

BBDD

SQL + Relaciones

Como *trabajar con varias tablas* en BD ?

Vamos a modelar el caso de PRODUCTOS y CATEGORÍAS

- Cada producto pertenece a una única categoría.
- Una categoría puede estar asociada a más de un producto.

Requerimientos:

- Todos los datos los tenemos en una BD y queremos aplicar servicios/consultas.

Repaso:

- ¿Por qué vamos a usar dos tablas para almacenar estos datos?

Diseño de las tablas

db_productos productos	
🔑	id_producto : int(11)
📄	nombre : varchar(100)
#	precio : decimal(10,0)
#	id_categoria : int(11)

db_productos categorias	
🔑	id_categoria : int(11)
📄	nombre : varchar(100)

Consistencia de los datos

- ¿Qué pasa si borramos una categoría vinculada a un producto?
- O si agregamos un producto vinculado a una categoría que no existe.

Datos inconsistentes

- Los RDBMS (Bases de datos) tienen mecanismos para evitar estas inconsistencias.
 - Relaciones entre tablas.

Como *modelar* Relaciones en BBDD ?

Se utiliza uno o más **atributos** de una tabla referencia a la clave /s de otra.

Esa relación se representa mediante **Foreign KEY** (FK).

- Los *tipos* entre atributos de las tablas deben ser iguales
- Las operaciones de *eliminar/actualizar* datos se restringen (no puedo borrar si existe una relación)

IMPORTANTE

No es obligatorio usar FK para relacionar las tablas, sirve para restringir cuando borramos datos

Usando SQL: Sintaxis Foreign key

```
CREATE TABLE <table1>(
    atributte1 TYPE NOT NULL,
    ....
    PRIMARY KEY (OrderID),
);
```

```
CREATE TABLE <table2>(
    atributte1 TYPE NOT NULL,
    attributeFK TYPE NOT NULL,
    ....
    FOREIGN KEY (attributeFK ) REFERENCES table1(atributte1)
);
```

Diseño de las tablas

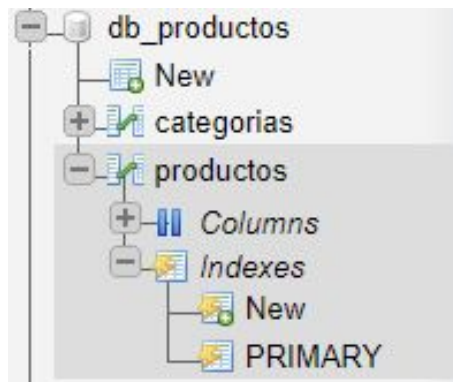
db_productos productos
id_producto : int(11)
nombre : varchar(100)
precio : decimal(10,0)
id_categoria : int(11)

db_productos categorias
id_categoria : int(11)
nombre : varchar(100)



Crear la relación desde PHP Admin

- Primero necesitamos asegurarnos que el campo `id_categoria` sea clave de la tabla `CATEGORIAS`.
- Luego crear un índice sobre el campo `id_categoria` de la tabla `PRODUCTOS` (restricción MySQL)

A screenshot of the 'Add index' dialog box in PHPMyAdmin. The dialog has a title bar 'Add index' with a close button. It contains the following fields and options:

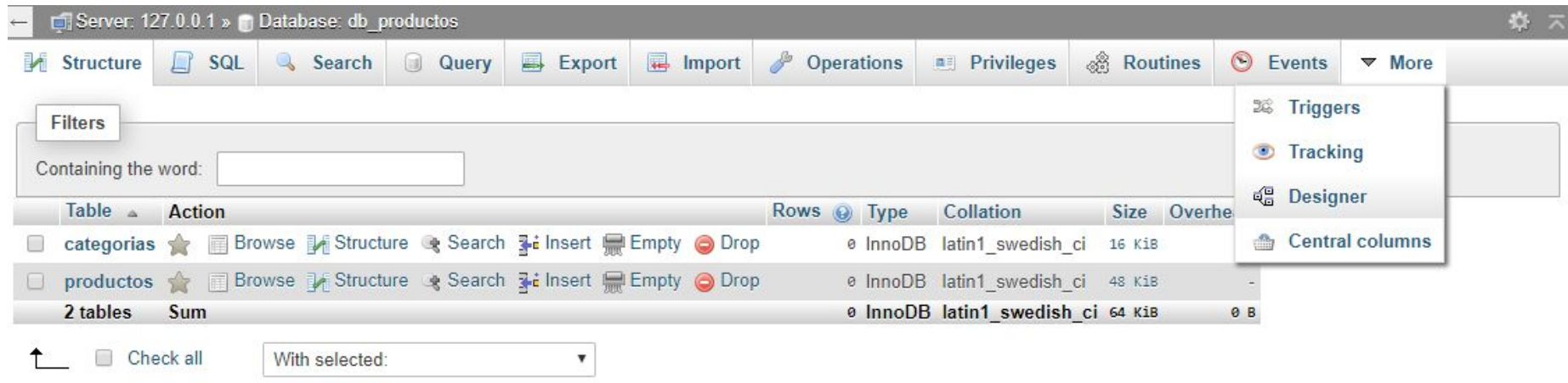
- Index name:** A text input field containing 'FK_id_categoria'.
- Index choice:** A dropdown menu set to 'INDEX'.
- + Advanced Options:** A link to expand advanced options.
- Table:** A dropdown menu showing the selected table.
- Column:** A table with two columns: 'Column' and 'Size'.

Column	Size
id_categoria [int(11)]	
-- Ignore --	
- Add 1 column(s) to index:** A button to add the selected column to the index.

At the bottom of the dialog are three buttons: 'Go', 'Preview SQL', and 'Cancel'.

Crear la relación desde PHP Admin

- Seleccionando la base de datos, elijo la vista diseñador.



The screenshot shows the phpMyAdmin interface for a database named 'db_productos' on a server at 127.0.0.1. The top navigation bar includes tabs for Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, Events, and a 'More' dropdown menu. The 'More' menu is currently open, displaying options: Triggers, Tracking, Designer (highlighted), and Central columns.

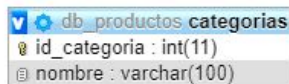
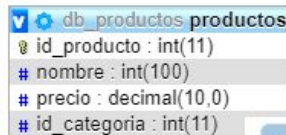
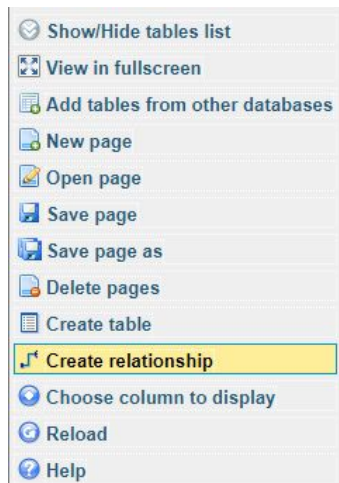
Below the navigation bar, there is a 'Filters' section with a search box labeled 'Containing the word:'. Below this is a table listing the database's structure:

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> categorias	★ Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	16 KiB	
<input type="checkbox"/> productos	★ Browse Structure Search Insert Empty Drop	0	InnoDB	latin1_swedish_ci	48 KiB	-
2 tables	Sum	0	InnoDB	latin1_swedish_ci	64 KiB	0 B

At the bottom, there is a 'Check all' checkbox and a dropdown menu labeled 'With selected:'.

Crear la relación desde PHP Admin

- Crear relación
- Elegir la clave referenciada CATEGORIAS::id_categoria.
- Luego la clave foránea PRODUCTOS::id_categoria.
- onDelete/onUpdate -> RESTRICT (No deja borrar)




Prueba

id_categoria	nombre
1	Indumentaria
2	Computación

id_producto	nombre	precio	id_categoria
1	Zapatillas de tenis	4500	1
2	Monitor 32	14500	2
3	Camiseta térmica	900	1

- Probamos borrar una categoría asociada a un producto.


```
DELETE FROM categorias WHERE id_categoria = 1
```

MySQL said: 

```
#1451 - Cannot delete or update a parent row: a foreign key constraint fails (`db_productos`.`productos`, CONSTRAINT `productos_ibfk_1` FOREIGN KEY (`id_categoria`) REFERENCES `categorias` (`id_categoria`))
```

- Probamos dar de alta un nuevo producto a una categoría no existente

```
INSERT INTO productos(nombre, precio, id_categoria) VALUES ('Sillón 2 cuerpos', 3250, 3)
```

MySQL said: 

```
#1452 - Cannot add or update a child row: a foreign key constraint fails (`db_productos`.`productos`, CONSTRAINT `productos_ibfk_1` FOREIGN KEY (`id_categoria`) REFERENCES `categorias` (`id_categoria`))
```

Sitio web para productos

- Ahora vamos a suponer un sitio que muestra los productos y las categorías.

```
<?php
require_once('ddbb.php');
$productos = getProductos();
$categorias = getCategorias();
?>
<!DOCTYPE html>
<html lang="en">
<head>
...
</head>
<body>
...
```

```
<h1>Categorías</h1>
<ul>
    <?php foreach ($categorias as $categoria) { ?>
        <li><?php echo $categoria->nombre; ?></li>
    <?php } ?>
</ul>

<h1>Productos</h1>
<ul>
    <?php foreach ($productos as $producto) { ?>
        <li><?php echo $producto->nombre; ?></li>
    <?php } ?>
</ul>
</body>
</html>
```

Sitio web para productos

- ¿Cómo podemos mostrar cada producto con el nombre de la categoría a la cual pertenece?
 - Traemos los productos existentes.
 - Iteramos por cada uno de ellos y obtenemos la categoría.
 - Construimos la colección resultante con los datos de PRODUCTOS y CATEGORÍAS.

Sitio web para productos

```
function getProductosConCategoria() {  
    //Primero traemos todos los productos.  
    $productos = getProductos();  
    //Creamos la consulta para obtener una categoria  
    $db = connect();  
    $query = $db->prepare("SELECT * FROM categorias WHERE id_categoria=?");  
    //Vamos a construir el arreglo resultante  
    $productosConDetalle = array();  
    foreach($productos as $producto) {  
        $p['producto'] = $producto->nombre;  
        $p['precio'] = $producto->precio;  
  
        $query->execute(array($producto->id_categoria));  
        $categoria = $query->fetch(PDO::FETCH_OBJ);  
        $p['categoria'] = $categoria->nombre;  
        array_push($productosConDetalle, $p);  
    }  
  
    return $productosConDetalle;  
}
```

Sitio web para productos

- ¿Parece correcto consultar la base de datos una vez por cada producto para traer una categoría?
- ¿Cuántas veces pedimos por la categoría con id_categoria = 1 en este ejemplo?
- ¿Podríamos traer esa misma información en una única consulta a la base?
 - SQL: Join

SQL: Join

- Permite combinar registros de distintas tablas mediante columnas de las tablas.
- El resultado de la consulta puede retornar información de todas las tablas involucradas.
- Distintos tipos de JOIN (que se verán en otra materia)

```
SELECT t1.column1, t1.column2, t2.column1 FROM t1 JOIN t2  
ON t1.column1 = t2.column1
```


SQL: Join

SELECT * FROM productos JOIN categorias ON
productos.id_categoria = categorias.id_categoria

id_producto	nombre	precio	id_categoria	id_categoria	nombre
1	Zapatillas de tenis	4500	1	1	Indumentaria
2	Monitor 32	14500	2	2	Computación
3	Camiseta térmica	900	1	1	Indumentaria

- Hay que tener cuidado con las columnas con igual nombre.

SQL: Join

SELECT productos.*, categorias.nombre as categoria FROM
productos JOIN categorias ON productos.id_categoria =
categorias.id_categoria

id_producto	nombre	precio	id_categoria	categoria
1	Zapatillas de tenis	4500	1	Indumentaria
2	Monitor 32	14500	2	Computación
3	Camiseta térmica	900	1	Indumentaria

Sitio web para productos

```
function getProductosConCategoria() {  
    $db = connect();  
    $query = $db->prepare("SELECT productos.*, categorias.nombre as categoria FROM productos JOIN  
categorias ON productos.id_categoria = categorias.id_categoria");  
    $query->execute();  
  
    return $query->fetchAll(PDO::FETCH_OBJ);  
}
```

- Una única consulta a la base para traer todos los resultados.
- Se puede hacer el FETCH_OBJ para procesar los datos.