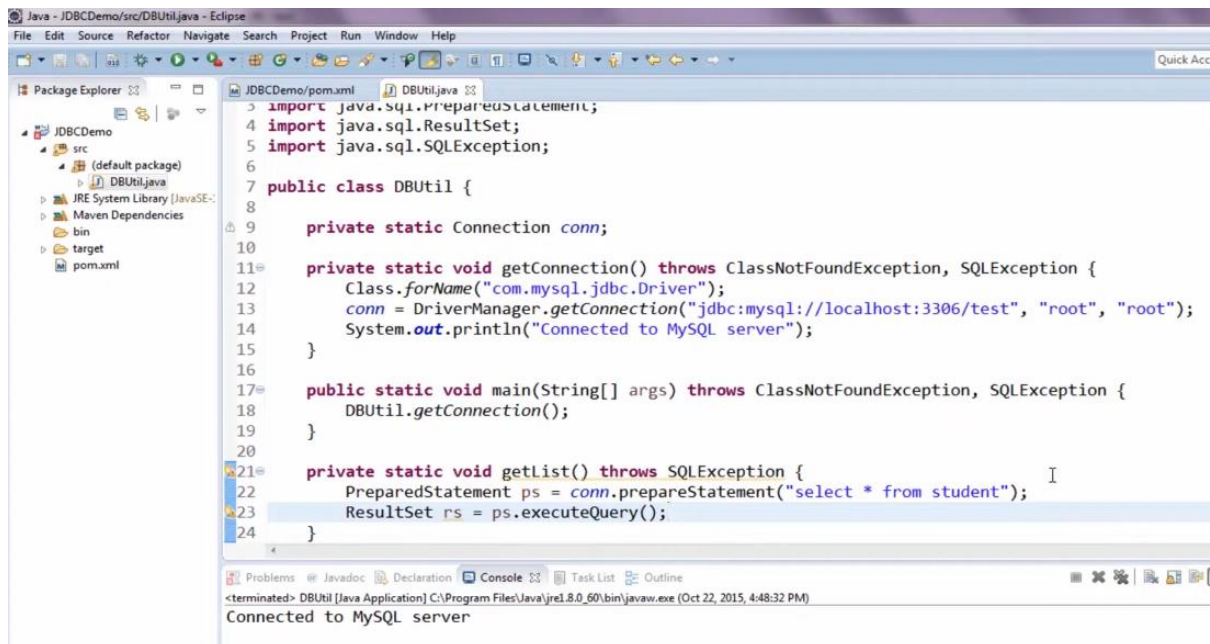
El funcionamiento sería el siguiente: se desarrolla una aplicación en Java con un IDE (NetBeans, por ejemplo) en la que se incluye un controlador hecho por Oracle llamado JDBC, que contiene una librería con los métodos necesarios para crear dicha aplicación y que pueda conectarse a una base de datos.



Debemos añadir esa librería .jar a la aplicación de Java para poder trabajar con los métodos de conexión:



El último paso sería codificar la aplicación Java para conectarse a la base de datos y poder consultar su información mediante el método “ResultSet”, leyendo la información con código SQL incrustado dentro del código Java:



```
Java - JDBCdemo/src/DBUtil.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer
JDBCdemo
src
  (default package)
    DBUtil.java
JRE System Library [JavaSE-6]
Maven Dependencies
bin
target
pom.xml

JDBCdemo/pom.xml DBUtil.java
1 import java.sql.*;
2
3
4 import java.sql.ResultSet;
5 import java.sql.SQLException;
6
7 public class DBUtil {
8
9     private static Connection conn;
10
11     private static void getConnection() throws ClassNotFoundException, SQLException {
12         Class.forName("com.mysql.jdbc.Driver");
13         conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "root");
14         System.out.println("Connected to MySQL server");
15     }
16
17     public static void main(String[] args) throws ClassNotFoundException, SQLException {
18         DBUtil.getConnection();
19     }
20
21     private static void getList() throws SQLException {
22         PreparedStatement ps = conn.prepareStatement("select * from student");
23         ResultSet rs = ps.executeQuery();
24     }
25 }

Problems Javadoc Declaration Console Task List Outline
<terminated> DBUtil [Java Application] C:\Program Files\Java\jre1.8.0_60\bin\javaw.exe (Oct 22, 2015, 4:48:32 PM)
Connected to MySQL server
```

- Actividad 3: rellenar campos de una base de datos.

En esta última actividad mostraremos cómo hemos creado el script con la creación de la base de datos “departamentoVentas”. Usaremos la sintaxis MySQL para crearla.

Para crear una tabla, primero necesitamos escribir “drop database if exists” para eliminar e una posible base de datos anterior para que no cree conflicto al crearla de nuevo, ya que una base de datos es persistente una vez creada. Seguidamente escribimos “create database if not exists” para crearla y definimos la codificación de caracteres estándar en español.

El siguiente paso es “use departamentoVentas” para indicar qué base de datos queremos usar; luego crearemos una tabla con “create table if not exists” junto con el nombre de la tabla, los campos y sus atributos con el tipo de dato que contendrán y si debe ser obligatoriamente rellenado o no; al final de la tabla se indica el tipo de motor que se usará con “engine”.

En el siguiente paso añadiremos las restricciones de clave primaria y de clave externa con el tipo de borrado y modificación en cascada. Finalmente, usaremos “insert into” y “values” para rellenar cada campo de la tabla con el dato correspondiente. En la última imagen se puede ver el script completo.

```

1 drop DATABASE if EXISTS departamentoVentas;
2 CREATE DATABASE IF NOT EXISTS departamentoVentas
3     DEFAULT CHARACTER SET UTF8
4     DEFAULT COLLATE UTF8_SPANISH_CI;
5
6 USE departamentoVentas;
7
8 CREATE TABLE if NOT EXISTS propietario (
9     pnum varchar(5) not null,
10    nombre varchar(30) not null,
11    apellido varchar(50) not null,
12    direccion varchar(50) not null,
13    telefono int(9) NOT NULL
14 ) engine INNODB;
15
16 create table if not EXISTS inmueble (
17     inum varchar(5) not null,
18     calle varchar(50) not null,
19     area varchar(30),
20     poblacion varchar(30) not null,
21     tipo varchar(10) not null,
22     habitaciones tinyint(2) not null,
23     alquiler int(3) not null,
24     pnum varchar(5) not null
25 ) ENGINE INNODB;
26
27 CREATE TABLE if not EXISTS inquilino (
28     qnum varchar(5) not null,
29     nombre varchar(30) not null,
30     apellido VARCHAR(50) not null,
31     direccion VARCHAR(50) not null,
32     telefono int(9) not null,
33     tipo varchar(10) not null,
34     alquiler int(3) not null
35 ) engine innodb;
36
37 alter table propietario
38 add CONSTRAINT pk_propietario PRIMARY KEY (pnum);
39
40 alter table inmueble
41 add CONSTRAINT pk_inmueble primary key (inum);
42
43 alter table inmueble
44 ADD CONSTRAINT fk_inmueble FOREIGN key (pnum) REFERENCES propietario(pnum)
45     on delete CASCADE
46     on update CASCADE;
47
48 alter table inquilino
49 add CONSTRAINT pk_inquilino PRIMARY key (qnum);
50
51 insert into propietario (pnum, nombre, apellido, direccion, telefono) values ("P46", "Amparo", "Felipe", "Asensio
52 24, Castellón", 964230680);
53 insert into propietario (pnum, nombre, apellido, direccion, telefono) values ("P87", "Manuel", "Alejandro", "Av.
54 Libertad 15, Vinaroz", 964450760);
55 insert into propietario (pnum, nombre, apellido, direccion, telefono) values ("P40", "Alberto", "Estrada", "Av. del
56 Puerto 52, Castellón", 964200740);
57 insert into propietario (pnum, nombre, apellido, direccion, telefono) values ("P93", "Yolanda", "Robles", "Purísima 4,
58 Segorbe", 964710430);
59
60 insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IA14", "En medio,
61 128", "Centro", "Castellón", "Casa", 6, 600, "P46");
62 insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IL94", "Riu Ebre,
63 24", "Ronda Sur", "Castellón", "Piso", 4, 350, "P87");
64 insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IG4", "Sorell, 5",
65 "Grao", "Castellón", "Piso", 3, 300, "P40");
66 insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IG36", "Alicante,
67 1", "", "Segorbe", "Piso", 3, 325, "P93");
68 insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IG21", "San
69 Francisco, 10", "", "Vinaroz", "Casa", 5, 550, "P87");
70 insert into inmueble (inum, calle, area, poblacion, tipo, habitaciones, alquiler, pnum) values ("IG16",
71 "Capuchinos, 19", "Rafalafena", "Castellón", "Piso", 4, 400, "P93");
72
73 insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q76", "Juan", "Felip",
74 "Barceló 47, Castellón", 964282540, "Piso", 375);
75 insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q56", "Ana",
76 "Grangel", "San Rafael 45, Almazora", 964551110, "Piso", 300);
77 insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q74", "Elena",
78 "Abaso", "Navarra 76, Castellón", 964205560, "Casa", 700);
79 insert into inquilino (qnum, nombre, apellido, direccion, telefono, tipo, alquiler) values ("Q62", "Alicia",
80 "Mori", "Alloza 45, Castellón", 964229580, "Piso", 550);

```

BIBLIOGRAFÍA/WEBGRAFÍA

- Delgado, H. (2021): *Cómo crear una Base de Datos en phpMyAdmin en MySQL*, en Akus.net [recurso electrónico], consulta el 23 de noviembre de 2021 (<https://disenowebakus.net/crear-una-base-de-datos-phpmyadmin-mysql-php.php>).
- Fondevila Gómez, J. (2021): *Creación táboa de datos*, en Google Docs [recurso electrónico], consulta el 23 de noviembre de 2021 (<https://docs.google.com/spreadsheets/d/1-8MmC-Ms8CjT9lEmpwGd8P5evLWNxckHPgaFtTn734I/edit?usp=sharing>).
- Lopez, E. (2018): *Conectar una Base de datos MySQL con PHP*, en UnProgramador [recurso electrónico], consulta el 23 de noviembre de 2021 (<https://unprogramador.com/conectar-una-base-de-datos-mysql-con-php/>).
- Robledano, Á. (2019): *Qué es MySQL: Características y ventajas*, en OpenWebinars [recurso electrónico], consulta el 23 de noviembre de 2021 (<https://openwebinars.net/blog/que-es-mysql/>).