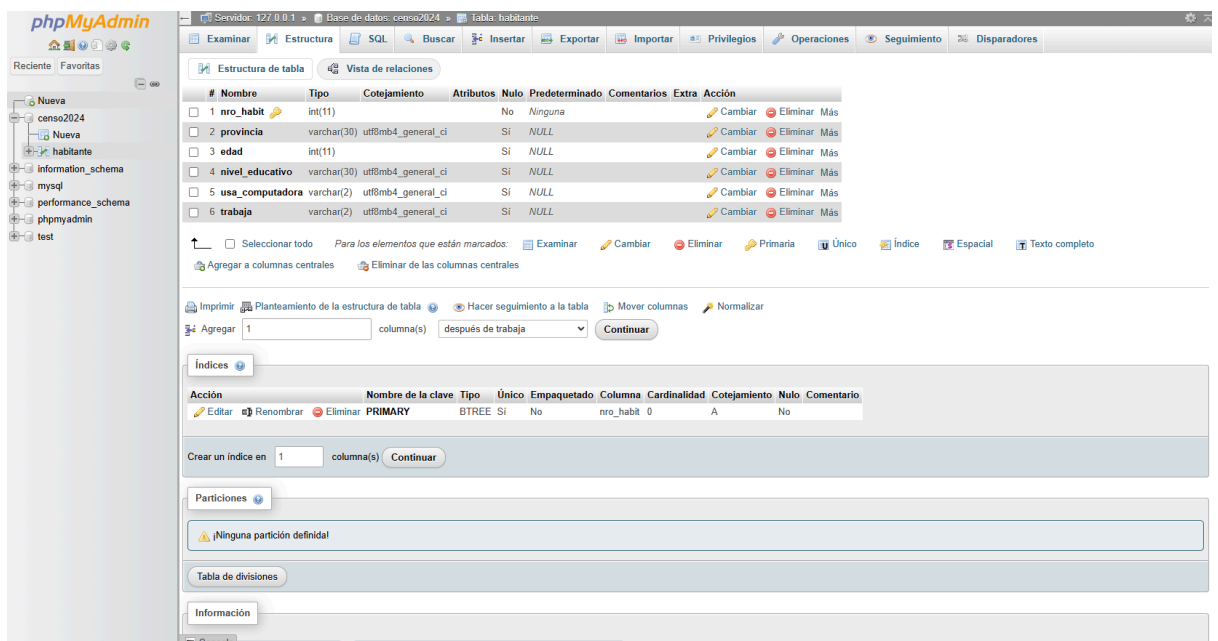


MySQL

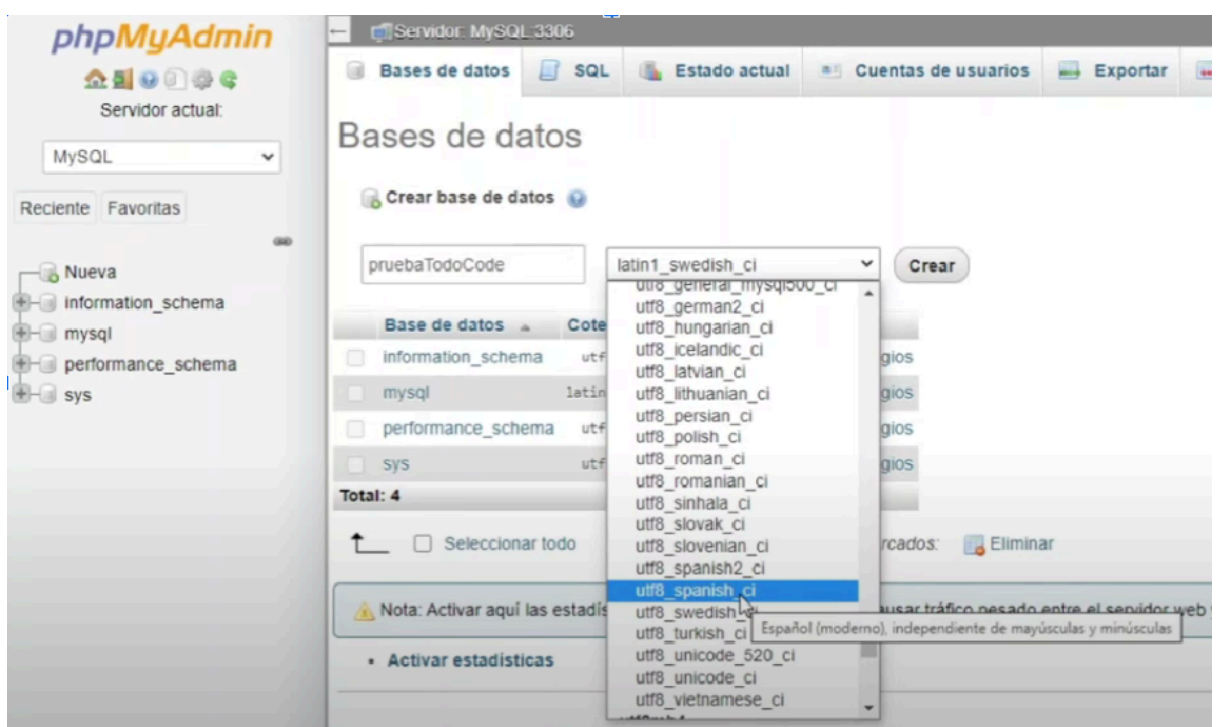
1. Acceder a phpMyAdmin

Abrir <http://localhost/phpmyadmin/> en el navegador. phpMyAdmin es la interfaz web para gestionar bases MySQL/MariaDB en local que usa Laboratorio de programación.



2. Crear una base de datos

Al crear una base de datos podés elegir un nombre descriptivo. En la sección **Collation** seleccioná una codificación que soporte caracteres en español: **utf8mb4_spanish_ci** o **utf8mb4_general_ci** (utf8mb4 es preferible para incluir emojis y todos los caracteres Unicode). Luego presioná **Crear**.



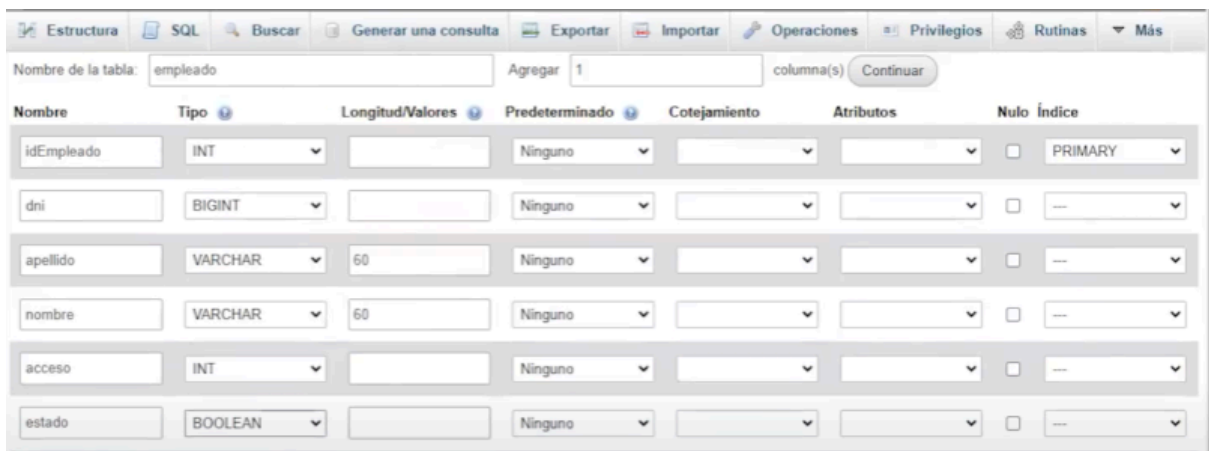
3. Crear tablas y definir columnas

Al crear una tabla, ingresás nombre de la tabla y columnas.



Para cada columna hay que definir:

- **Nombre** (ej. idEmpleado)
- **Tipo** (INT, VARCHAR(n), DATE, DATETIME, TEXT, FLOAT, DECIMAL, etc.)
- **Longitud/valores** (para VARCHAR → longitud, por ej. VARCHAR(100))
- **Atributos**: PRIMARY KEY (clave primaria), UNIQUE, NOT NULL, AUTO_INCREMENT (A.I.), DEFAULT (valor por defecto).



Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice
idEmpleado	INT		Ninguno			<input type="checkbox"/>	PRIMARY
dni	BIGINT		Ninguno			<input type="checkbox"/>	---
apellido	VARCHAR	60	Ninguno			<input type="checkbox"/>	---
nombre	VARCHAR	60	Ninguno			<input type="checkbox"/>	---
acceso	INT		Ninguno			<input type="checkbox"/>	---
estado	BOOLEAN		Ninguno			<input type="checkbox"/>	---

Una vez cargada le damos a guardar. Despues se puede modificar en *Estructura*.

Servidor: 127.0.0.1 > Base de datos: obrador2023 > Tabla: empleado

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Disparadores

Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	idEmpleado	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
<input checked="" type="checkbox"/> 2	dni	bigint(20)			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 3	apellido	varchar(60) utf8mb4_general_ci			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 4	nombre	varchar(60) utf8mb4_general_ci			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 5	acceso	int(11)			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 6	estado	tinyint(1)			No	Ninguna			Cambiar Eliminar Más

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

Espacial Texto completo

En código sería algo así:

```
CREATE TABLE empleado (
  idEmpleado INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL,
  dni VARCHAR(20) UNIQUE,
  fecha_ingreso DATE,
  activo TINYINT(1) DEFAULT 1
)
```

4. Relaciones y claves foráneas

En phpMyAdmin → *Estructura* → *Relaciones* o *Vista de relaciones* podés crear **FOREIGN KEY** (clave foránea). Primero seleccionás la columna en la tabla hija, después la base de datos, luego la tabla padre y la columna referenciada.

Servidor: 127.0.0.1 > Base de datos: obrador2023 > Tabla: empleado

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Disparadores

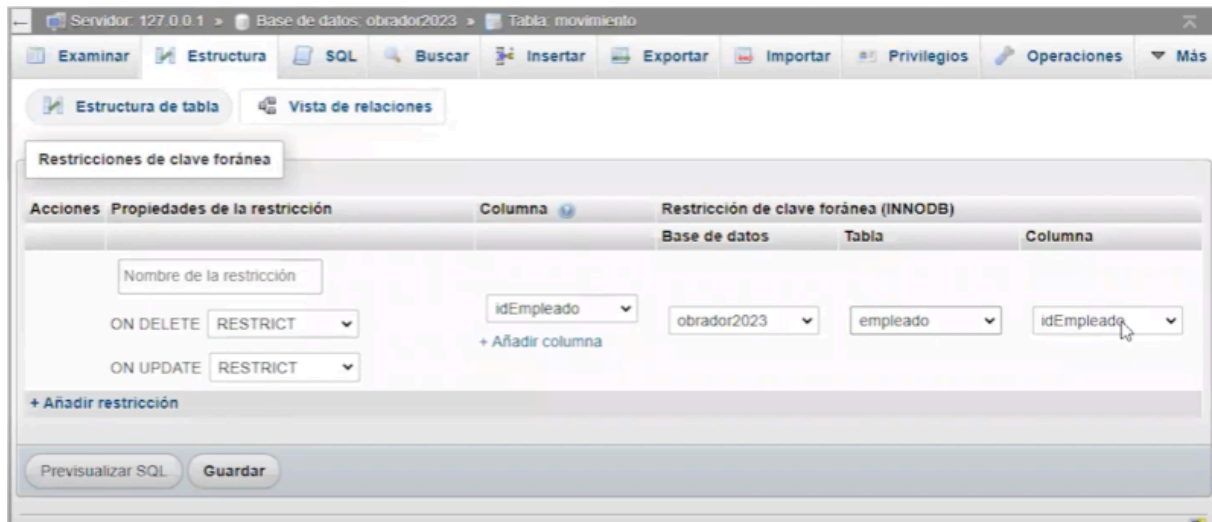
Estructura de tabla Vista de relaciones

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	idEmpleado	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Más
<input checked="" type="checkbox"/> 2	dni	bigint(20)			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 3	apellido	varchar(60) utf8mb4_general_ci			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 4	nombre	varchar(60) utf8mb4_general_ci			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 5	acceso	int(11)			No	Ninguna			Cambiar Eliminar Más
<input type="checkbox"/> 6	estado	tinyint(1)			No	Ninguna			Cambiar Eliminar Más

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

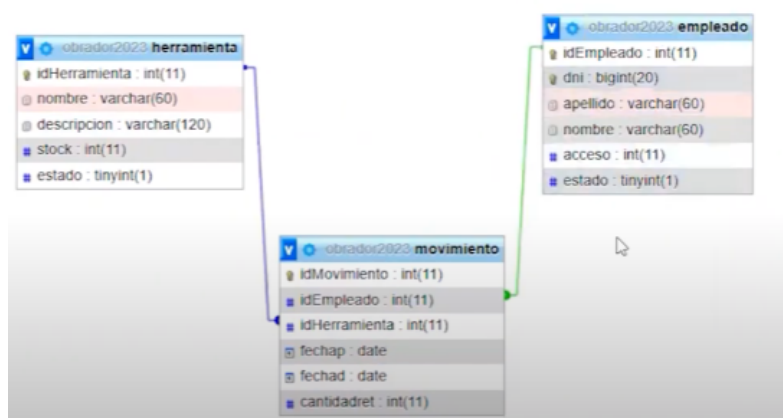
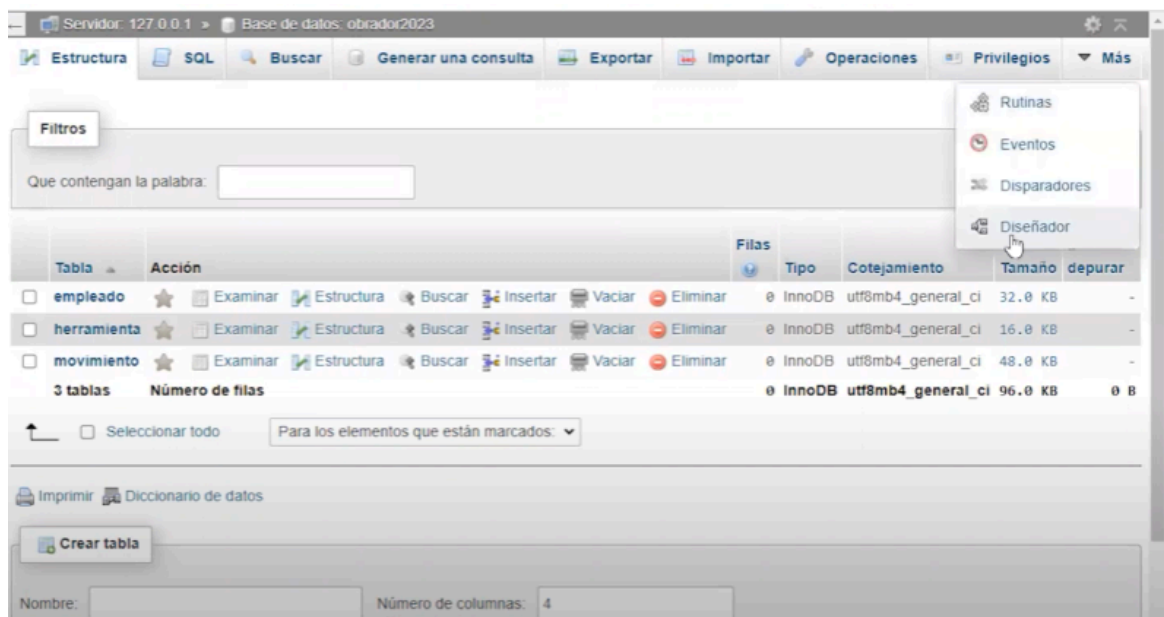
Espacial Texto completo

Ejemplo: la tabla `herramienta` tiene `idEmpleado` que referencia `empleado.idEmpleado`:



5. Diseño visual de las tablas y guardar layout

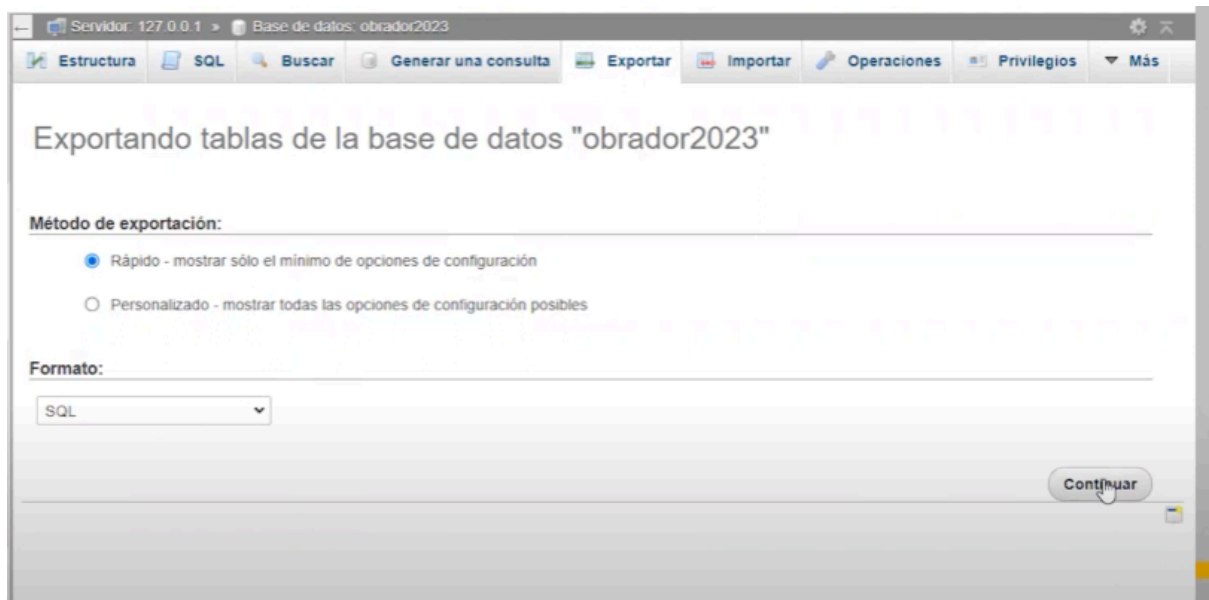
En phpMyAdmin existe la *vista de diseñador* (Designer) para ver gráficamente tablas y relaciones. Podés organizar las tablas arrastrando y guardar el diseño para revisarlo después.



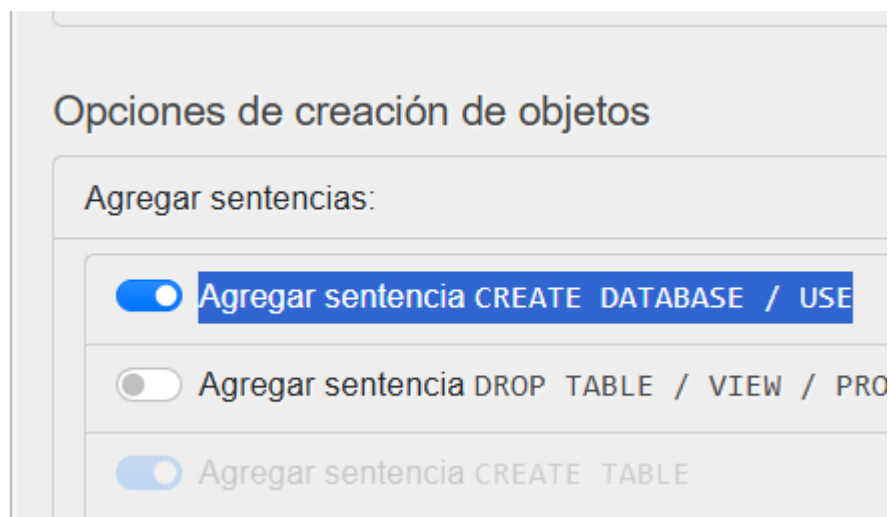
6. Exportar e importar bases de datos

6.1. Exportar

En la pestaña *Exportar* podés generar un archivo `.sql` que contiene la definición y los datos. Ese `.sql` sirve para compartir o respaldar la base.

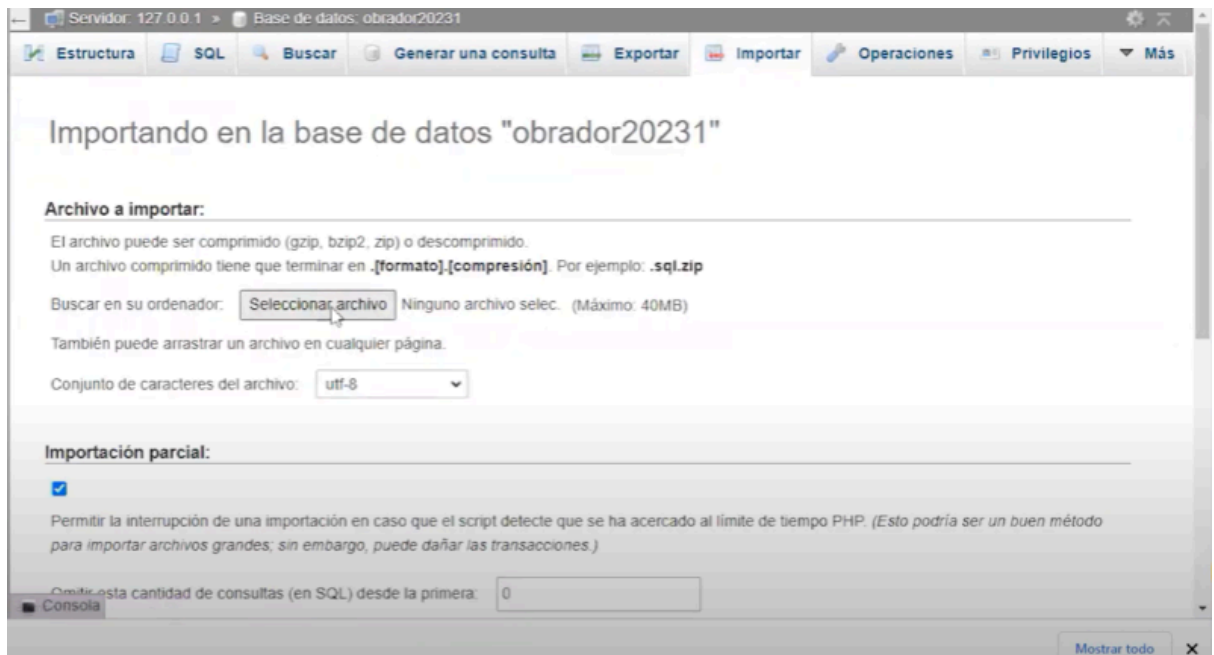


Para exportar de forma personalizada (según lo que piden los profes) siempre hay que activar esta opción:



6.2. Importar

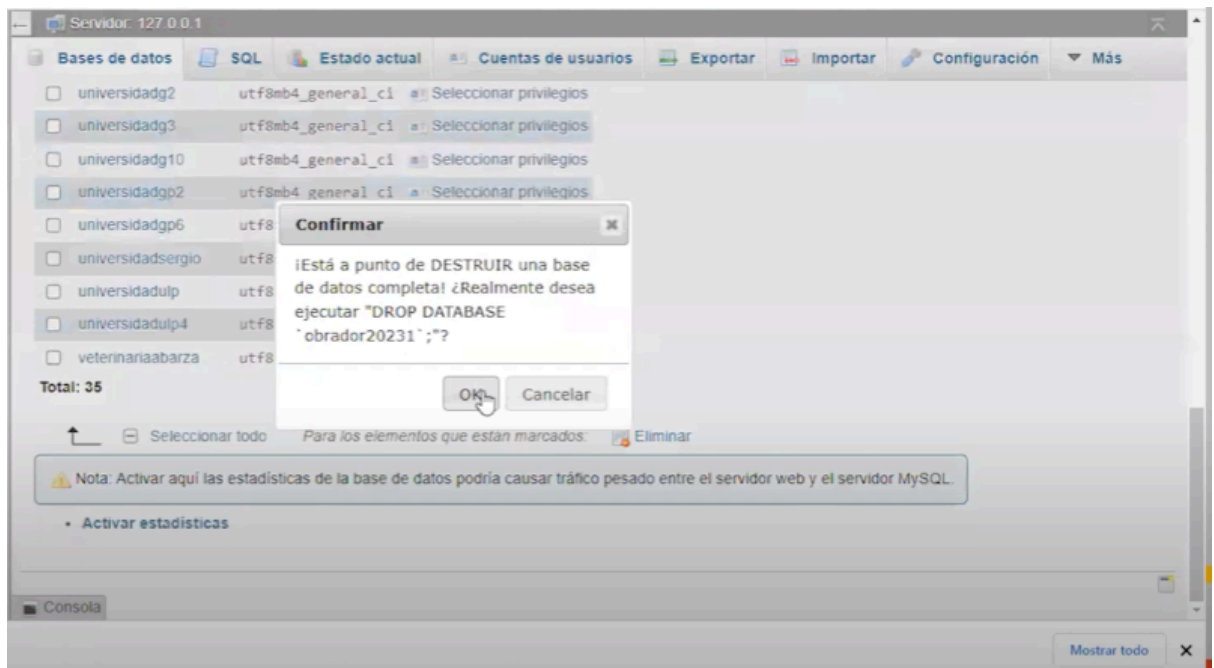
Para importar en otra máquina: crear una base vacía (el nombre puede ser distinto a la base de datos original), luego *Importar* el archivo `.sql` y ejecutar.



7. Eliminar bases de datos

Ruta típica: Servidor -> Bases de datos -> seleccionar base -> Eliminar

En caso de querer eliminar una base de datos nos vamos arriba donde dice “Servidor 127...”, después a Base de datos, y de ahí seleccionamos la base de datos que queremos eliminar

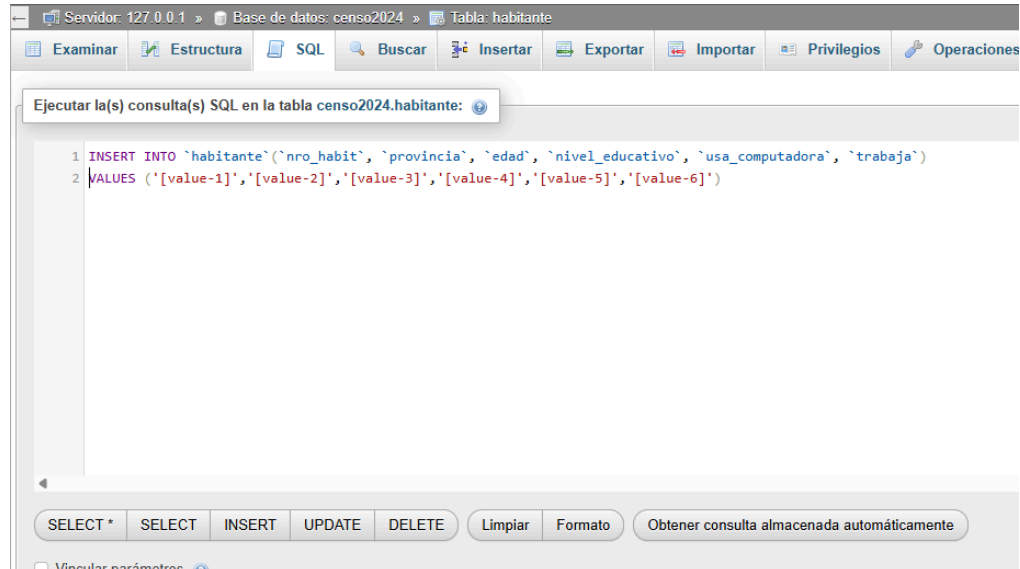


Si querés “desactivar” registros sin perder historial, preferí un campo **estado** o **activo** en lugar de borrar filas.

8. Comandos SQL básicos

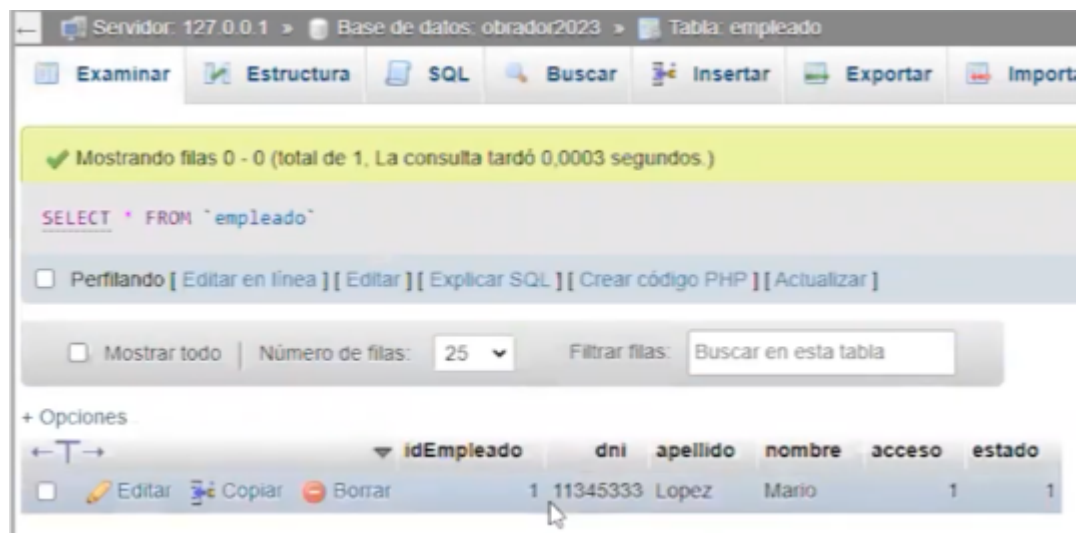
8.1. INSERT

Insertar filas. En phpMyAdmin podés usar la pestaña *Insertar* o ejecutar SQL:



`INSERT INTO empleado (nombre,dni,fecha_ingreso) VALUES ('Leonel', '12345678', '2025-09-22');`

Una vez insertados los datos, en *Examinar* podemos ver la tupla agregada.



Si hay `AUTO_INCREMENT` en `idEmpleado`, no es necesario proveerlo al insertar, ya que SQL lo hará de forma automática de 1 a n.



También podemos ingresar tuplas desde la sección de “Insertar”

Formulario de inserción para la tabla 'herramienta'.

Columna	Tipo	Función	Nulo	Valor
idHerramienta	int(11)			
nombre	varchar(60)			Taladro Bosch
descripcion	varchar(120)			
stock	int(11)			
estado	tinyint(1)			

Continuar

8.2. SELECT

Sintaxis básica: (SELECT atributos FROM tabla)

SELECT columna1, columna2 FROM tabla;

-- O

SELECT * FROM tabla; -- proyecta todos los campos

```
1 listar id, nombre, stock de todas las herramientas.  
2 SELECT idHerramienta, nombre, stock  
3 FROM herramienta
```

8.3. WHERE

Where funciona como un filtro o condicion:

SELECT *
FROM empleado WHERE activo > 1;

SELECT *
FROM herramienta
WHERE stock >= 5;

WHERE stock <= 10;

8.4. **LIKE**

Búsqueda por patrón en string usando % como comodín:

- **LIKE** 'pala%' → empieza con "pala"
- **LIKE** '%pala' → termina en "pala"
- **LIKE** '%pala%' → contiene "pala" en cualquier posición

SELECT *
FROM herramienta **LIKE** 'pala'

8.5. **BETWEEN**

Determina un rango (between : entre):

SELECT *
FROM movimiento
WHERE fecha **BETWEEN** '2023-07-01' **AND** '2023-07-31';

8.6. **ORDER BY**

Determina un ordenamiento:

ASC (Ascendente A - Z) esta viene por defecto.
DESC (Descendente Z - A).

SELECT *
FROM empleado
ORDER BY nombre **ASC**;

SELECT *
FROM empleado
ORDER BY nombre **DESC**;

8.7. **IN / NOT IN**

IN y NOT IN podrian ser como una especie de filtros:

SELECT *
FROM empleado
WHERE acceso **IN** (2,3); Filtra para mostrar entre 2 y 3

SELECT *
FROM empleado
WHERE acceso **NOT IN** (2,3); Me muestra todos aquellos que no tienen 2 o3

8.8. COUNT, SUM, AVG

Funciones de agregación:

Para contar entre las columnas se usa SELECT al principio y luego COUNT. Se pasa por parametro () la columna a la que quieres contar.

El * representa que vas a considerar todas las columnas, y el AS se usa para renombrar la columna temporal de COUNT, ya que sino crea una genérica para el total con su nombre count.

```
SELECT COUNT(*) AS total
FROM empleado;
WHERE acceso > 1
```

```
SELECT herramienta SUM(cantidad) AS total_herramientas
FROM movimiento;
```

AVG saca el promedio

```
SELECT AVG(precio) AS promedio_precio
FROM producto;
```

8.9. GROUP BY

En el caso de que queramos agrupar se usa GROUP BY, con el atributo que queramos contar.

```
SELECT COUNT(*) AS cantidad
FROM empleado
GROUP BY acceso;
```

En el caso anterior solo vemos la cantidad, pero no sabemos a qué acceso pertenecen, para eso podemos poner al lado del SELECT el atributo o columna, que tiene que coincidir con el GROUP BY, no podríamos poner idEmpleado por ejemplo.

```
SELECT acceso, COUNT(*) AS cantidad
FROM empleado
GROUP BY acceso;
```

8.10. UPDATE

Modifica o actualiza los registros:

```
UPDATE empleado SET activo = 0 WHERE idEmpleado = 123;
```

Siempre usar **WHERE** para no afectar todas las filas por accidente, en este caso se usa idEmpleado para hacerlo de forma específica.

8.11. DELETE

Borra registros:

```
DELETE FROM movimiento WHERE fecha = '2023-07-07';
```

Sin **WHERE** se borran todas las filas de la tabla. **Recomendación:** en muchas bases se usa un campo **estado** o **activo** para “dar de baja” lógicamente sin eliminar historial, ya que eliminar un empleado en este caso implica eliminar todo registro de este.

8.12. JOIN

Une tablas para obtener información combinada:

```
SELECT dni, apellido, nombre  
FROM empleado JOIN movimiento ON (empleado.idEmpleado =  
movimiento.idEmpleado)  
WHERE fechap BETWEEN 2023-01-21 AND 2023-02-22
```