Clase 6: Introducción a base de datos

Sitio: Centro de E-Learning - UTN.BA
Curso: Curso de Backend Developer - Turno Noche
Libro: Clase 6: Introducción a base de datos

Imprimido por: Nilo Crespi
Día: Friday, 23 de January de 2026, 10:17

Tabla de contenidos

1. Introducción

Las bases de datos son fundamentales en el manejo de la información digital, permitiendo almacenar, organizar y recuperar datos de manera eficiente. En esta clase, exploraremos los conceptos esenciales de las bases de datos, con especial énfasis en las bases de datos relacionales. Este modelo organiza la información en tablas conectadas entre sí, lo que facilita su administración y mejora la accesibilidad. Además, analizaremos sus características clave, como la integridad de los datos, la seguridad y la optimización de consultas, aspectos esenciales para cualquier sistema de información moderno.

También profundizaremos en la manipulación de datos mediante SQL, el lenguaje estándar para interactuar con bases de datos relacionales. Aprenderemos a realizar consultas con la sentencia SELECT, insertar y modificar registros, y aplicar filtros utilizando operadores como AND, OR y NOT. Finalmente, abordaremos funciones avanzadas como MIN, MAX, AVG y SUM, que nos permiten realizar cálculos y análisis sobre los datos almacenados. Con estos conocimientos, sentaremos una base sólida para gestionar bases de datos de manera eficiente y efectiva.

2. Bases de datos

Bases de datos

Una base de datos es un sistema para archivar información en una computadora cuyo propósito general es mantener información y hacer que esté disponible cuando se la solicite.

Entonces, la podemos definir como una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los sistemas de información de una empresa o negocio en particular.

Base de datos relacionales

En este tipo de base de datos, la información se almacena en tablas y se relaciona entre las filas. La ventaja del modelo relacional es que los datos se almacenan de tal modo que los usuarios visualizan con mayor facilidad las relaciones entre las tablas. Este enfoque permite a los usuarios obtener información de la base de datos sin necesitar sistemas profesionales de administración de información. Las filas serán cada uno de los registros y las columnas las características (atributos o campos) de cada registro en particular.

Características

- **Independencia lógica y física de los datos.**
- **Redundancia mínima.**
- **Acceso concurrente por parte de múltiples usuarios.**
- **Integridad de los datos.**
- **Consultas complejas optimizadas.**
- **Seguridad de acceso y auditoría.**
- **Respaldo y recuperación.**
- **Acceso a través de lenguajes de programación estándar.**

Características más importantes de los modelos relacionales:

- **Una base de datos relacional se compone de varias tablas o relaciones.**
- **Son atómicos, ya que las entradas de la tabla tienen un solo valor y no se admiten múltiples relaciones. Por lo tanto, la intersección de una fila con una columna tiene un solo valor.**
- **La información puede ser recuperada o almacenada mediante “consultas” o “queries” que ofrecen una amplia flexibilidad y poder para administrar la información.**
- **La información es representada como datos explícitos. No existen apuntadores, ligas o bien flechas visibles para el usuario entre las tablas, sino que se relacionarán mediante códigos.**

Elementos de una base de datos relacional

Estas BD poseen distintos elementos que hay que conocer y que desarrollaremos a continuación:

- **Tablas.**
- **Filas o registros.**
- **Columnas o campos.**
- **Valores.**
- **Campos Llave.**
- **Relaciones.**
- **Esquemas.**

Tablas y columnas

Cada base de datos se compone de una o más tablas que guardan un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queramos guardar en la tabla, cada fila de la tabla conforma un

registro.

Código alumno	Nombre alumno	Apellidos alumno	Dirección	Población	Código postal	Provincia	Teléfono alumno
A0001	Francisco	Lucas Cordero	Avda. Juan Carlos I, 11	Badajoz	06001	Badajoz	606 606606
A0002	José	Gómez Gómez	C/ Almendralejo, 2	Mérida	06800	Badajoz	626 626626
A0003	Pedro	Yuste Luengo	Avda. Antonio Hurtado, 5	Cáceres	08002	Cáceres	616 616616
A0004	Joaquín	Suárez Delgado	Paseo Cánovas, 2	Cáceres	08003	Cáceres	636 636636
A0005	Sandra	Gómez Aranaz	C/Zurbarán, 23	Badajoz	06002	Badajoz	646 646646
A0006	Cristina	Alonso Franco	Paseo Roma, s/n	Mérida	06800	Badajoz	606 000000

Las tablas se componen de dos estructuras:

- **Registros: Corresponde a cada fila que compone la tabla, son las casillas horizontales, en nuestro ejemplo corresponde a un tenista.**
- **Campos: Corresponde al nombre de la columna (vertical). No pueden existir dos campos con el mismo nombre en una misma tabla, deben ser únicos.**

Llaves o claves primarias

Una clave primaria es un campo (o varios) que identifica unívocamente 1 solo registro (fila) en una tabla. Para un valor del campo clave existe solamente 1 registro. Los valores no se repiten ni pueden ser nulos.

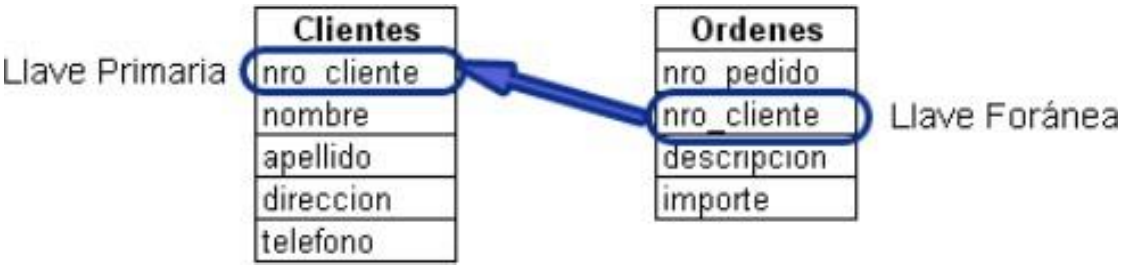
Una tabla sólo puede tener una clave primaria. Cualquier campo (de cualquier tipo) puede ser clave primaria, debe cumplir como requisito, que sus valores no se repitan.

Llaves o claves foráneas

Son campos llave que permiten referenciar unívocamente a un registro que está dentro de otra tabla.

Para poder añadir una fila con un valor de clave foránea específico, debe existir una fila en la tabla relacionada con el mismo valor de clave primaria.

Llave primaria- llave foránea



En el caso de la tabla "Ordenes":

nro_pedido es una Llave Primaria

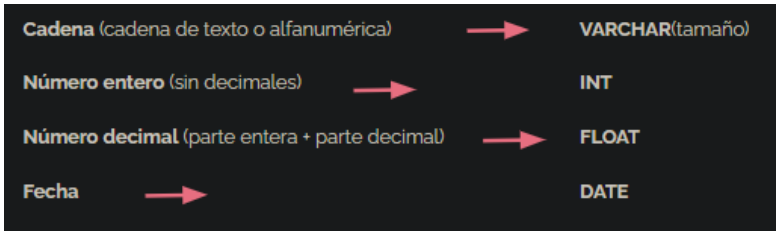
En este caso, el campo llave identifica a cada orden de compra, por lo tanto es para registros que están dentro de la tabla.

nro_cliente es una Llave Foránea

Tipos de campo

Alfanuméricos	Contienen cifras y letras. Presentan una longitud limitada (255 caracteres).
Numéricos	Existen de varios tipos, principalmente, enteros (sin decimales) y reales (con decimales).
Booleanos	Poseen dos formas: Verdadero y falso (Sí o No)
Fechas	Almacenan fechas facilitando posteriormente su explotación. Almacenar fechas de esta forma posibilita ordenar los registros por fechas o calcular los días entre una fecha y otra.
Memos	Son campos alfanuméricos de longitud ilimitada. Presentan el inconveniente de no poder ser indexados (veremos más adelante lo que esto quiere decir).
Autoincrementables	Son campos numéricos enteros que incrementan en una unidad su valor para cada registro incorporado. Su utilidad resulta más que evidente: Servir de identificador ya que resultan exclusivos de un registro.

Tipos de datos SQL



Tipos de datos MySQL

https://www.w3schools.com/mysql/mysql_datatypes.asp

Tipo de datos

Field	Type	Null	Key	Default	Extra
ID_EMPLEADO	int(10) unsigned	NO	PRI		
NOMBRE	varchar(30)	NO			
APELLIDOS	varchar(50)	NO			
F_NACIMIENTO	date	NO			
SEXO	varchar(1)	NO			
CARGO	varchar(30)	NO			
SALARIO	float unsigned	NO			

¿Con qué tipo de dato crearías siguientes los campos?

- **'Hola mundo'**
- **7.36**
- **5896**
- **'Estamos aprendiendo bases de datos'**
- **4 de octubre de 2023 --> 04102023**

3. ¿Qué necesitamos?

¿Qué necesitamos?

Entorno de desarrollo

PhpMyAdmin

- **Es un software de código abierto.**
- **Aporta una interfaz visual basada en el navegador a MySQL, facilitando el trabajo con la base de datos.**
- **Está escrito en PHP. (podemos gestionarla en la web)**
- **La mayoría de alojamiento web incluyen acceso a phpMyAdmin.**

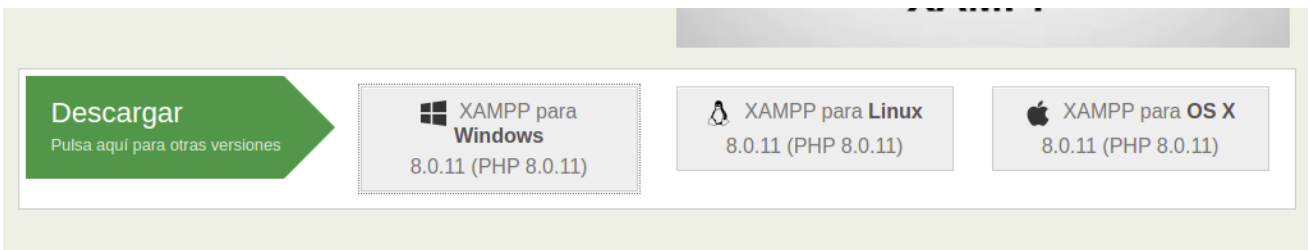
XAMPP

XAMPP es una distribución de Apache completamente gratuita y fácil de instalar que contiene MariaDB, PHP y Perl.

Página oficial para la instalación.

https://www.apachefriends.org/es/index.html

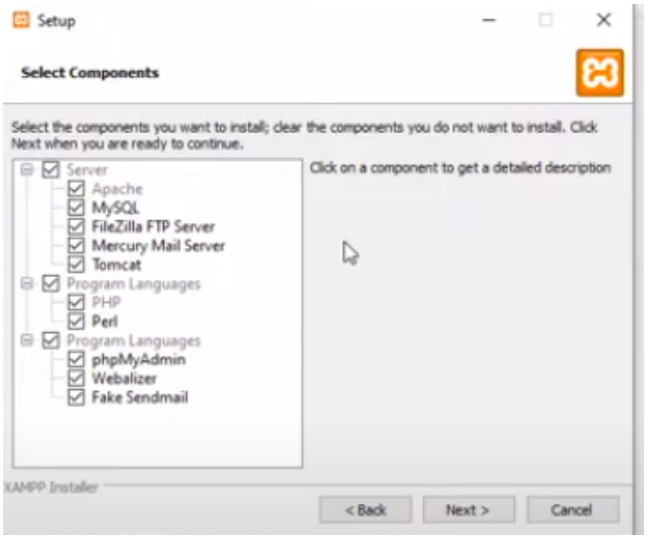
Descarga e Instalación



Instalación

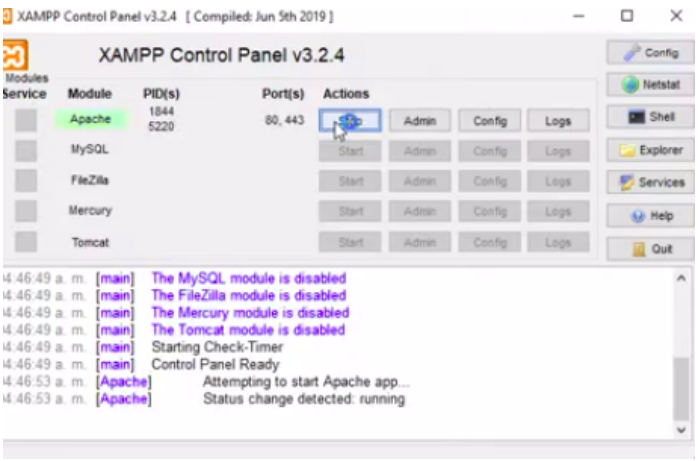
- **Ejecutar como administrador o sudo.**

- **Seleccionar: MySQL, Perl, phpMyAdmin. PHP obligatorio.**



START

Start a APACHE Y MYSQL



Ingresar a phpMyAdmin

http://localhost/phpmyadmin/

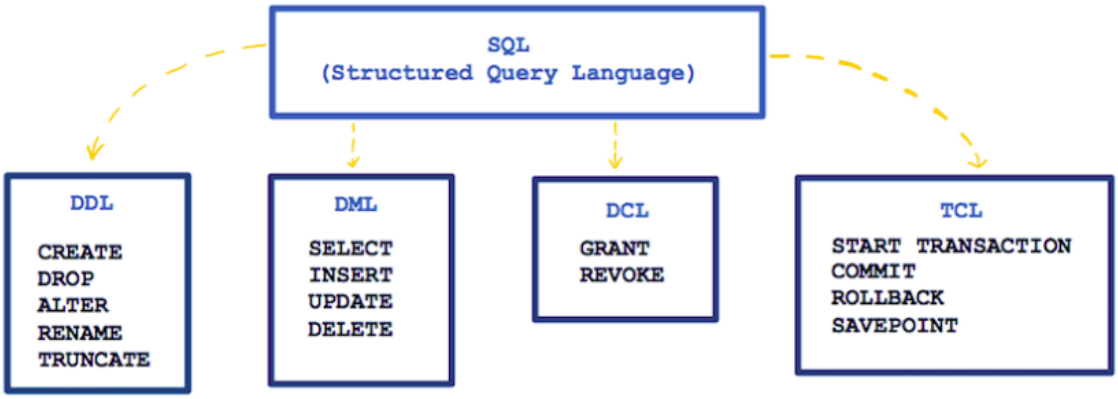
4. SQL

SQL

El DML (Data Manipulation Language) o Lenguaje de Manipulación de Datos es la parte de SQL dedicada a la manipulación de los datos. Las sentencias DML son las siguientes:

- **SELECT: se utiliza para realizar consultas y extraer información de la base de datos.**
- **INSERT: se utiliza para insertar registros en las tablas de la base de datos.**
- **UPDATE: se utiliza para actualizar los registros de una tabla.**
- **DELETE: se utiliza para eliminar registros de una tabla.**

Ahora nos vamos a centrar en el uso de la sentencia SELECT.



SELECT

Nos permite indicar cuáles serán las columnas que tendrá la tabla de resultados de la consulta que estamos realizando. Las opciones que podemos indicar son las siguientes:

- **El nombre de una columna de la tabla sobre la que estamos realizando la consulta. Será una columna de la tabla que aparece en la cláusula FROM.**
- **Una constante que aparecerá en todas las filas de la tabla resultado.**
- **Una expresión que nos permite calcular nuevos valores.**

Ejemplo

Base de datos: Escuela

Tabla: Alumnos

Obtener un dato de todos los alumnos.

Obtener el nombre de todos los alumnos.

```
SELECT nombre
FROM alumno;
```

Obtener varios datos de todos los alumnos

```
SELECT nombre, apellido1, apellido2
```

```
FROM alumno;
```

Obtener datos utilizando una condición where

```
SELECT * FROM `alumnos` WHERE apellido = 'Weasley';
```

Actividad optativa

Creá una base de datos y una tabla con 5 o más registros.

Luego realizá consultas, utilizando select.

Deberán retornar:

- **Todos los datos de una tabla.**
- **Un dato específico de una tabla.**
- **Dos o más datos de una tabla.**
- **Datos con una condición where**

5. Consultas sql

Consultas sql

Insert into

```
INSERT INTO `alumnos`(`nombre`, `apellido`,
`fecha_nacimiento`, `dni`) VALUES
('Luna','Lovegood','2001/10/05','34051155');
```

Update

Lo utilizamos para modificar datos.

```
UPDATE `alumnos` SET
`dni`='40568999' WHERE id_alumno=15;
```

Delete

Lo utilizamos para eliminar registros de una tabla.

```
DELETE FROM `alumnos` WHERE id_alumno = 1;
```

También se pueden eliminar todas las filas de una tabla:

```
DELETE * FROM `alumnos`;
```

¡Para practicar!

- Insertá 5 campos haciendo uso de la sentencia INSERT en la tabla de tu base de datos.
- Actualizá 3 registros específicos empleando UPDATE.
- Eliminá un dato utilizando DELETE.

Ejemplo

	#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
☐	1	id_alumno 🔑	int(11)			No	*Ninguna*		AUTO_INCREMENT	Cambiar Eliminar Más
☐	2	nombre	varchar(50)	utf8mb4_general_ci		No	*Ninguna*			Cambiar Eliminar Más
☐	3	apellido	varchar(50)	utf8mb4_general_ci		No	*Ninguna*			Cambiar Eliminar Más
☐	4	fecha_nacimiento	date			No	*Ninguna*			Cambiar Eliminar Más
☐	5	dni	varchar(8)	utf8mb4_general_ci		No	*Ninguna*			Cambiar Eliminar Más

↑ ☐ Seleccionar todo *Para los elementos que están marcados:* Examinar Cambiar Eliminar Primaria Único Índice

6. Operadores sql

Operadores sql

Operadores AND, OR Y NOT

La WHERE cláusula se puede combinar con los operadores AND, OR y . NOT

Los operadores AND y OR se utilizan para filtrar registros en función de más de una condición:

- **AND -> muestra un registro si todas las condiciones separadas por AND son VERDADERAS.**
- **OR -> operador muestra un registro si alguna de las condiciones separadas por OR es VERDADERA.**
- **NOT -> muestra un registro si la(s) condición(es) NO ES VERDADERA.**

```
SELECT * FROM `escuela` WHERE `apellido`= 'Weasley' AND
`casa`= 'Gryffindor' ;


SELECT * FROM `escuela` WHERE `apellido`= 'Weasley' OR
`casa`= 'Gryffindor' ;



SELECT * FROM `escuela` WHERE NOT `casa`= 'Gryffindor';
```

Order by

ORDER BY se utiliza para clasificar el conjunto de resultados en orden ascendente o descendente.

```
SELECT * FROM alumnos

ORDER BY ciudad DESC;
```

```
SELECT * FROM `escuela` ORDER BY `apellido` DESC;
```

Cláusula LIMIT

- **Se utiliza para especificar el número de registros a devolver.**
- **Es útil en tablas grandes con miles de registros. Devolver una gran cantidad de registros puede afectar el rendimiento.**

```
SELECT * FROM escuela
LIMIT 3;
```

LIMIT con WHERE

```
SELECT * FROM `escuela`
WHERE `casa`='Gryffindor' LIMIT 3;
```

Funciones MySQL MIN() y MAX()

- **MIN() devuelve el valor más pequeño de la columna seleccionada.**
- **MAX() devuelve el valor más grande de la columna seleccionada.**

```
SELECT MIN(edad) FROM alumnos;

SELECT MIN(edad) AS EdadMinima FROM
alumnos;
```

Funciones MySQL MIN() y MAX() - Ejemplos

1. **Agregar Nombre y Apellido al resultado:**

   SELECT nombre, apellido, MIN(edad) AS edadMinima
   FROM alumnos;

2. **Filtrar por Edad Mínima:**

   SELECT nombre, apellido, edad
   FROM alumnos
   WHERE edad = (SELECT MIN(edad) FROM alumnos);

3. **Mostrar varios registros con la misma Edad Mínima:**

   SELECT nombre, apellido, edad
   FROM alumnos
   WHERE edad = (SELECT MIN(edad) FROM alumnos);

Funciones COUNT(), AVG() y SUM()

- **La función COUNT() cuenta el número de filas que cumplen ciertas condiciones**
- **La función AVG() calcula el promedio de los valores de una columna**
- **La función SUM() calcula la suma de los valores de una columna en una tabla.**

```
SELECT COUNT(DNI) FROM escuela;
SELECT AVG(edad) FROM escuela;
SELECT SUM(edad) FROM escuela;
```

Funciones COUNT(), AVG() y SUM() - Ejemplos

1. **Consulta con Filtro por Edad:**

   SELECT COUNT(DNI)
   FROM escuela
   WHERE edad > 18;

2. **Consulta con Agrupamiento por Género:**

```
SELECT genero, AVG(edad)
FROM escuela
GROUP BY genero;
```

3. **Consulta con Filtro por Género y Rango de Edad:**

```
SELECT SUM(edad)
FROM escuela
WHERE genero = 'masculino' AND edad > 20;
```

4. **Consulta con Subconsulta:**

```
SELECT COUNT(DNI)
FROM escuela
WHERE edad > (SELECT AVG(edad) FROM escuela);
```

Operador IN - NOT IN

IN permite especificar múltiples valores en una cláusula WHERE .

Es una forma abreviada de múltiples condiciones OR.

```
SELECT * FROM `escuela` WHERE `ciudad` IN
('Buenos Aires', 'Santa Fe');

SELECT * FROM `escuela` WHERE `ciudad` NOT IN
('Buenos Aires', 'Santa Fe');
```

7. Actividad

Actividad

Creá y ejecutá una query que incluya:

- Operadores (AND, OR NOT)
- Límite
- Ordenen según un campo específico (ASC - DESC)

Actividades de práctica

Parte 1 - Creación de la base de datos:

1. Definir la Estructura de la Tabla: Crear la tabla "escuela" utilizando las siguientes columnas: DNI(clave primaria), nombre, apellido, edad, genero y id_curso_inscrito.

2. Poblar de Datos: agregar al menos 5 registros ficticios en la tabla "escuela" con valores para las columnas definidas.

Parte 2 - Realización de Consultas:

Realizar las siguientes consultas:

1. Contar la cantidad total de estudiantes en la base de datos.
2. Calcular el promedio de edad de todos los estudiantes.
3. Sumar todas las edades de los estudiantes.
4. Contar la cantidad de estudiantes masculinos mayores de 20 años.
5. Calcular el promedio de edad para cada género (masculino y femenino).
6. Contar la cantidad de estudiantes cuya edad es superior al promedio de edad.