

A graphic on the left side of the slide features four overlapping horizontal bars in purple, orange, yellow, and blue. The text 'Agencia de Aprendizaje a lo largo de la vida' is written across these bars in white. An orange arrow points to the right from the end of the orange bar.

Agencia de  
Aprendizaje  
a lo largo  
de la vida

# FULL STACK PYTHON

## Clase 22

SQL 1

# Introducción a Base de Datos



# Les damos la bienvenida

Vamos a comenzar a grabar la clase

## Clase 21

**SPA y Asincronía**

- Introducción a SPA.
- SPA. ¿Qué es y qué beneficios tiene?
- Ejemplo práctico de un SPA en Vue.
- Enviar y pedir datos a un servidor
- Asincronía.
- Consumo de API REST a través de fetch y Axios..

## Clase 22

**Introducción a Base de Datos**

- ¿Qué es una Base de datos?
- BBDD relacionales y no relacionales.
- Entorno MySQL. Instalación. Clientes MySQL.
- DER. Entidad, atributo y tipo de datos. Primary key.
- Creación de una BD.
- Backup y restauración de bases de datos.

## Clase 23

**Lenguaje y Sublenguajes SQL**

- Gestión y manipulación de datos con SQL.
- Gestión y manipulación de datos.
- Sublenguajes DDL y DML.
- Consultas: Estructura consulta SQL. Cláusulas SELECT, FROM, WHERE.
- Alias y literales. ORDER BY.

# Bases de Datos

Una **base de datos** es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Es una forma de almacenar información en forma más eficiente de lo que sería un archivo de texto.

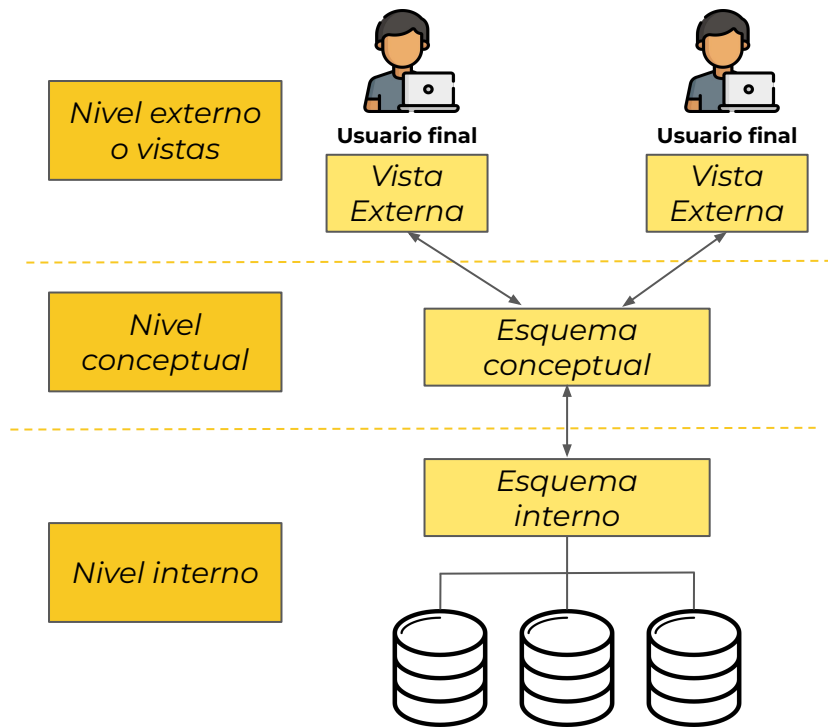
Se crean y mantienen a través de un **DBMS o motor de base de datos**, que facilita la definición, construcción, manipulación y compartición de bases de datos entre usuarios y aplicaciones.

Contienen datos que pueden diferir entre sí pero poseen relaciones en común. Por ejemplo: alumnos y libros en el contexto de una biblioteca.

# ¿Por qué son necesarias las bases de datos?

Si tenemos una empresa es conveniente tener un sistema para registrar ventas, empleados, sueldos, etc. Estos datos pueden guardarse en una **base de datos** con tablas para registrar esa información.

Esta información podrá ser **consultada** por usuarios (*vistas*), **administrada** por un sistema con tablas relacionadas (*esquema conceptual*) y **almacenada** en una base de datos (*esquema interno*). [Video](#)



# Ejemplo de uso de una base de datos

Los **productos** existen en la base de datos; al buscarlos, la aplicación consulta la lista de productos de acuerdo a la condición de la búsqueda..

Búsquedas relacionadas: cafetera dulce gusto precio - cafetera dulce gusto - cafetera philips - cafetera - oster prima latte - cafetera oster - nespresso

Electrodomésticos y Aires A.C. > Pequeños  
Electrodomésticos > Para Cocina  
Cafeteras

Ordenar por Más relevantes ▾

**Cafetera**  
5.790 resultados

Llegan hoy ☐

**FULL** te ahorra envíos  
Con tu carrito de compras ☐

Envío gratis ☐

Condición  
Nuevo (1.681)  
Usado (1.250)

Saeco Lirika Otc Cafetera Express Automatica Cappuccino Color Gris Oscuro  
\$ 174.900  
Llega gratis mañana sábado

Cafetera Nescafé Moulinex Dolce Gusto Piccolo XS de corte manual red para cápsulas monodosis 220V  
Vendido por Moulinex  
\$ 37.999  
\$ 31.910 16% OFF  
Llega gratis el lunes

★★★★★ 1302

Para ingresar, los **usuarios** deben haberse dado de alta primero, de esta manera serán ingresados a la base de datos..

Las **categorías** son también datos que se encuentran en la base de datos, por eso se puede contar cuántos productos hay en cada una.

Con estos datos podemos empezar a pensar que podemos organizar la información en tablas: productos, proveedores, usuarios, categorías. Estas tablas tendrán atributos asociados que serán **sustantivos**.

Todos los **datos de los productos** que aparecen en la lista están dentro de la base de datos almacenados (precio, ubicación, vendedor, descripción, etc) .

# SGBD más conocidos

- Existen diferentes Sistemas de Gestión de Base de Datos:
  - **Relacionales:** MySQL, MaríaDB, PostgreSQL, SQL Server, entre otras.
  - **No relacionales:** Mongo DB, Redis, Elasticsearch y Cassandra.





# Bases de datos relacionales

Las bases de datos relacionales permiten **gestionar el acceso a los datos, su almacenamiento, modificación, eliminación, consulta y el múltiple acceso** desde distintas aplicaciones y usuarios. Además permiten gestionar permisos para que una parte de los datos estén disponibles para ciertos usuarios y no para otros. Todo esto es resuelto por un motor de base de datos, generando una independencia entre la base de datos y la aplicación que la consulte.

Son más eficientes en cuanto al almacenamiento y búsqueda de información, comparadas con un archivo de texto plano donde la lectura la haríamos en forma secuencial (línea por línea), cargándola toda en memoria, ni tampoco podríamos acceder y guardar información al mismo tiempo.

# Bases de datos no relacionales

- No tienen un identificador que sirva de relación entre un conjunto de datos y otros.
- Normalmente la información se organiza en **documentos** y es muy útil cuando no tenemos un esquema exacto de lo que se va a almacenar.
- Suelen utilizar **documentos JSON**, a diferencia de las bases de datos relacionales que despliegan su información en tablas.
- Las bases de datos más competitivas suelen permitir operaciones de ambos tipos: relacionales y no relacionales.

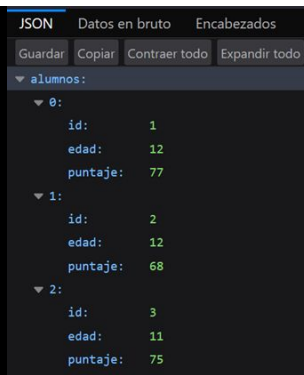
*La diferencia entre el éxito y el fracaso de una base de datos recae en el **diseño del modelo**. De nada sirve elegir la base de datos más apropiada para nuestro sistema, si luego no se hace un buen diseño.*

# Bases de datos no relacionales

Una tabla puede transformarse en documentos, cada uno formado por cada fila de la tabla. Solo es una cuestión de visualización.

ID	EDAD	PUNTAJE
1	12	77
2	12	68
3	11	75

```
{
  "alumnos": [
    {
      "id": 1,
      "edad": 12,
      "puntaje": 77
    },
    {
      "id": 2,
      "edad": 12,
      "puntaje": 68
    },
    {
      "id": 3,
      "edad": 11,
      "puntaje": 75
    }
  ]
}
```



# Bases de datos no relacionales

En una base de datos no relacional una unidad de datos puede llegar a ser demasiado compleja como para plasmarlo en una tabla.

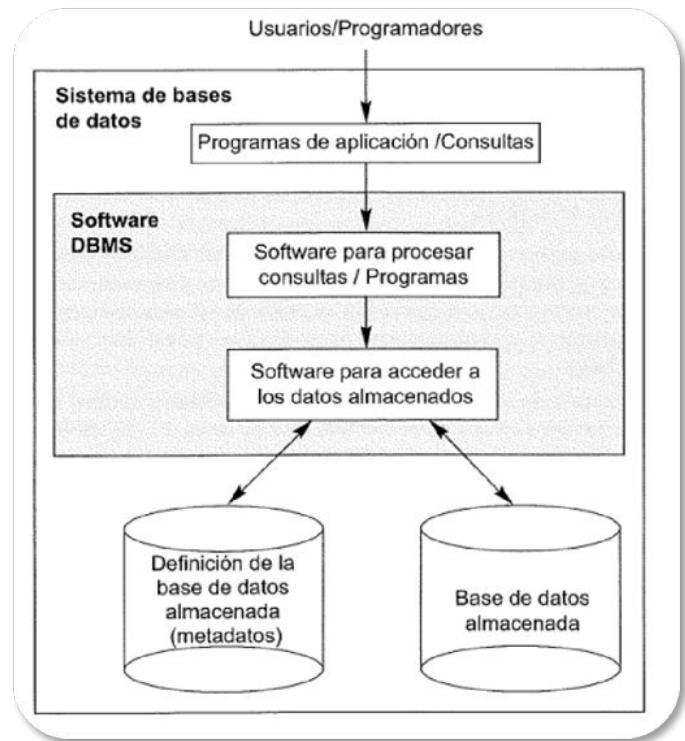
En la imagen de la derecha al tener elementos jerárquicos, es más difícil plasmarlo en una tabla plana. Una solución sería plasmarlo en varias tablas y, por tanto, necesitar de relaciones.

```
1  [
2  {
3    "student_id": 1,
4    "age": 12,
5    "subjects": {
6      "mathematics": {
7        "scores": [
8          7,
9          8,
10         7,
11         10
12       ],
13       "final_score": 8
14     },
15     "biology": {
16       "scores": [
17         6,
18         6,
19         5,
20         7
21       ],
22       "final_score": 6
23     }
24   }
25 }
26 ]
```

# Ubicación de las Bases de Datos

Dentro de un entorno de un sistema de bases de datos se encuentran en el nivel más bajo. Generalmente se los considera como la **parte "física"**, ya que, aunque sean un contenido lógico, se encuentran almacenadas o creadas en un dispositivo físico. Por ejemplo: un servidor.

Para que un usuario acceda a los datos en una Base de Datos, necesita de un software especial conocido como **SGBD (Sistema Gestor de Base de Datos)** o **DBMS (Data Base Managment System)**.



# ¿Cómo empezamos a pensar en una BD?

Se realiza una **entrevista** con el cliente para hacer un relevamiento de datos. El relato del cliente nos permite identificar qué información va a necesitar. Si el cliente nos cuenta que tiene empleados, su registro es candidato a ser una tabla que va a tener **atributos asociados** (*sustantivos*): nombre, apellido, DNI, fecha de nacimiento, etc. Estas tablas se llaman **entidades** y permitirán almacenar los datos. Por ejemplo: en el sistema de venta online tendría productos, marcas y conceptos candidatos a ser tablas o atributos.

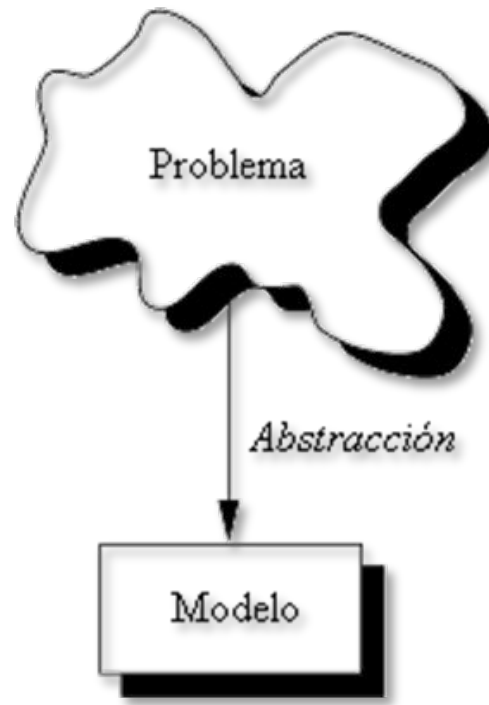
Entidad (tabla)	Empleados			
	Campos/Atributos (columnas)			
Filas/Registros	DNI	Apellido	Nombre	Fecha Nac.
	12.345.678	Gómez	Juan	25/09/1945
	23.456.789	Fernández	Ana	12/07/1973

# Abstracción y Modelado de datos

La **abstracción** de datos es una técnica o metodología que permite diseñar estructuras de datos, consiste en representar bajo ciertos lineamientos de formato las características esenciales de una estructura de datos.

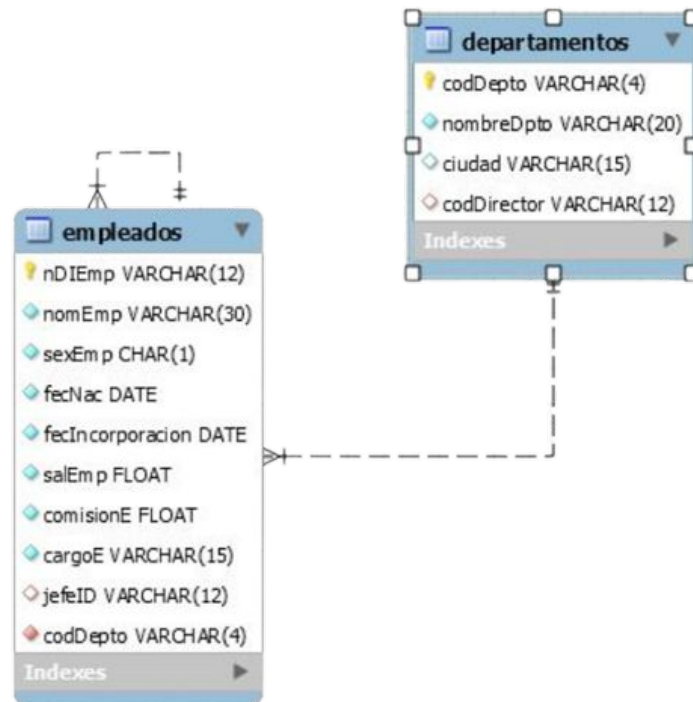
El **modelado** de datos permite describir:

- **Las estructuras de datos de la base:**  
Tipo de datos y sus relaciones.
- **Las restricciones de integridad:**  
Conjunto de condiciones que deben cumplir los datos para reflejar la realidad deseada.
- **Operaciones de manipulación de los datos:** agregado, borrado, modificación y recuperación de datos.



# Modelo Entidad-Relación

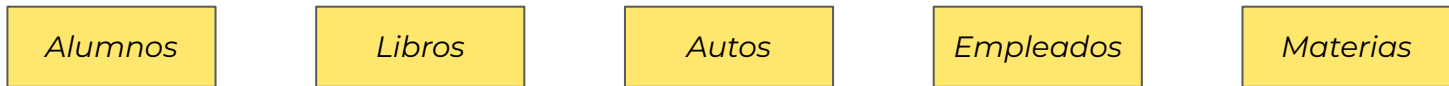
- Método para diseñar Bases de Datos. Se representa a través de diagramas y está formado por varios elementos.
- Para modelar los datos utilizamos un **Diagrama de Entidad-Relación (DER)**, que pertenece al **Lenguaje de Modelado Unificado** (UML, *Unified Modeling Language*). Este diagrama representa entidades (tablas) y las relaciones lógicas entre ellas.
- Una vez modelados los datos se implementan en un SGBD.



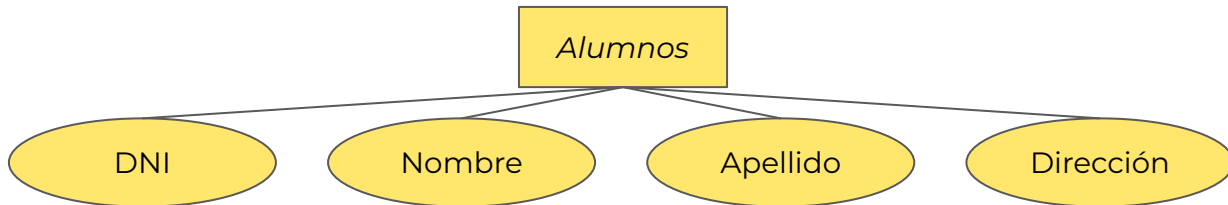


# Diagrama Entidad Relación | Componentes

**ENTIDADES:** Representan cosas u objetos (ya sean reales o abstractos). Se representan en los diagramas como **rectángulos**. Se suelen colocar en plural.



**ATRIBUTOS:** Definen o identifican las características propias y por lo general únicas de una entidad, pueden ser de distintos tipos (numéricos, texto, fecha, etc.), y se representan por medio de un **óvalo o elipse**. Cada entidad contiene distintos atributos, que dan información sobre ella misma.



# Diagrama Entidad Relación | Componentes

**RELACIONES:** Se representan con rombos y tienen una característica conocida como “cardinalidad”, que indica el sentido y la cantidad de “relaciones” existentes entre una entidad y otra. Los tipos de relaciones pueden ser:

- **1 a N (uno a muchos):** **una** persona tiene **muchos** autos y viceversa, **muchos** autos pueden ser de **una** persona.



- **1 a 1 (uno a uno):** a **un** alumno le pertenece únicamente **un** boletín y viceversa, **un** boletín pertenece únicamente a **un** alumno.



# Diagrama Entidad Relación | Componentes

- **N a N (muchos a muchos):** **muchos** alumnos pueden tener **muchas** materias y viceversa, **muchas** materias pueden contener a **muchos** alumnos.



# Diagrama Entidad Relación | Ejemplo

Una empresa de venta de electrodomésticos tiene clientes, pedidos y productos. Desea modelar a través de un DER cómo se implementaría la Base de Datos:

1. Detectar las **entidades**:



*Clientes*

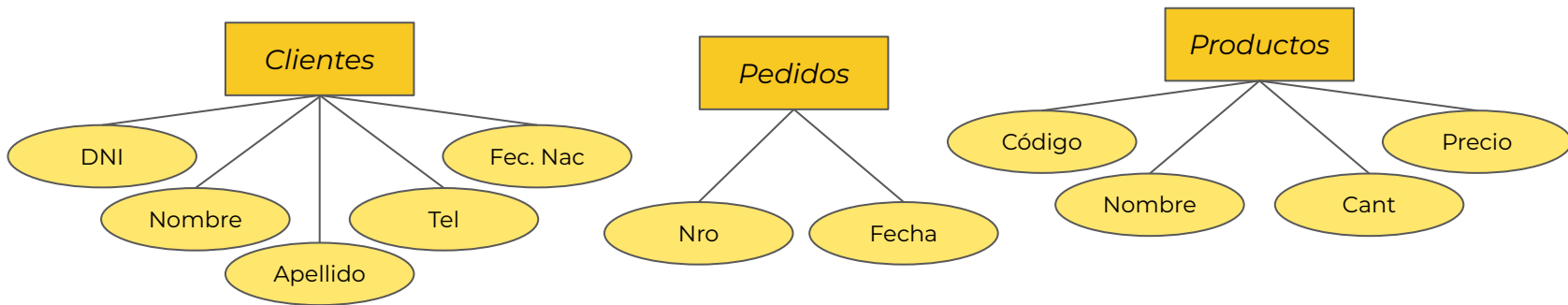


*Pedidos*



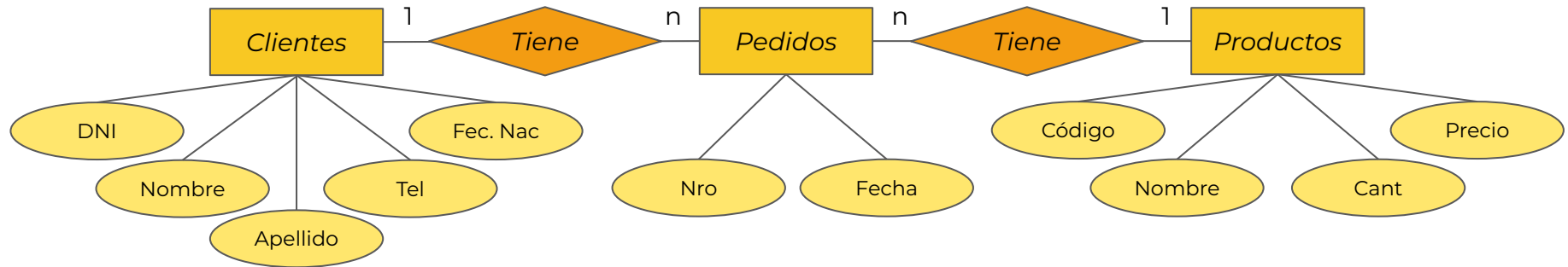
*Productos*

1. Detectar los **atributos**:



# Diagrama Entidad Relación | Ejemplo

3. Conocidas las **entidades** y sus **atributos**, establecemos las **relaciones** existentes entre sí: un cliente puede realizar varios pedidos (ya que en cada compra que realice, se efectuará un nuevo pedido) y que un pedido puede tener varios productos (ya que una misma compra/pedido pueden haber más de un artículo encargado).



# Diagrama Entidad Relación | Ejemplo

4. Una vez que tenemos el DER lo pasamos a forma de TABLA:

Clientes			
Id	RazonSocial	CUIT	Tel

Pedidos		
Nro	IdCliente	Fecha

ItemPedidos			
Pedido	IdProducto	Cantidad	Precio

Productos		
Código	Nombre	Precio

# Tipos de Datos

Los **atributos de las entidades** deben cumplir o pueden ser únicamente de ciertos tipos de datos. Entre ellos, los más importantes / utilizados son:

- **NUMÉRICOS:** se utilizan para representar valores/atributos de carácter numérico tanto enteros, como decimales.
- **TEXTO (VARCHAR):** se utilizan para representar valores de texto, como ser cadenas de caracteres.
- **DATE (FECHA):** se utilizan para representar fechas, horas, minutos, segundos, etc.
- **BOOLEAN (LÓGICO):** se utilizan para representar valores verdaderos o falsos (*true* or *false*).

# Tipos de Datos

Los tipos de datos del ejemplo anterior podrían ser los siguientes:

Clientes			
ID	Razon Social	Tel	CUIT
INT	VARCHAR(100)	VARCHAR(20)	DATE

Productos		
Id	Nombre	Precio
INT	VARCHAR(20)	DOUBLE

Pedidos		
Nro	IdCliente	Fecha
INT	INT	DATE

ItemPedidos			
IdPedido	IdProducto	Cantidad	Precio
INT	INT	INT	DOUBLE



# Primary key y Foreign Key

- Las **claves primarias (Primary Keys)** son valores que identifican de manera única a cada fila o registro de una tabla, esto quiere decir que no se puede repetir. Por ejemplo: un DNI, un código de producto, etc.
- Una **clave foránea (Foreign Key)** es un campo de una tabla “X” que sirve para enlazar o relacionar entre sí con otra tabla “Y” en la cual el campo de esta tabla es una clave primaria (Primary Key). Para que un campo sea una clave, esta tiene que ser una clave primaria en otra tabla.




# Primary key y Foreign Key


En el ejemplo anterior podemos identificar las claves primarias, que identificarán de manera única a cada fila o registro de una tabla





Clientes			
ID	RazonSocial	Telefono	CUIT
INT	VARCHAR(100)	VARCHAR(20)	varchar



Productos		
Id	Nombre	Precio
INT	VARCHAR(20)	DOUBLE



Pedidos		
Nro	IdCliente	Fecha
INT	INT	DATE



ItemPedidos			
IdPedido	IdProducto	Cantidad	Precio
INT	INT	INT	DOUBLE

# Instalación MySQL Server | Paso a paso

1. Descargar el instalador de <https://dev.mysql.com/downloads/installer/>
2. Ejecutar el instalador y seleccionar **Server Only Execute**.
3. Darle a *next* hasta llegar a la pantalla **Authentication Method**:  
Seleccionar *Use Legacy Authentication Method*.
4. En la siguiente pantalla *setear* contraseña para el usuario root en MySQL  
*Root Password*.
5. Darle next y al llegar a **Apply Configuration** apretar *Execute*.

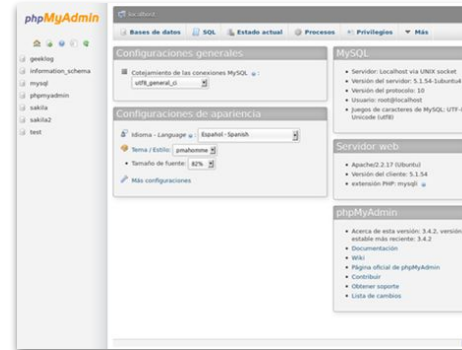
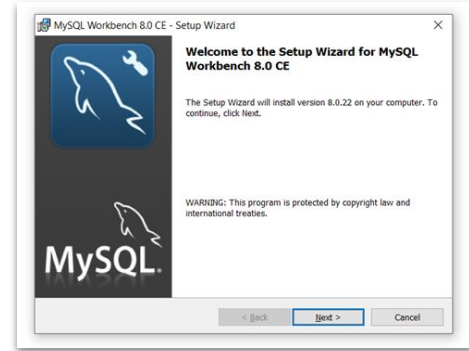
**Importante:** Para la instalación se recomienda ver los tutoriales que aparecen en esta presentación y utilizar los archivos que están en el Aula Virtual y la carpeta de Drive compartida.

# Herramientas para manejo de Base de Datos

**MYSQL WORKBENCH:** Es una herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL.

**PHPMYADMIN:** Es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet.

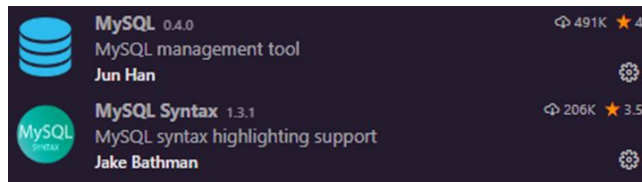
Puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios y exportar datos en varios formatos.



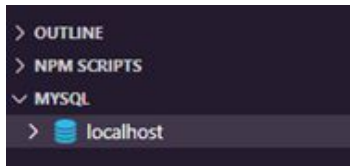
# Herramientas para manejo de Base de Datos

## VISUAL STUDIO CODE:

1. Descargar las siguientes extensiones en VSCode:



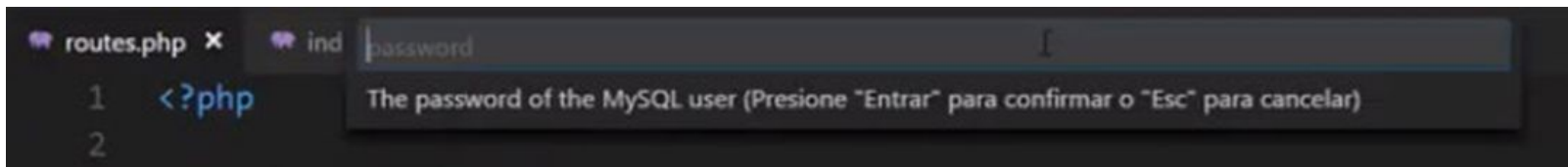
1. Cerrar y volver a abrir Visual Studio Code.
2. Apretar el símbolo + en el apartado MySQL. Al ser la primera vez que se configura no aparecerá ninguna base de datos:



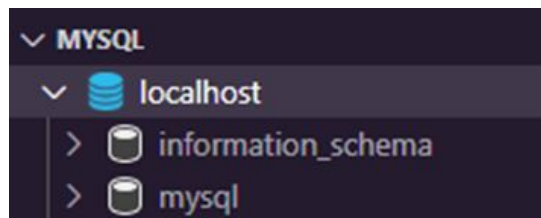
# Herramientas para manejo de Base de Datos

## VISUAL STUDIO CODE (continuación):

4. Rellenar usuario y contraseña, los demás datos (puerto y SSL) y apretar ENTER.



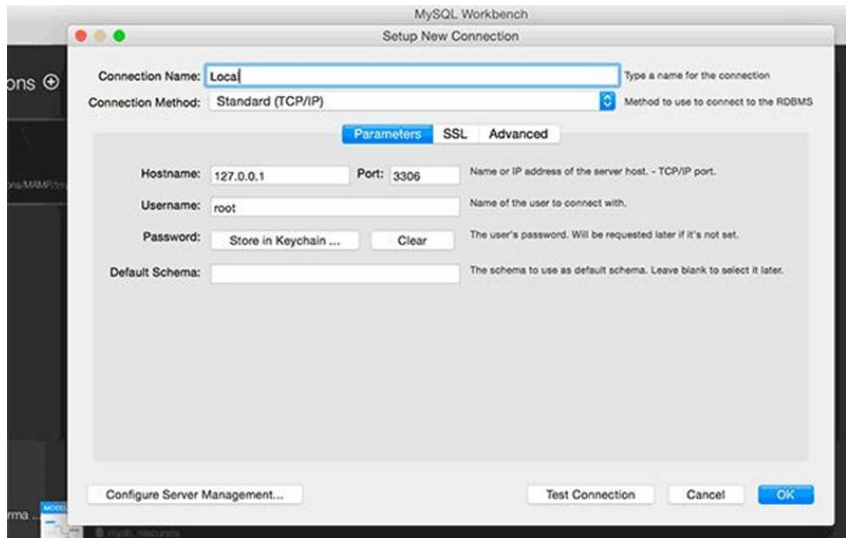
4. Deberá aparecer localhost:



# Conectarse al servidor MySQL

Para que un programa cliente (VSCode, *MySQL Workbench*, *phpMyAdmin*, etc.) se conecte al servidor MySQL, debés utilizar los parámetros de conexión adecuados, como el nombre del host donde se ejecuta el servidor y el nombre de usuario y contraseña de tu cuenta MySQL.

Cada parámetro de conexión tiene un valor predeterminado, pero puede anular los valores predeterminados según sea necesario utilizando las opciones del programa especificadas en la línea de comandos o en un archivo de opciones.

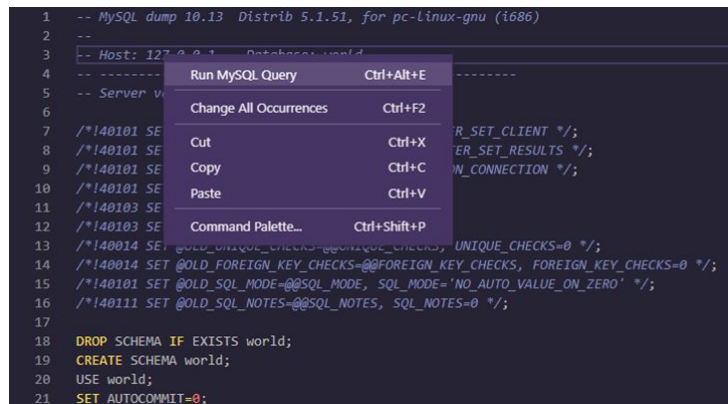
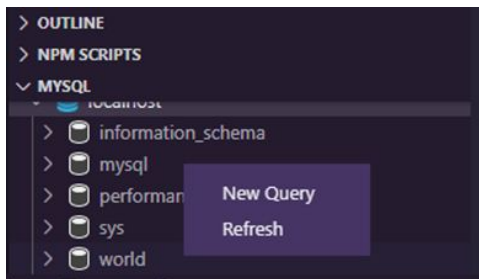


**MySQL Workbench:** En MySQL Connections deben establecer una nueva conexión con el signo + y poner los datos de la nueva conexión.

# Crear una base de datos de prueba

**WORLD.SQL:** Los pasos se detallan para **VSCode** pero para MySQL Workbench y phpMyAdmin resultan similares.

1. Descargar **world.sql** del Aula Virtual y abrir con Visual Studio Code.
2. Apretar botón derecho > Run SQL Query.





# Ver una BD y acceder a tablas

Una vez que nos conectamos al LocalHost, previa conexión con XAMPP, podremos acceder a ver las bases de datos y sus tablas:

The screenshot shows a database management interface with a sidebar on the left containing a tree view of schemas and tables. The main area displays a table named 'usuarios' with columns: idx, usuario, nombre, sexo, nivel, email, telefono, marca, compania, saldo, and activo. A context menu is open over the 'usuarios' table, showing options like 'Select Rows - Limit 1000', 'Table Inspector', 'Copy to Clipboard', 'Send to SQL Editor', 'Create Table...', 'Create Table Like...', 'Alter Table...', 'Table Maintenance...', 'Drop Table...', and 'Truncate Table...'. An arrow points from the 'usuarios' table to the 'Registros seleccionados' annotation. Another arrow points from the 'Bases de datos (schemas)' annotation to the 'usuarios' table. A third arrow points from the 'Tablas (entidades)' annotation to the 'usuarios' table.

Bases de datos (schemas)

Tablas (entidades)

Registros seleccionados

# Creando nuestra primer BD

Crearemos nuestra primera BD llamada **empleados\_departamentos**. Utilizaremos el archivo *bd\_empleados\_departamentos.sql* para ejecutar la sentencia SQL que la crea. Para ello seguiremos los siguientes pasos:

1. Abrir el archivo que contiene la sentencia SQL.
2. Crear una nueva consulta SQL y pegar todo el texto dentro.
3. Ejecutar desde el ícono del rayo.
4. Quedará creada la Base de Datos con dos tablas: *departamentos* y *empleados*



# Creando nuestra primer BD

Para **crear una tabla** utilizamos CREATE TABLE e indicamos cuáles son las columnas (atributos/campos) que conformarán nuestra tabla (1).

Para agregar registros utilizamos INSERT INTO nombredelatabla VALUES y estos datos van separados por comas en el mismo orden en que fueron incorporados los campos (2). Los datos serán incorporados en la tabla (3).

1

```
CREATE TABLE `departamentos` (  
  `codDepto` varchar(4) COLLATE utf8_bin NOT NULL,  
  `nombreDpto` varchar(20) COLLATE utf8_bin NOT NULL,  
  `ciudad` varchar(15) COLLATE utf8_bin DEFAULT NULL,  
  `codDirector` varchar(12) COLLATE utf8_bin DEFAULT NULL,  
  PRIMARY KEY (`codDepto`),  
  KEY `FK_EmpDir` (`codDirector`),  
  CONSTRAINT `FK_EmpDir` FOREIGN KEY (`codDirector`) REFERENCES `empleados` (`codEmp`),  
  ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin  
);
```

2

```
LOCK TABLES `departamentos` WRITE;  
INSERT INTO `departamentos` VALUES ('1000','GERENCIA','CIUDAD REAL','31.840.269');  
INSERT INTO `departamentos` VALUES ('1500','PRODUCCIÓN','CIUDAD REAL','16.211.383');  
INSERT INTO `departamentos` VALUES ('2000','VENTAS','CIUDAD REAL','31.178.144');  
INSERT INTO `departamentos` VALUES ('2100','VENTAS','BARCELONA','16.211.383');  
INSERT INTO `departamentos` VALUES ('2200','VENTAS','VALENCIA','16.211.383');  
INSERT INTO `departamentos` VALUES ('2300','VENTAS','MADRID','16.211.383');  
INSERT INTO `departamentos` VALUES ('3000','INVESTIGACIÓN','CIUDAD REAL','16.211.383');  
INSERT INTO `departamentos` VALUES ('3500','MERCADERO','CIUDAD REAL','16.211.383');  
INSERT INTO `departamentos` VALUES ('4000','MANTENIMIENTO','CIUDAD REAL','16.211.383');  
INSERT INTO `departamentos` VALUES ('4100','MANTENIMIENTO','BARCELONA','16.211.383');  
INSERT INTO `departamentos` VALUES ('4200','MANTENIMIENTO','VALENCIA','16.211.383');  
INSERT INTO `departamentos` VALUES ('4300','MANTENIMIENTO','MADRID','16.211.383');  
UNLOCK TABLES;
```

3

codDepto	nombreDpto	ciudad	codDirector
1000	GERENCIA	CIUDAD REAL	31.840.269
1500	PRODUCCIÓN	CIUDAD REAL	16.211.383
2000	VENTAS	CIUDAD REAL	31.178.144
2100	VENTAS	BARCELONA	16.211.383
2200	VENTAS	VALENCIA	16.211.383
2300	VENTAS	MADRID	16.759.060
3000	INVESTIGACIÓN	CIUDAD REAL	16.759.060
3500	MERCADERO	CIUDAD REAL	22.222.222




# Ver los datos de las tablas



Haciendo clic con el botón derecho en nuestra tabla y seleccionando **Select Rows – Limit 1000** veremos los resultados de nuestra primer consulta SQL:


1 • `SELECT * FROM empleadoss_departamentoss.departamentos;`

Result Grid

Filter Rows:

Edit:   

Export/Import:  

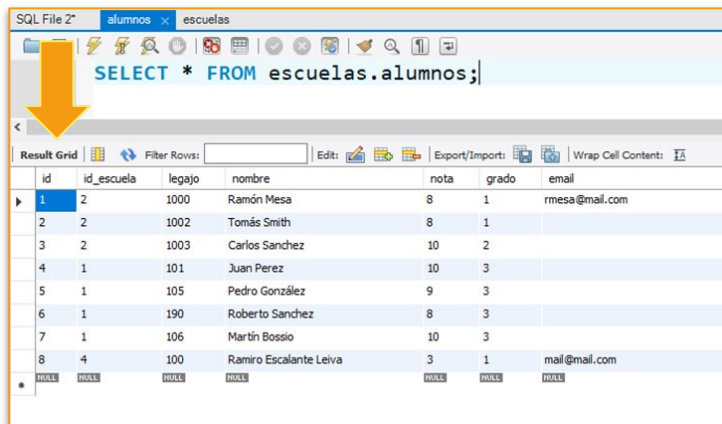
Wrap Cell Content: 

	codDepto	nombreDpto	ciudad	codDirector
	1000	GERENCIA	CIUDAD ...	31.840.269
	1500	PRODUCCIÓN	CIUDAD ...	16.211.383
	2000	VENTAS	CIUDAD ...	31.178.144
	2100	VENTAS	BARCEL...	16.211.383
	2200	VENTAS	VALENCIA	16.211.383
	2300	VENTAS	MADRID	16.211.383

Con CTRL + Enter ***ejecutamos*** la consulta, o con el ícono del rayito: ⚡

# Claves principales

Una clave principal es un **identificador único para cada registro** de la tabla. Para definirla tenemos que analizar las claves candidatas, aquellas que podrían ser claves principales, un valor propio de ese registro que identifique **de forma única** esa instancia del dato. Cada registro debería tener un identificador único, para evitar duplicados:



SQL File 2\* alumnos escuelas

```
SELECT * FROM escuelas.alumnos;
```

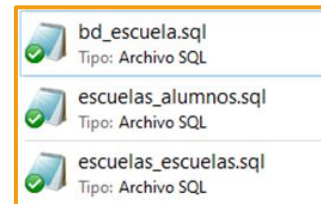
Result Grid

	id	id_escuela	legajo	nombre	nota	grado	email
▶	1	2	1000	Ramón Mesa	8	1	rmesa@mail.com
	2	2	1002	Tomás Smith	8	1	
	3	2	1003	Carlos Sanchez	10	2	
	4	1	101	Juan Perez	10	3	
	5	1	105	Pedro González	9	3	
	6	1	190	Roberto Sanchez	8	3	
	7	1	106	Martín Bossio	10	3	
	8	4	100	Ramiro Escalante Leiva	3	1	mail@mail.com
*	id	id_escuela	legajo	nombre	nota	grado	email

# Exportar una BD (backup)

Podemos exportar una Base de datos desde Workbench con el objetivo de hacer un backup:

1. Ir a **Server – Data export**
2. Seleccionar la base de datos (*schema*) que se desea exportar del cuadro de la izquierda dentro de Object Selection.
3. Seleccionar del cuadro de la derecha aquellas tablas que se desean exportar.
4. Determinar a qué carpeta se exportará la base de datos y cómo se exportarán los datos:
  - a. Si elegimos **Export to Dump Project Folder** se exportarán las tablas por separado.
  - b. Con **Export to Self-Contained File** podremos darle un nombre al archivo, pero con todas las tablas juntas.
5. Hacer clic en **Start Export** y colocar la contraseña del host.



# Material extra

# Artículos de interés

Material extra:

- [¿Qué es un gestor de datos y para qué sirve?](#)
- [¿Qué son las bases de datos NoSQL?](#)
- [Diagrama entidad relación](#)
- [¿Qué es y para qué sirve UML?](#)
- [Lenguaje de definición de datos \(DDL\)](#)
- [Curso de SQL de Píldoras informáticas](#)
- [Cómo hacer un backup de MySQL con phpMyAdmin](#)



# Artículos de interés

## Videos:

- [Instalar XAMPP 2022 para Windows 10](#)
- [Instalar XAMPP y phpMyAdmin](#)
- [Instalar XAMPP y MySQL Workbench](#)
- [Instalar MySQL y MySQL Workbench](#)
- [Primer encuentro con una Base de Datos MySQL \(MySQL Workbench\)](#)
- [Primer encuentro con una Base de Datos MySQL \(XAMPP y phpMyAdmin\)](#)
- [Cómo instalar XAMPP en Windows y corregir problemas de puertos y permisos](#)

## Sitios e instaladores:

- MySQL: <https://dev.mysql.com/downloads/installer/>
- phpmyadmin: <https://www.phpmyadmin.net/>
- MySQL Workbench: <https://dev.mysql.com/downloads/workbench/>

# No te olvides de dar el presente

# Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.
- Realizar los Ejercicios de repaso.

**Todo en el Aula Virtual.**

**Muchas gracias por tu atención.**

**Nos vemos pronto**