

GESTOR DE BBDD RELACIONAL Y MULTIHILO Y MULTIUSUARIO

Una base de datos relacional es una base de datos en que varias tablas están relacionadas entre si

Por ejemplo una tabla de clientes puede estar relacionada con una tabla de clientes .

Al relacionarla nos da el pedido que ha hecho cada cliente

Multihilo significa que la base de datos soporta varios procesos o hilos de ejecución ,a la vez . Esto quiere decir que podemos hacer varios pedidos o consultas de datos a la base de datos, de forma simultánea.

Multiusuario que permite que varios usuarios interactúen a la vez, o bien pidiendo datos o bien modificándolos.

Cada uno de estos usuarios tendrá su usuario su contraseña y su nivel de acceso que puede ser diferente en cada caso.

Una característica muy importante de mysql, es que es libre, esto implica, entre otras cosas, que:

- no tienes que pagar por este gestor de bases de datos.
- Permite descargar el código fuente con el que está creado este gestor y modificarlo a tu gusto

¿cómo conectarnos a el gestor de bases de datos?

4. Tener claro que vamos a encontrarnos con dos gestores de datos de MySql.
 - a. Gestor de bases de datos en local.

- b. Gestor de bases de datos en remoto

Al instalar el WAMP (O MAMP , XAMP....) estamos instalando en local el gestor de datos MySQL, que se trata de un gestor de base de datos de prueba, porque para poder gestionar nuestra base de datos en nuestra web definitiva, en WWW debe ser en remoto, el que nos va a proporcionar nuestro ISP (proveedor de Servicios de Internet), hay muchos.

Así nosotros probaremos nuestra base de datos en local, y luego lo subiremos a remoto, y cuando la subamos deberemos cambiar una serie de datos, porque en local no es lo mismo que en remoto.

5. haremos nuestras primeras pruebas en local.

Podemos hacerlo de dos formas:

- a. mediante nuestra consola de comandos
 - b. mediante phpmyadmin que es una herramienta muy buena que incluye el WAMP, XAMP.....
es una interfaz gráfica que facilita todas las actividades que se pueden hacer desde una base de datos MySQL, de una base de :
 - crear base de datos
 - crear tablas
 - modificar tablas
 - insertar registros
 - consultar registros
 - etc...

3. Pasos para instalar el phpMyAdmin desde el paquete WAMP
 - a. Comprobamos que los servicios del WAMP están arrancados
 - b. Pulsamos con el botón derecho del ratón y vemos que se despliega un menú
 - c. Pulsamos phpMyAdmin y entramos en la interfaz gráfica.
 - d. Vemos que hay una serie de bases de datos que ya están instaladas. Son bases de datos que se instalan con el paquete y que no podéis eliminar. Porque contienen información para que todo el gestor de datos MySQL funcione de forma correcta
 - e. A continuación de estas bases de datos predeterminadas, aparece un botón llamado bases de datos. Pulsamos en él y se cargan las bases de datos que tenemos por defecto. Son 4
 4. Pasos para instalarlo desde el navegador, para el caso de que hayas instalado phpadmin de forma independiente
 - a. Abrimos un navegador
 - b. En la barra de dirección ponemos la dirección IP de tu ordenador
 - c. Para saber la dirección IP de tu ordenador, ejecutas la consola de comandos poniendo CMD en el buscador
 - d. Dentro ejecutas la instrucción ipconfig y te sale al dirección IP interna que suele ser algo así como 192.168.56.1
 - e. Si introduces esta dirección en el navegador podrás acceder a un menú de tu paquete en local, que también te permite entrar en PHPMYADMIN
 - f. O poner directamente en la barra de navegador
192.168.56.1/phpmyadmin
 - g. Desde el WAMP también puedo acceder a la consola
MySQL>CONSOLA. Pide un password que hay que dejar en blanco
-
- Vamos a crear una primera base de datos desde phpmyadmin
 - e.1. Entramos en el botón bases de datos de la barra de herramientas.
 - e.2. Luego pulsamos en el botón crear base de datos
 - e.3. Indicamos un nombre: curso php
 - e.4. Crear>nos sale un cartel en el que pone: "la nueva base de datos ha sido creada" y se actualiza el panel de la izda con todas las bases de datos y la del contenido del botón base de datos.
 - e.5. Eliminamos la base de datos: pulsamos en el checkbox y eliminamos (Ojo las bases de datos que vienen por defecto no se pueden eliminar, el checkbox viene desactivado). pregunta si de verdad desea ejecutar DROP DATABASE (instrucción SQL que hace la tarea de eliminar la base de datos) SQL Lenguaje de consulta de datos estructurado, llamado SQL
 - Vamos a crear ahora una base de datos desde consola
 1. Abrimos consola desde el WAMP
 2. Creamos el primer comando mysql> create database prueba;ENTER

3. Te indica que la consulta o query ha sido correcta, y que una fila ha sido afectada
4. Comprobamos desde phpMyadmin, como la base de datos prueba está creada
5. Ahora eliminamos la base de datos desde la consola con la sentencia

```
mysql> drop database prueba ENTER
```

nos sale un cuadro de diálogo en el que pone que la operación se ha realizado con éxito, y si vamos a phpmyadmin vemos que la base de datos ya no está
6. Show databases. En consola nos da un listado de todas las bases de datos

7.

Es importante que conozcamos el lenguaje SQL lenguaje de acceso a datos universal, por lo que nos sirve para aplicar en distintos modelos de bases de datos relacionales. Por ejemplo :

- a. ACCESS,
- b. MySQL
- c. Oracle
- d. SQL server

Vamos a intentar conectar nuestras páginas php con las bases de datos MySQL, esto se realiza con consultas SQL. Vamos a ver solo lo necesario para crear, modificar y consultar una base de datos MySQL desde una página web php

- Comencemos q dejando claro conceptos como
 - Tabla: estructura que se compone de filas (registros en b de datos) y columnas(campos en base de datos)
finalidad almacenar información
 - ej

○ nombre	○ clave
○ Antonio	○ 1234
○ maría	○ 5555
 - Por ejemplo si queremos almacenar nombres y claves para que en un futuro unas personas puedan hacer login dentro de una página web o una base de datos lo almacenamos en una tabla
 - IMPORTANTE:
 - LAS COLUMNAS O CAMPOS DEBEN TENER UN NOMBRE(NOMBRE-CLAVE EN EL EJEMPLO)
 - CADA CAMPO DEBE DEFINIR EL TIPO D DATOS QUE ALMACENARÁ

En el ejemplo la primera columna almacena datos de tipo texto y la segunda datos de tipo numérico o de tipo texto
 -

Sus siglas en español se traducen como lenguaje estructurado de consultas, aunque no solo sirve para hacer consultas, también permite:

- crear bases de datos nuevas,
 - añadir y eliminar campos,
 - modificar las propiedades de los campos.
 - Establecer que tipo de relación existe entre dos tablas de una base de datos
- Lenguaje creado por IBM, antiguamente llamado Sequel

Proceso:

- Partimos de un ordenador (servidor local o remoto) con una base de datos.
- Si queremos obtener información de esa base de datos , se hace una solicitud SQL
- Esa solicitud se le hace al servidor, el cual interactúa con la base de datos que está alojada en su interior, y da una respuesta con los datos que le preguntamos
- Necesitamos un gestor de bases de datos (SGDB= Sistema Gestor de Bases de Datos), y aquí pueden ser muchos: MySQL, ORACLE, ACCESS,... todos ellos utilizan el estándar SQL para pedir información a una base de datos

SQL no es considerado un lenguaje de programación propiamente dicho, porque carece de las estructura de un lenguaje de programación (condicionales, bucles,...), se considera un lenguaje de acceso a datos, o un lenguaje estructurado de consulta. Hay un estándar (ANSI SQL) de este lenguaje aunque luego cada gestor tiene su propio dialecto.

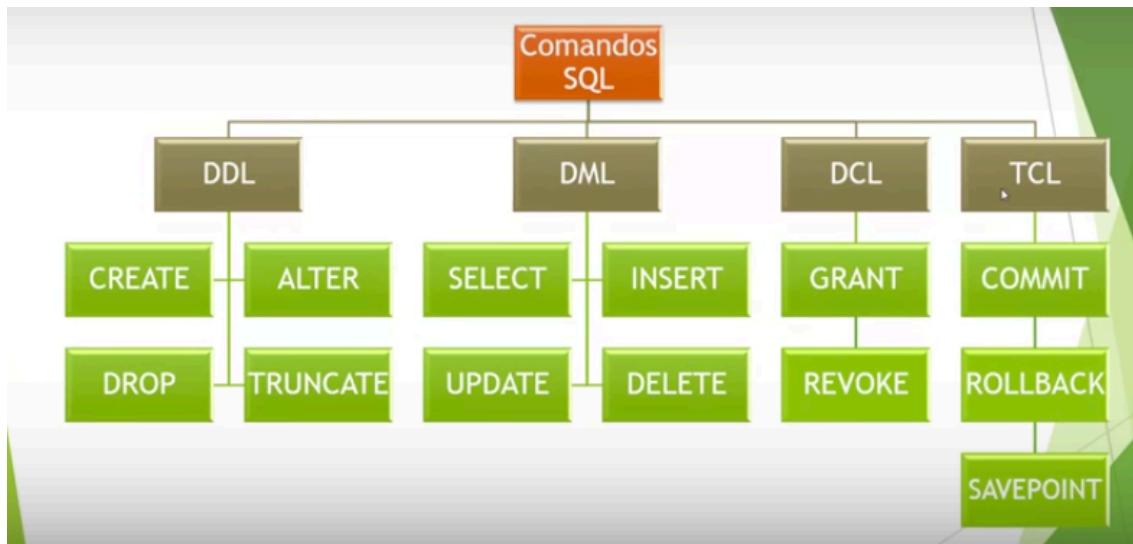
SECUENCIA SQL

Una secuencia SQL no es ni más ni menos que una frase en inglés compuesta de comandos.

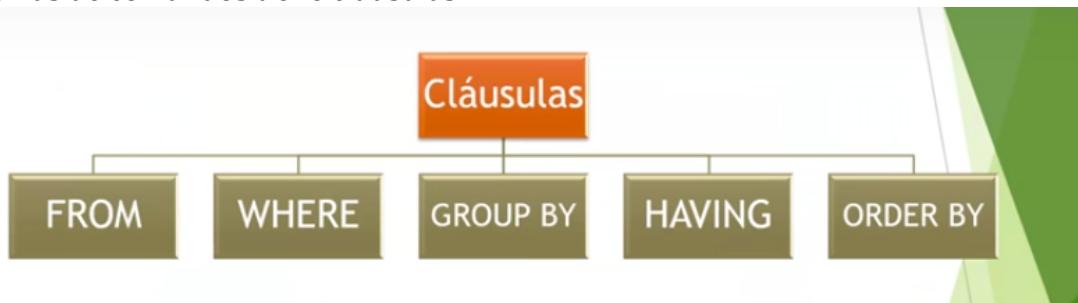
Hay 4 grandes tipos DE COMANDOS SQL:

- Los comandos DDL (Data Definition Language). Para crear y modificar la estructura de una base de datos. Puedes modificar, borrar , crear tablas, añadir un campo a una tabla....
- Los comandos DML (Data Manipulation Language). Se utilizan para hacer consultas de selección y de acción:
 - seleccionar registros de una base de datos. Consultas
 - insertar nuevos registros
 - actualizar registros
- Los comandos DCL (Data Control Language): para proporcionar seguridad
- Los comandos TCL (transacción control lenguaje). Gestión de los cambios en los datos

COMANDOS SQL



Además de comandos tiene cláusulas



En definitiva una instrucción SQL que me permita obtener información de una base de datos o modificarlos, no es más que la unión de:
COMANDO+ CLAUSULA+OPERADORES+ FUNCIONES DE AGREGADO

Uniendo todo esto que no es más que una frase en inglés, creamos una instrucción SQL. NO es necesario que lleve los 4 componentes. Lo único necesario es un comando y una cláusula.

Ejercicio

1. Te proporciono dos tablas en formato ODE
2. Creamos una base de datos nueva en phpmyadmin, llamado curso_SQL (sin espacios)
3. Vamos a la pestaña importar> y una vez importada, nos aseguramos de que el formato sea OpenDocumentSpreadsheet
4. Seleccionamos todos los checkbox de opciones específicas de formato.
5. Ya tendríamos almacenada la tabla de datos SQL

6. Ahora vamos a hacer una consulta que me devuelva los clientes de Madrid.
Con una sentencia SQL
7. Recordando que está formada por comando + clausula + operadores y funciones
8. PRIMERO SELECCIONAMOS LA CONSULTA CLIENTES. Si se trata de una consulta de selección simple, **EL COMANDO** que hay que utilizar es SELECT **CAMOS QUE QUIERO VER** EMPRESA, DIRECCION, POBLACION FROM CLIENTES.
9. **FROM ES EL COMANDO**
10. Con esta instrucción vemos todos los registros de los campos empresa, dirección, población.
11. Le damos a ejecutar y aparecen todos los clientes que he solicitado en la instrucción

SQL TEMA 7

REPASEMOS LAS CLAUSULAS

Cláusula	Descripción
FROM	Especifica la tabla de la que se quieren obtener los registros
WHERE	Especifica las condiciones o criterios de los registros seleccionados
GROUP BY	Para agrupar los registros seleccionados en función de un campo
HAVING	Especifica las condiciones o criterios que deben cumplir los grupos
ORDER BY	Ordena los registros seleccionados en función de un campo

REPASEMOS LOS OPERADORES: DE COMPARACION

Operador	Significado
<	Menor que
>	Mayor que
=	Igual que
>=	Mayor o igual que
<=	Menor o igual que
<>	Distinto que
BETWEEN	Entre. Utilizado para especificar rangos de valores
LIKE	Cómo. Utilizado con caracteres comodín (? *)
In	En. Para especificar registros en un campo en concreto

Los más conocidos, los 5 primeros: muy utilizados a la hora de establecer criterios Between. Operador de comparación para establecer rangos de valores. Ej rangos de fechas, rangos de precios.....

LIKE utilizado con los caracteres comodín

IN para especificar una serie de registros en un campo en concreto

LOGICOS

Operador	Significado
AND	Y lógico
OR	O lógico
NOT	Negación lógica

ORDEN DE ESCRITURA DE LAS CLAUSULAS



EJEMPLO PARA VER TRES CAMPOS DE LA TABLA PRODUCTOS DE SECCION CERAMICA

```
1 SELECT NOMBREARTICULO, SECCIÓN, PRECIO FROM PRODUCTOS WHERE SECCIÓN='CERÁMICA'|
```

CLAUSULA ORDER BY

Clausula que ordena los registros seleccionados, en función de un campo

Ejemplo:

Vamos a hacer una selección de dos productos de sección: deportes y cerámica

```
SELECT * FROM PRODUCTOS WHERE SECCIÓN='DEPORTES' OR SECCIÓN='CERÁMICA'|
```

Vamos a ordenar ahora todos de forma alfabética, por lo que si antes salían deportes primer, ahora sale cerámica

Y lo hacemos así:

```
DUCTOS WHERE SECCIÓN='DEPORTES' OR SECCIÓN='CERÁMICA' ORDER BY SECCIÓN|
```

si simplemente añadimos DESC, sería ordenación inversa (de mayor a menor)

Probemos ahora a hacer ordenar por precios (y se desordena el resto)

¿Y si queremos ordenar por dos criterios?

En el ejemplo por sección y por precio

y si encima queremos un orden uno ascendente otro descendente

```
SELECT * FROM PRODUCTOS WHERE SECCIÓN='DEPORTES' OR SECCIÓN='CERÁMICA' ORDER BY SECCIÓN, PRECIO DESC|
```

incluso un tercer criterio de ordenación:

```
= 'CERÁMICA' ORDER BY SECCIÓN, PAÍSDEORIGEN, PRECIO|
```

CONSULTAS DE AGRUPACION

También conocidas como consultas de totales

Son consultas en que se hacen cálculos por grupos:

- Cogemos los registros de una tabla y los agrupamos en base a un registro o campo
- Una vez que los tenemos agrupados, con estos registros realizamos un cálculo: un conteo, una media,

Funciones de agregado

Función	Descripción
→ AVG	Calcula el promedio de un campo
→ COUNT	Cuenta los registros de un campo
→ SUM	Suma los valores de un campo
→ MAX	Devuelve el máximo de un campo
→ MIN	Devuelve el mínimo de un campo

Vamos a hacer una primera consulta de agrupación que me diga cuanto suman los productos de las secciones

1. Creamos un campo de agrupación y un campo del cálculo.
-como campo de agrupación utilizamos el campo sección
-Campo de cálculo : PRECIO

```
SELECT SECCIÓN, SUM(PRECIO) FROM PRODUCTOS GROUP BY SECCIÓN
```

2. Vamos ahora a ordenar por precio .
Si intentamos hacerlo por ORDER BY PRECIO, dará error porque no hay ningún campo con ese nombre (en la consulta, en la tabla si)
- 3.
4. Esto lo solucionamos con lo que se conoce como ALIAS

```
) AS SUMA_ARTICULOS FROM PRODUCTOS GROUP BY SECCIÓN ORDER BY SUMA_ARTICULOS
```

```
: SELECT SECCIÓN, AVG(PRECIO) AS MEDIA_ARTICULOS FROM PRODUCTOS GROUP BY SECCIÓN HAVING SECCIÓN='DEPORTES' OR SECCIÓN='CONFECIÓN'
```

Vamos a ver ahora el caso en que si queremos sacar la media solo de cerámica y deportes. Utilizamos el Where En estos casos de consultas agrupadas es HAVING

```
IA_ARTICULOS FROM PRODUCTOS GROUP BY SECCIÓN WHERE
```

HAVING

Ahora en la tabla clientes quiero que nos digan cuantos clientes hay de cada población. LOhacemos con la función COUNT()

```
POBLACIÓN, COUNT(CÓDIGOCLIENTE) AS N_CLIENTES FROM CLIENTES GROUP BY POBLACIÓN
```

- Ojo!!No cuenta los registros en blanco. Por eso para más seguridad se utiliza un campo clave
- Ahora queremos saber el precio del articulo más caro de la sección de confeccion:

```
1 | SELECT SECCIÓN, MAX(PRECIO) AS PRECIO_MAS_ALTO FROM PRODUCTOS WHERE SECCIÓN='CONFECCIÓN' GROUP BY SECCIÓN
```

Y por último queremos saber la media de el precio de los artículos por secciones y que lo ordene luego por precio

```
SELECT SECCIÓN, AVG(PRECIO) AS MEDIA_ARTICULOS FROM PRODUCTOS GROUP BY SECCIÓN HAVING SECCIÓN='DEPORTES' OR SECCIÓN='CONFECCIÓN' ORDER BY MEDIA_ARTICULOS
```

OPERACIONES

- PRIMER EJERCICIO:SACAR EL IVA

```
1 | SELECT NOMBREARTÍCULO, SECCIÓN, PRECIO, ROUND(PRECIO*1.21,2) AS PRECIO_CON_IVA FROM PRODUCTOS
```

- SEGUNDO EJERCICIO: RESTA 3

```
1 | SELECT NOMBREARTÍCULO, SECCIÓN, PRECIO, PRECIO-3 AS PRECIO_DTO FROM PRODUCTOS
```

- DIA Y HORA ACTUALES A LA HORA DE EJECUTAR LA CONSULTA NEW()

```
1 | SELECT NOMBREARTÍCULO, SECCIÓN, PRECIO, FECHA, NOW() AS DIA_DE_HOY FROM PRODUCTOS WHERE SECCIÓN='DEPORTES'
```

CONSULTAS MULTITABLA

- Hasta ahora todas las consultas que hemos hecho están basadas en una única tabla, pero las bases de datos suelen tener muchas tablas
- Se le llama consulta multitabla o consulta de unión
- Estas consultas de unión se dividen en externa e interna y a continuación mostramos los operadores de cada uno de ellos :

► Consultas Multitabla / Consultas de Unión

► Unión Externa:

- ▶ Union
- ▶ Union All
- ▶ Except
- ▶ Intersect
- ▶ Minus

► Unión interna

- ▶ Inner join
- ▶ Left Join
- ▶ Right Join

Respecto a los operadores externos, los tres últimos no son soportados por todos los gestores de bases de datos, aunque realmente pertenezcan al estándar SQL

- Veamos los dos primeros:

- UNION

Permite unir en una consulta varias tablas que podamos tener almacenadas en nuestra base de datos

Requisitos que deben cumplir las tablas:

1. Ambas deben tener el mismo número de campos
2. Los campos deben tener tipos de datos compatibles. Ej un campo PRECIO con datos numéricos y otro campo PRECIO con datos de texto
3. El nombre de los campos puede ser diferente. EJ COD ART en una tabla y CODIGO ARTICULO , en la otra

Una vez que se cumplen los requisitos, y realizando una consulta de unión, lo que ocurre es que la información de ambas tablas se fusionan en una única consulta. Y esa consulta toma como campos los de la tabla 1

Ejemplo práctico

1. Primero modificar la tabla para que los datos sean así:

A	B	C	D	E	F	G	H	I
1	CÓDIGOARTÍCULO	SECCIÓN	NOMBREARTÍCULO	PRECIO	FECHA	IMPORTADO	PAÍSDEORIGEN	FOTO
2	AR50	ALTA COSTURA	TRAJE CABALLERO	1.284,58	11/03/2002	VERDADERO	ITALIA	
3	AR51	DEPORTES DE RIESGO	RAQUETA TENIS	1.093,47	20/03/2000	VERDADERO	USA	
4	AR52	DEPORTES DE RIESGO	MANCUERNAS	1.060,00	13/09/2000	VERDADERO	USA	
5	AR53	ALTA COSTURA	SERRUCHO	1.030,26	23/03/2001	VERDADERO	FRANCIA	
6	AR54	ALTA COSTURA	PANTALÓN SEÑORA	1.174,23	10/01/2000	VERDADERO	MARRUECOS	
7	AR55	ALTA COSTURA	CAMISA CABALLERO	1.067,13	11/08/2002	FALSO	ESPAÑA	
8	AR56	DEPORTES DE RIESGO	PISTOLA OLÍMPICA	1.046,73	02/02/2001	VERDADERO	SUECIA	
9	AR57	ALTA COSTURA	BLUSA SRA.	1.101,06	18/03/2000	VERDADERO	CHINA	
10	AR58	ALTA COSTURA	CAZADORA PIEL	1.522,69	10/07/2001	VERDADERO	ITALIA	
11	AR59	DEPORTES DE RIESGO	BALÓN RUGBY	1.111,64	11/11/2000	VERDADERO	USA	
12	AR60	DEPORTES DE RIESGO	BALÓN BALONCESTO	1.075,27	25/06/2001	VERDADERO	JAPÓN	
13	AR61	ALTA COSTURA	ABRIGO CABALLERO	1.500,00	05/04/2002	VERDADERO	ITALIA	
14	AR62	DEPORTES DE RIESGO	BALÓN FÚTBOL	1.043,91	04/07/2002	FALSO	ESPAÑA	
15	AR63	ALTA COSTURA	ABRIGO SRA	1.360,07	03/05/2001	VERDADERO	MARRUECOS	
16	AR64	DEPORTES DE RIESGO	CRONÓMETRO	1.439,18	03/01/2002	VERDADERO	USA	
17	AR65	ALTA COSTURA	CINTURÓN DE PIEL	1.004,33	12/05/2002	FALSO	ESPAÑA	
18	AR66	DEPORTES DE RIESGO	CAÑA DE PESCA	1.270,00	14/02/2000	VERDADERO	USA	
19	AR67	DEPORTES DE RIESGO	BOTA ALPINISMO	1.144,00	05/05/2002	FALSO	ESPAÑA	
20	AR68	DEPORTES DE RIESGO	PALAS DE PING PONG	1.021,60	02/02/2002	FALSO	ESPAÑA	
21								
22								
23								
24								

2. Tenemos ahora la tabla de productos antigua y la nueva
3. Imaginemos que queremos hacer una consulta en la que nos devuelva todos los registros de DEPORTES, o DEPORTES DE RIESGO

la sintaxis sería esta

```
1 SELECT * FROM PRODUCTOS WHERE SECCIÓN='DEPORTES' UNION SELECT * FROM PRODUCTOSNUEVOS WHERE SECCIÓN='DEPORTES DE RIESGO'
```

- 4.

ojo ¡!! Los nombres que aparecen en los campos en la consulta, son los de la primera tabla

Otro ejercicio:

Establecer dos criterios diferentes para cada tabla

Vamos a hacer que nos muestre los artículos de la tabla de productos cuyo precio es superior a 500 euros y los productos de la tabla segunda cuya sección sea "alta costura"

Ojo!! Ambas consultas deben tener el mismo número de campos, en el SELECT

```
1 | SELECT * FROM PRODUCTOS WHERE PRECIO>500 UNION SELECT * FROM PRODUCTOSNUEVOS WHERE SECCIÓN='ALTA COSTURA'
```

VAMOS A VER AHORA EL SEGUNDO OPERADOR UNION ALL

UNION en caso de que haya registros repetidos en la tabla uno y dos, esos registros repetidos los muestra solo una vez

UNION ALL no, lo repite tantas veces como existan

EJERCICIO

1. Lo primero de todos duplicamos un elemento con INSERT INTO en la otra tabla (lo tenemos en productos y en productos nuevos)

```
1 | INSERT INTO 'PRODUCTOSNUEVOS' ('CÓDIGOARTÍCULO', 'SECCIÓN', 'NOMBREARTÍCULO', 'PRECIO', 'FECHA', 'DEPORTES', 'PAÍSESORIGEN', 'FOTO') VALUES ('AR05', 'DEPORTES', 'MANZUERAS', '50.0000', '2000-09-12', 'VERDADERO', 'USA', 'IRALA')
```

2. Si lo probamos con unión solo saldrá una vez (devolverá solo uno, y siempre el de la tabla uno)

```
1 | SELECT * FROM PRODUCTOS WHERE SECCIÓN='DEPORTES' UNION SELECT * FROM PRODUCTOSNUEVOS
```

3. Si ahora cambiamos el unión por UNION ALL lo que hemos dicho a la consulta es decirle que nos muestren todos

Vamos ahora a :

- manejar la consola SQL ,
- ver cómo crear y eliminar tablas con SQL,
- ver como crear y eliminar campos con SQL
- insertar registros con SQL

Todo esto podemos manejarlo desde PHPMyAdmin sin necesidad de programar ni una sola línea de lenguaje SQL, pero realmente nuestro objetivo es desde una página web php, que poda comunicarnos y manipular una base de datos remota desde esta página. Esto solo se puede hacer si se conoce SQL

Hemos creado una base de datos

- Desde la misma pestaña base de datos de PHPMyAdmin

Bases de datos

Crear base de datos Cotejamiento

Nota: Activar aquí las estadísticas de la base de datos podría causar tráfico pesado entre el servidor

Base de datos	Cotejamiento	Opciones
information_schema	utf8_general_ci	[] Comprobar los privilegios
mysql	latin1_swedish_ci	[] Comprobar los privilegios
performance_schema	utf8_general_ci	[] Comprobar los privilegios
test	latin1_swedish_ci	[] Comprobar los privilegios

- Desde la consola del sistema

```
c:\wamp\bin\mysql\mysql5.6.17\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 113
Server version: 5.6.17 MySQL Community Server (GPL)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

- Y también desde la consola SQL de PHPMyAdmin

Ejecute la o las consultas SQL en el servidor "mysql wampserver":

```
1 CREATE DATABASE PRUEBAS;
```

Limpiar

[Delimitador] Mostrar esta consulta otra vez Mantener la caja de texto con la consulta

Continuar

- Vamos a crear una tabla desde la consola desde PHPMyAdmin dentro de la BBDD pruebas llamada DATOSPERSONALES, con sus campos de texto que serían NIF, NOMBRE, APELLIDO, EDAD
 - NIF su tipo es varchar 10
 - nombre varchar 15
 - Apellido varchar 20
 - EDAD int 2(o INTEGER para indicar que es un numero entero)

```
1 CREATE TABLE DATOSPERSONALES (NIF VARCHAR(10), NOMBRE VARCHAR(15), APELLIDO VARCHAR(20), EDAD INT(2));
```

¿Cómo eliminar un campo de una tabla?

primero elegimos la tabla a modificar (ALTER TABLE) y luego decimos que lo que queremos es borrar un campo de dicha tabla.

```
1 ALTER TABLE DATOSPERSONALES DROP EDAD;
```

¿cómo agregamos un campo?

Si queremos agregar EDAD, Con ADD que significa añadir . COLUMN (lo que agrega es una columna)

```
1 ALTER TABLE DATOSPERSONALES ADD COLUMN EDAD INT (2);
```

Para agregar registros dentro de una tabla utilizamos INSERT INTO (Introducir dentro de la tabla,)

Servidor: mysql wampserver » Base de datos: pruebas » Tabla: datospersonales

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Disparos

MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0.0300 seg)

```
INSERT INTO DATOSPERSONALES (NIF, NOMBRE, APELLIDO, EDAD) VALUES ("51972854W", "MARÍA", "GOMEZ", 27)
```

Mostrar ventana de consultas SQL

Para comprobarlo en PHPMyAdmin...

Servidor: mysql wampserver » Base de datos: pruebas » Tabla: datospersonales

Examinar Estructura SQL Buscar Insertar Exportar Importar

La selección actual no contiene una columna única. La edición de la grilla y los enlaces de copiado, eliminación y edición

Mostrando filas 0 - 0 (total de 1. La consulta tardó 0.0000 seg)

```
SELECT * FROM `datospersonales`
```

Número de filas: 25

+ Opciones

NIF	NOMBRE	APELLIDO	EDAD
51972854W	MARIA	GOMEZ	27

Vamos a ver ahora como conectar a una base de datos mysql desde una pagina web php

Esto se hace normalmente porque vas buscando una información. Un registro o grupo de registros, información que se encuentra almacenada en esa base de datos
Y después de ver cómo conectarnos a esa base de datos, tenemos que saber cómo hacer consultas de información con instrucciones SQL

Hay dos grandes consultas en SQL, las consultas de selección y las consultas de acción

CONSULTAS DE ACCIÓN : Las que modifican la información que se encuentra almacenada en la base de datos

CONSULTAS DE SELECCIÓN : Las que permiten capturar u obtener información la base de datos , sin modificarla.

¿cómo se generan las consultas de Selección? Con el comando SELECT, los campos que deseas ver de esa consulta, de la tabla que sea

```
SELECT NIF, NOMBRE, APELLIDO, EDAD FROM DATOSPERSONALES
```

A continuación del comando SELECT, introduces los campos que quieras que se vean , separados por comas, a continuación clausula FROM, seguido de la tabla de la que quiere hacer la consulta.

```
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> USE PRUEBAS;
Database changed
mysql> SELECT * FROM DATOSPERSONALES;
+-----+-----+-----+
| NIF | NOMBRE | APELLIDO | EDAD |
+-----+-----+-----+
| 51972854M | MARqMA | GOMEZ | 27 |
| 50193568M | Juan | Gómez | 18 |
| 51982457B | Elena | Martín | 27 |
+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

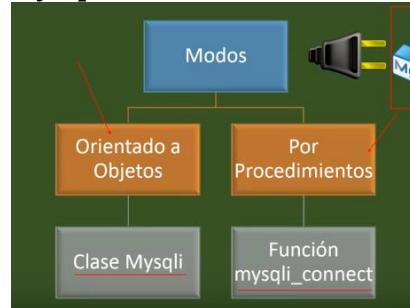
Para conectar con una base de datos MYSQL, necesitamos 4 datos muy importantes:



Una vez establecida la conexión con la base de datos, para poder obtener información, podemos trabajar de dos formas:

- Utilizar instrucciones PHP orientadas a objetos para poder conectarnos con esa base de datos y obtener información. Con la **clase mysqli** con sus propiedades y métodos
- Podemos utilizar instrucciones por procedimientos para obtener el mismo resultado. Mediante la **función mysqli_connect (en versiones antiguas)**

mysql_connect , anteriores a la versión 5.5 de PHP)



PRACTICA:

Vamos a crear una conexión con nuestra base de datos:

1. Primero almacenamos los cuatro datos importantes en nuestra página web llamada "conexión_BBDD".
2. Segundo conectamos a la base de datos.
hemos visto que hay dos formas
 - a. Por programación orientada a objetos
 - b. Por procedimientos
para ello creamos una variable a la que le asignamos la función mysqli_funcion que nos pide los datos anteriores: nombre de hostinig, usuario, contraseña y nombre
 - c. A continuación creamos la consulta o query, para que nos devuelva todos los campos de la tabla DATOS PERSONALES, Y asignamos a una variable (resultado), la función mysqli_query que nos pide los datos de la conexión y la consulta.
 - d. Ese resultado es el resulset que almacena la respuesta a la consulta y para saber lo que lleva esa tabla utilizamos la instrucción mysqli_fetch_row y lo almacenamos en la variable fila y luego lo imprimimos mediante un array.

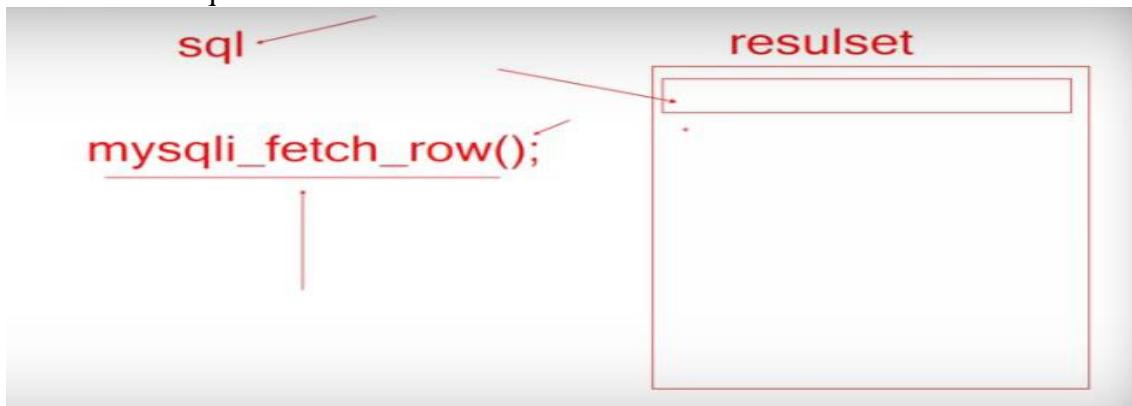
```
<?php
$db_host="localhost";
$db_nombre="pruebas";
$db_usuario="root";
$db_contra="";
$conexion=mysqli_connect($db_host,$db_usuario,$db_contra,$db_nombre);
$consulta="SELECT * FROM DATOSPERSONALES";
$resultados=mysqli_query($conexion, $consulta);
$fila=mysqli_fetch_row($resultados);
echo $fila[0] . " ";
echo $fila[1] . " ";
echo $fila[2] . " ";
echo $fila[3] . " ";

```

	sesiones_jugadas.php	1KB PHP S...	03/04/2015 ...
	proporciona_datos.php	1KB PHP S...	03/04/2015 ...
	flujo_ejecucion.php	1KB PHP S...	11/04/2015 ...
	datos_canciones.php	1KB PHP S...	11/04/2015 ...
	ejemplo_operadores.php	1KB PHP S...	11/04/2015 ...
	ejemplo_estatica.php	1KB PHP S...	11/04/2015 ...
	trabajo_operadores.php	2KB PHP S...	20/04/2015 ...
	validacion.php	1KB PHP S...	20/04/2015 ...
	Uso_constantes.php	1KB PHP S...	25/04/2015 ...
	operaciones_matematicas.php	1KB PHP S...	29/04/2015 ...
	ejemplo_operadores.html	2KB PHP S...	29/04/2015 ...
	funciones_matematicas.php	1KB PHP S...	03/05/2015 ...
	prioridad_operadores.php	1KB PHP S...	04/05/2015 ...
	ejemplo_comparaciones.php	2KB PHP S...	04/05/2015 ...
	validacion_condiciones.php	1KB PHP S...	08/05/2015 ...
	ejemplo_while_1.php	1KB PHP S...	11/05/2015 ...
	trabajo_strings.php	1KB PHP S...	15/05/2015 ...
	calculadora.php	2KB PHP S...	15/05/2015 ...
	ejemplo_if.php	1KB PHP S...	15/05/2015 ...
	ejemplo_f_preferidas.php	1KB PHP S...	22/05/2015 ...
	ejemplo_param_referencia.php	1KB PHP S...	26/05/2015 ...
	POO_1.php	1KB PHP S...	11/06/2015 ...
	vehiculos.php	2KB PHP S...	11/06/2015 ...
	Uso_comisionario.php	1KB PHP S...	17/06/2015 ...
	Concesionario.php	2KB PHP S...	17/06/2015 ...
	tar.php	1KB PHP S...	23/06/2015 ...
	arrays.php	1KB PHP S...	23/06/2015 ...
	array_dimensiones.php	1KB PHP S...	29/06/2015 ...

- e. Con esto conseguimos que nos imprima todos los campos del primer registro

Vimos cómo el el recorset o resulset, no es más que una tabla virtual donde se almacena la instrucción sql.



Es decir al ejecutar esta instrucción sql , la respuesta se almacena en una tabla virtual. Esta función mysqli_fetch_row(), lo que hace es acceder la primera vez al primer registro de esa resulset. Si la llamas una segunda vez automáticamente lo que hace es acceder al segundo registro y así sucesivamente... por lo tanto vamos a copiar y pegar dos veces la instrucción con los echo...(ojo añadir un br al final del primero para que salte a la línea siguiente.

```
$resultados=mysqli_query($conexion, $consulta)

$fila=mysqli_fetch_row($resultados);
echo $fila[0] . " ";
echo $fila[1] . " ";
echo $fila[2] . " ";
echo $fila[3] . " ";

$fila=mysqli_fetch_row($resultados);
echo $fila[0] . " ";
echo $fila[1] . " ";    I
echo $fila[2] . " ";
echo $fila[3] . " ";
```

Este código te devuelve los dos primeros registros.

Si siguiéramos copiando y pegando una tercera vez, te leerá la tercera línea, y as

...pero si tienes 3.000 líneas no lo vas a cortar y pegar 3000 veces, así que la solución es hacer un bucle

Primero inicias una variable registros a 1
Y luego haces el bucle

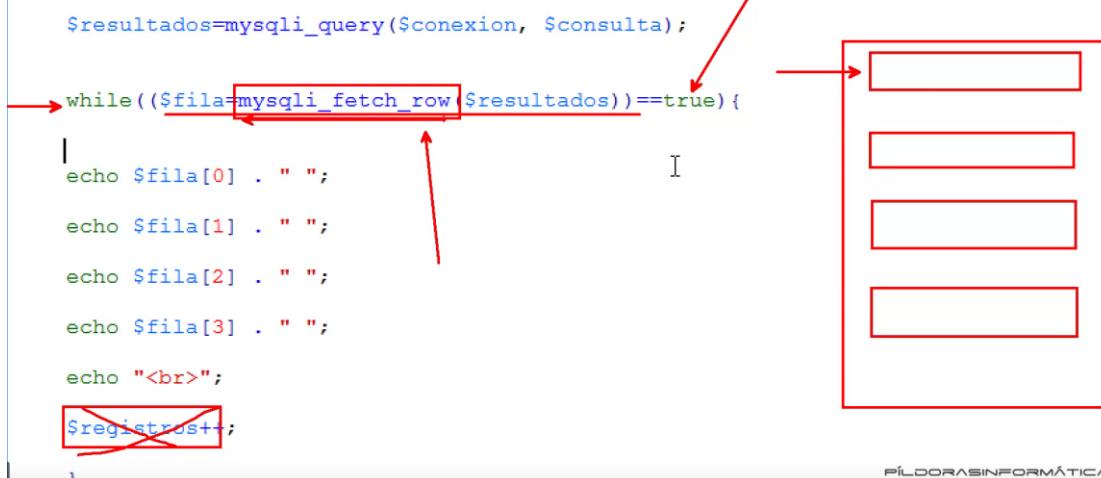
```
$resultados=mysqli_query($conexion, $consulta);  
  
$registros=1;  
  
while($registros<=3){  
  
    $fila=mysqli_fetch_row($resultados);  
  
    echo $fila[0] . " ";  
  
    echo $fila[1] . " ";  
  
    echo $fila[2] . " ";  
  
    echo $fila[3] . " ";  
  
    echo "<br>";  
  
    $registros++;  
}
```

While tiene como condición la instrucción \$registros<=3 , por que sabemos que son 3 registros.

Qué conseguimos con esto. Partimos de la variable registros es 1, y el while indica que mientras registros sea menor que tres se ejecute la función mysqli_fetch_row(), con el parámetro \$resultados donde se almacena la conexión a la base de datos.

Según va aumentando el bucle se va imprimiendo cada vez un echo, hasta imprimir todos.

El problema está en que no sepamos el total de registros en la tabla. Así que la forma definitiva de hacerlo sería: cambiar la condición del while, y cambiarlo por una condición que indique: "ejecuta la instrucción mientras siga habiendo información".



```
$resultados=mysqli_query($conexion, $consulta);  
  
while(($fila=mysqli_fetch_row($resultados))==true){  
  
    echo $fila[0] . " ";  
  
    echo $fila[1] . " ";  
  
    echo $fila[2] . " ";  
  
    echo $fila[3] . " ";  
  
    echo "<br>";  
  
$registros++;  
}
```

Con esto conseguimos que el código cuando entra una vez en el bucle while, ejecuta la instrucción , y si es capaz de ejecutarla es por que en la tabla hay información(registros)

Funciona así: cuando entra en esta condición del while, y es (true), verdad que encuentra registro, lo imprime,así hasta que llega al final, y cuando llega al final, y entonces devuelve false, sale del bucle.

La condición del While se podría simplificar, eliminando el true, puesto que al ser una condición booleana, se toma por defecto que esto es true.

En definitiva podría quedar así:

```
while($fila=mysqli_fetch_row($resultados)) {  
  
    echo $fila[0] . " ";  
  
    echo $fila[1] . " ";  
    ]  
    echo $fila[2] . " ";  
  
    echo $fila[3] . " ";  
  
    echo "<br>";  
  
}
```

Por último deberíamos cerrar la conexión.
Me explico: hemos abierto una conexión, esa conexión ha permanecido abierta todo el rato. Esto consume recursos. Si la dejas abierta ahora no vas a tener ningún problema pero para optimizar recursos, es una vez que ya no necesites acceder a la base de datos, cerra r la conexión.

Y la conexión se cierra con el **mysqli close**, indicando que conexión quieres cerrar.

.

```
echo $fila[0] . " ";  
echo $fila[1] . " ";  
echo $fila[2] . " ";  
echo $fila[3] . " ";  
echo "<br>";  
}  
  
mysqli_close($conexion);  
?>
```

Un mismo documento php, se puede conectar a un montón de bases de datos por lo que será importante que se vayan cerrando las conexiones, para optimizar recursos

- Cómo manejar los errores que puedan ocurrir en el proceso de conexión con la base de datos.
 - Por haber escrito mal la dirección de la base de datos
 - Por haber escrito mal el nombre de la base de datos

Errores

- ❖ para que nos avisara que había un fallo en la conexión utilizamos el if con mysqli_connect_errno, después de el mysqli_connect.

```
$conexion=mysqli_connect($db_host,$db_usuario,$db_contra,$db_nombre);
if(mysqli_connect_errno()){
    echo "Fallo al conectar con la BBDD";
    exit();
}

$consulta="SELECT * FROM DATOSPERSONALES";
```

Con un mensaje de error que indique que hay un fallo en la conexión. Esta función se ejecuta cuando hay algún error en la conexión con la base de datos.
muy importante!. Que después de ejecutar el mensaje, salga del código php, (con el exit(), por que si no seguirá leyendo código, y pasará a intentar ejecutar la consulta,...y dará fallo tras fallo.

provoco poner mal la dirección del servidor (pongo localhooost o 12777,0,0,1 en lugar de localhost o 127.0.0.1)

- ❖ para que no nos de error de tipografía como en el ejemplo....

51972854W MARIA GOMEZ 27

utilizamos el set_Charset (después de la conexión), para indicar que queremos utilizar el juego de caracteres utf 8

mysqli_set_charset(\$conexion, "utf8");

y entonces ya nos saldrán los acentos normales.

- ❖ Ahora vamos a imaginarnos que el error está en la base de datos (por ejemplo

porque ponemos mal el nombre de la base de datos...pruebassss);
En este caso podremos quitar el comando opcional de mysqli_connect ,
\$db_nombre y lo sustituimos, líneas mas abajo, por **mysqli_select_db(\$conexion, \$db_nombre)** añadiendo or die (*no se encuentra la BBDD*)

```
<?php  
    $db_host="localhost";  
    $db_nombre="pruebassss";  
    $db_usuario="root";  
    $db_contra="";  
  
    $conexion=mysqli_connect($db_host,$db_usuario,$db_contra);  
  
    if(mysqli_connect_errno()){  
  
        echo "Fallo al conectar con la BBDD";  
  
        exit();  
  
    }  
  
    mysqli_select_db($conexion, $db_nombre) or die ("No se encuentra la BBDD");  
  
    mysqli_set_charset($conexion, "utf8");  
  
    $consulta="SELECT * FROM DATOSPERSONALES";  
  
    $resultados=mysqli_query($conexion, $consulta);
```

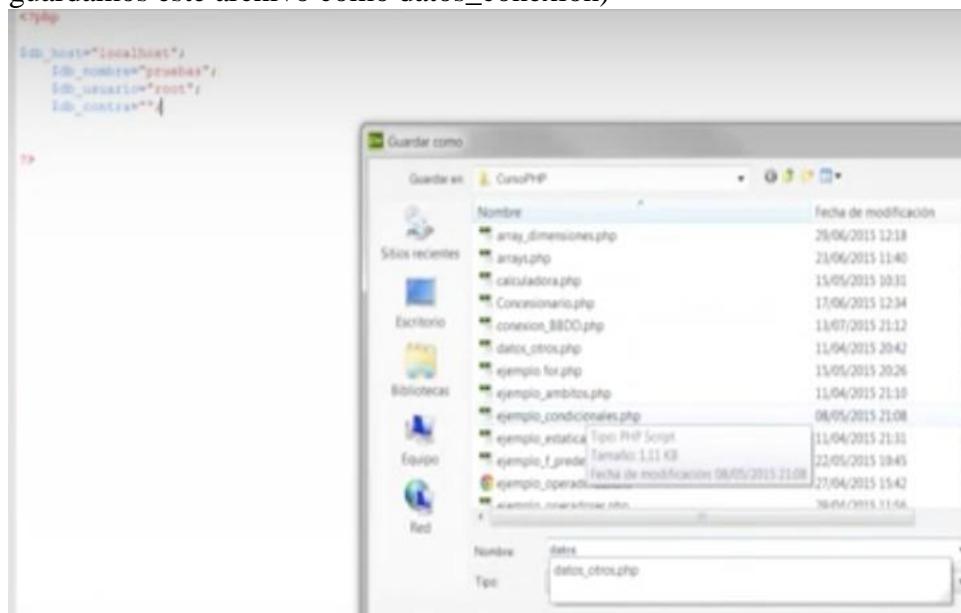
De este modo nos va a devolver un mensaje que es más exacto puesto que dirá **no se encuentra la BBDD**, en caso de que haya un error en el nombre.

Hemos visto:

- ✓ como conectar con una base de datos Mysql,
- ✓ como realizar una consulta mysql para obtener información
- ✓ como manejar los errores

vamos a ver ahora como **compartir los datos de conexión a BBDD**, entre diferentes archivos php. De esta forma no será necesario repetir todos los datos de configuración en los diferentes archivos.

1. Primero introducimos en un documento a parte los datos de conexión (ej guardamos este archivo como datosConexion)



2. En todos aquellos archivos que queramos establecer una conexión a la base de datos debemos introducir la función **require** y decirle que nos incluya este archivo.(una vez que enlazamos el php con el documento datosConexion, DW lo reflejará)

En el archivo anterior, quedaría algo así

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Documento sin título</title>
</head>

<body>

<?php
+
require("datosConexion.php");

$conexion=mysqli_connect($db_host,$db_usuario,$db_contra);

if(mysqli_connect_errno()){

    echo "Fallo al conectar con la BBDD";
    exit();
}

mysqli_select_db($conexion, $db_nombre) or die ("No se encuentra la BBDD");
mysqli_set_charset($conexion, "utf8");

$consulta="SELECT * FROM DATOSPERSONALES";
$resultados=mysqli_query($conexion, $consulta);

while($fila=mysqli_fetch_row($resultados)) {

    echo $fila[0] . " ";
}
```

3. Hasta ahora hemos creado la tabla nosotros desde phpMyAdmin, con pocos registros, pero si quisiera trabajar con un volumen de registros extenso, debemos importar ahora una tabla de Excel. Vamos a importar una tabla de Excel facilitada llamada **productos**

- Primero debemos transforma una extensión **xlsx** de Excel, a **ods** (si es un Access, primero se exporta a Excel, y luego de Excel se pasa a ods)
- Desde PHP myadmin primero seleccionas la base desde la que quieras importar>pestaña importar> te aseguras que está elegida ods y el primer checkbox.
- Después adaptamos el código anterior a la nueva tabla, **Pruebas**, y además metemos los datos en una tabla con filas y columnas

```
$consulta="SELECT * FROM PRODUCTOS";

$resultados=mysqli_query($conexion, $consulta);

while($fila=mysqli_fetch_row($resultados)){
    echo "<table><tr><td>";
    echo $fila[0] . "</td><td> ";
    echo $fila[1] . "</td><td> ";
    echo $fila[2] . " </td><td>";
    echo $fila[3] . " </td><td>";
    echo $fila[4] . " </td><td>";
    echo $fila[5] . " </td><td>";
    echo $fila[6] . " </td><td></tr></table>";
    echo "<br>";
    echo "<br>";

}

mysqli_close($conexion);
```

- A continuación le damos estilo en el head a través de etiqueta style

```
<style>

    table{
        width:50%;
        border:1px dotted #FF0000;
        margin:auto;
    }

</style>
```

- e. Y por ultimo creamos la instrucción

```

mysqli_select_db($conexion, $db_nombre) or die ("No se encuentra la BBDD");

mysqli_set_charset($conexion, "utf8");

$consulta="SELECT * FROM PRODUCTOS WHERE PAÍSDEORIGEN='ESPAÑA'";

$resultados=mysqli_query($conexion, $consulta);

while($fila=mysqli_fetch_row($resultados)){

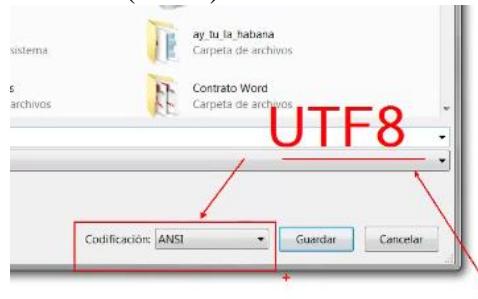
```

Además de una base de datos de gran volumen, podemos importar datos para la consola

1. Desde el bloc de notas con una instrucción SQL que nos agregue registros dentro de una tabla.
- ```

CREATE TABLE PRODUCTOS3 (CODIGOARTICULO VARCHAR(4), SECCION VARCHAR(11), NOMBREARTICULO VARCHAR(20));
INSERT INTO PRODUCTOS3 (CODIGOARTICULO, SECCION, NOMBREARTICULO)
VALUES
('ARO1', 'DEPORTES', 'RAQUETA'),
('ARO2', 'FERRETERIA', 'CHINCHETA');

```
2. Lo guardamos con la extensión sql. Eje *insruccions.sql*
  3. Y también cambiamos la codificación UTF8 por la que viene por defecto en bloc de notas (ANSI)



4. Ahora puedes importarlo. De dos maneras, como archivo sql o como odt

Esto nos permite exportar una tabla de una base de datos a otra

1. Seleccionamos la tabla
2. Le damos a la pestaña importar
3. Y le decimos que nos exporte como sql
4. Si este archivo lo localizamos en la carpeta exportada, y lo abrimos con el bloc de notas, veremos que en su interior es una instrucción sql (si lo abrimos con DW veremos los datos mucho más ordenados y claros)
5. Nos vamos abora a otra base de datos,
6. Le damos a importar
7. Importamos el anterior archivo exportado , como sql y en codificación utf8

## COMO CREAR UNA PÁGINA WEB DE BÚSQUEDA

En el ejercicio anterior utilizábamos la función mysqli\_fetch\_row para que nos devuelva fila a fila los registros de una tabla, mediante arrays indexados.

Ahora vamos a trabajar con otra función parecida, que en lugar de trabajar con arrays indexados trabaja con arrays asociativos. Que en lugar de acceder a un índice accede a un nombre de campo: mysqli\_fetch\_array()

Hacemos el mismo ejercicio anterior pero ahora con esta función.

Esta función nos pide dos parámetros: el resultado (almacenado en nuestro ejemplo en la variable resultados) y una constante MYSQL\_ASSOC (constante que indica que queremos trabajar con un Array asociativo). Trabajamos con el nombre de los campos

Ejemplo: buscar código artículo, nombre artículo, y país de origen.

```
$resultados=mysqli_query($conexion, $consulta);

while($fila=mysqli_fetch_array($resultados, MYSQL_ASSOC)) {
 echo "<table><tr><td>";
 echo $fila['CÓDIGOARTÍCULO'] . "</td><td> ";
 echo $fila['NOMBREARTÍCULO'] . "</td><td> ";
 echo $fila['PAÍSDEORIGEN'] . " </td><td></tr></table>";

 echo "
";
 echo "
";
```

Esto equivale a mysqli\_fetch\_assoc()

por otro lado, para poder hacer una página de búsqueda en condiciones, deberemos trabajar con los caracteres comodín. Tenemos dos caracteres comodín: El % y el guion bajo (\_)

El carácter comodín porcentaje, se dice que sustituye a una cadena de caracteres

El guion bajo sustituye a un único carácter

Ejemplo, en nuestra tabla tenemos, en campo NOMBRE ARTÍCULO, uno que es TRAJE DE CABALLERO, y otro que es CAMISA DE CABALLERO

Queremos hacer una consulta que devuelva todos los artículos cuyo último nombre acabe en CABALLERO. Si en el criterio de la búsqueda SQL ponemos %caballero, le estamos diciendo a la base de datos que queremos ver todas las datos que terminan en caballero, porque le estamos diciendo que antes de el nombre CABALLERO, puede ir precedido de una o muchas caracteres.

Si lo colocamos al final (En caso de que queramos seleccionar todos los balones, los datos balón rugby, balón baloncesto baloncesto.....) tendríamos que poner el porcentaje al final.

Para poder utilizar caracteres comodín con consultas MySQL, debemos utilizar el operador LIKE. Vamos a verlo

```
LECT * FROM PRODUCTOS WHERE NOMBREARTÍCULO LIKE '%CABALLERO'
```

Imaginemos que en el campo hay varios registros CENICERO, yo quiero seleccionar todos, pero se que algunos están mal escritos: CENICERO. Si hacemos una consulta normal, deberemos escribir cenicero, y no saldrán solo los que están bien escritos; o CENIZERO y nos saldrán lo que están bien.

Pero si queremos sacar ambos, escribiríamos

```
NOMBREARTÍCULO LIKE 'CENI_ERO'
```

En este caso nos devuelve tanto el que está bien escrito como el que está mal

Con estos conceptos sentamos las bases para sentar una página de búsqueda

## VAMOS A CREAR NUESTRA PRIMERA PÁG DE BÚSQUEDA PARTIMOS DE UN FORMULARIO MUY SENCILLO

Buscar:  Dale!

Por ejemplo queremos buscar los registros que contengan la palabra cenicero. Lo ponemos en el recuadro, y luego le damos a DALE,

```
>

<form action="pagina_busqueda.php" method="get">
 <label>Buscar: <input type="text" name="buscar"></label>
 <input type="button" name="enviando" value="Dale!">
</form>
>
>
```

OJO!! no es un button es un submit

Ahora vamos a crear pagina\_busqueda.php.

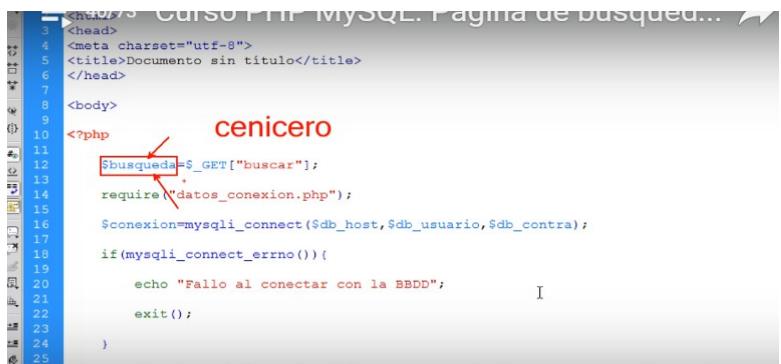
### 1. Creamos esta página php

```
<?php
$busqueda=$_GET["buscar"];
require("datos_conexion.php");
$conexion=mysqli_connect($db_host,$db_usuario,$db_contra);
if(mysqli_connect_errno()){
 echo "Fallo al conectar con la BBDD";
 exit();
}

mysqli_select_db($conexion, $db_nombre) or die ("No se encuentra la BBDD");
mysqli_set_charset($conexion, "utf8");
$consulta="SELECT * FROM PRODUCTOS WHERE NOMBREARTICULO LIKE '%$busqueda%'";
$resultados=mysqli_query($conexion, $consulta);
while($fila=mysqli_fetch_array($resultados, MYSQL_ASSOC)){
 echo "<table><tr><td>";
 echo $fila['CÓDIGOARTÍCULO'] . "</td><td> ";
 echo $fila['NOMBREARTICULO'] . "</td><td> ";
 echo $fila['SECCIÓN'] . "</td><td> ";
 echo $fila['IMPORTADO'] . "</td><td> ";
 echo $fila['PRECIO'] . "</td><td> ";
 echo $fila['PAÍSDEORIGEN'] . " </td><td></tr></table>";

 echo "
";
}
1K / 15 Unicode (UTF-8)
```

2. Tenemos en cuenta como hemos llamado el name de ese cuadro de texto buscar : lo hemos llamado buscar
3. al principio del código php creamos la variable \$busqueda, y almacenamos en ella lo que el usuario haya introducido en el cuadro de texto buscar, mediante la función \$\_GET["buscar"]



The screenshot shows a browser window with the URL "http://127.0.0.1/curso/III/MySQL\_Página de búsqueda/index.php". The page title is "Documento sin título". The content area displays the word "cenicero" in red, indicating it is the search term. Below the search term, there is some placeholder text: "No se encontraron resultados para la búsqueda: 'cenicero'". The code in the browser's developer tools shows a line of PHP code with a red box highlighting the variable \$busqueda, which is assigned the value from the GET parameter "buscar".

```
<?php
$busqueda=$_GET["buscar"];
```

4. y una vez que tenemos almacenado en esta variable lo que ha escrito el usuario, es el siguiente paso sería irnos a la secuencia SQL y asociamos la variable con la consulta, de esta manera lo que el usuario meta en el cuadro será la consulta.

```
21 exit();
22
23 }
24
25 mysqli_select_db($conexion, $db_nombre) or die ("No se encuentra la BBDD");
26
27 mysqli_set_charset($conexion, "utf8");
28
29 $consulta="SELECT * FROM PRODUCTOS WHERE NOMBREARTÍCULO='$busqueda'";
30
31 $resultados=mysqli_query($conexion, $consulta);
32
33
34
```

Comprobar que funciona

Para que además cuando un usuario introduzca un contenido como CABALLERO y le salgan todos los elementos: TRAJE CABALLERO; PANTALON CABALLERO..... debemos poner en SQL y antes de la variable \$busqueda, un % asi y sustituir el igual por LIKE

**LIKE %\$busqueda**

Pero con esta sentencia si ponen balón, no devuelve nada , por lo que añadimos % detrás para que coja todas las búsquedas

**LIKE %\$busqueda%**

Vamos a ver otra forma de hacer una página de búsqueda. Hasta ahora lo hemos hecho en dos archivos diferentes.

Ahora vamos a unir el formulario de búsqueda y los resultados en la misma página

Reutilizamos prácticamente todo el código anterior

Creamos un nuevo archivo php que une el formulario de búsqueda y la página de resultado. Le llamamos búsqueda\_resultados.php

1. Cogemos todo el código de la página\_busqueda creada en el doc anterior y lo pego en el head de mi nuevo documento: De esta manera me aseguro que al cargar la página va a leer el php incluso antes de leer lo que haya en el body
2. Modificaciones:
  - a. Variable búsqueda: al no proceder ya de una página diferente, Sino de la misma. Por lo que eliminamos la instrucción `$_GET`. Por lo que eliminamos o comentamos toda la variable `$busqueda`
  - b. Además el código php, aunque efectivamente tiene que ser formulario, por lo que debemos introducir todo esto en una función. (Creamos función ejecuta\_consulta y con un parámetro llamado `$labusqueda`), y metemos todo el código.
  - c. Por lo que en la consulta SQL también debemos cambiar el `$busqueda` por `%%$labusqueda%` (parámetro de la función)
3. Ahora creamos el formulario dentro de bloque php:
  - a. Creamos una variable `$mibusqueda` donde almacenamos un dato de un archivo html llamado "buscar", con la instrucción `$_GET`.
  - b. Creamos una nueva variable llamada `$mipag`, donde vamos a almacenar la instrucción necesaria para que se envíe la información a esa misma página, no a una diferente. Con `$_SERVER` indicamos cual es la página del servidor a la que quiero enviar la información. Sería a ella misma con instrucción `PHP_SELF`.
  - c. Ahora le digo a nuestra web (con un if), que si la variable "mi búsqueda" es diferente de NULO, entonces que ejecute la consulta y si no que muestre el formulario. Así si es diferente a NULL se ejecuta la función ejecuta\_consulta. Esta función recibía un parámetro que es lo que hay almacenado en la variable `$mibusqueda`.
- En caso contrario constrúyeme un formulario que me envíe información a la misma página en la que me encuentro.
- d. En el action debemos decir que envíela página al mismo sitio donde la tenemos almacenada. (`$mipag`)

```
<?php

function ejecuta_consulta($labusqueda) {
 // $busqueda=$_GET["buscar"];
 require("datosConexion.php");
 $conexion=mysqli_connect($db_host,$db_usuario,$db_contra);
 if(mysqli_connect_errno()){
 echo "Falló al conectar con la BBDD";
 exit();
 }
 mysqli_select_db($conexion,$db_nombre) or die ("No se encuentra la BBDD");
 mysqli_set_charset($conexion, "utf8");
 $consulta="SELECT * FROM PRODUCTOS WHERE NOMBREARTICULO LIKE '%$labusqueda'";
 $resultados=mysqli_query($conexion,$consulta);
 while($fila=mysqli_fetch_array($resultados, MYSQL_ASSOC)){
 echo "<table><tr><td>";
 echo $fila['CÓDIGOARTÍCULO'] . "</td><td>";
 echo $fila['NOMBREARTÍCULO'] . "</td><td> ";
 echo $fila['SECCIÓN'] . "</td><td> ";
 echo $fila['IMPORTE'] . "</td><td> ";
 echo $fila['PRECIO'] . "</td><td> ";
 echo $fila['PAÍSDEORIGEN'] . " </td><td></tr></table>";
 echo "
";
 echo "
";
 }
 mysqli_close($conexion);
}

head>
```

```

<?php

$mibusqueda=$_GET["buscar"];

$mipag=$_SERVER["PHP_SELF"];

if ($mibusqueda!=NULL) {

 ejecuta_consulta($mibusqueda);

} else{

 echo("<form action='" . $mipag . "' method='get'>

<label>Buscar:<input type='text' name='buscar'></label>

<input type='submit' name='enviando' value='Dale!'>

</form>");

}

```

Repasamos: Al cargar la pagina, esta función se lee pero no e ejecuta hasta que no es llamada

Pasa directamente a body .

En el body nos dará un error en local (en remoto no) porque no encuentra el name buscar de \$-GET.

Almacena dentro de \$mipag la página del servidor que tiene que llamar al formulario que va a ser ella misma.

Que si la búsqueda es diferente de NULL, que ejecute la consulta

Si no crea un formulario que llama a la pagina del servidor que es ella misma con método get.

La primera vez que lea la pagina búsqueda =NULL no se va a almacenar nada ; El término buscar no va a saber a qué se refiere.

Luego la primera vez no entrará en if, sino que entra en else

En el else encontramos la construcción de un formulario, cuya acción es llamar a la pag del servidor que es ella misma.

Con el método get para pasar la información a través del URL

Una casilla con name buscar y

Botón que al ser submit lo que hace es recargar la página.

¿qué pasa pues que ahora al recargar la pagina, en la casilla búsqueda ya hay información

al recargar la pagina se vuelve a tener el código de body; y al leerlo por 2> vez la variable \$mibusqueda ya va a ser capaz de almacenar esta información (la información almacenada es la que se haya escrito en input type text después almacena la propia pagina en la variable \$mipag, y en esta ocasión entra en el if y no en el else.

Porque \$mibusqueda ya no es nulo, así que pasará esta información a la URL y ejecutará esta función del if

Al ejecutar la función le pasamos por parámetro lo que hay almacenado en la búsqueda que es el término de búsqueda

```
INSERT INTO TABLA (CAMPO1, CAMPO2, CAMPO3) VALUES (VALOR1, VALOR2, VALOR3)
```



## INSERTAR REGISTROS EN BBDD DESDE UN FORMULARIO WEB

Insertamos en la tabla de productos unos registros nuevos

Vamos a hacer esto mismo desde una página php. ¿Cómo hacemos para que desde una página web se introduzca una información en una base de datos?

1. Creamos una página php como insertar\_registro.php

Con el siguiente código:

require :vinculamos con los datos de conexión.

Luego conectamos con la base de datos: usuario y contraseña

El if con la función mysqli para el caso en que haya un error de conexión con el servidor

A continuación lo mismo para el error de base de datos

Y el charset

A continuación cambiamos la instrucción SQL en una variable \$consulta

```
'P''V

//$busqueda=$_GET["buscar"];

require ("datos_conexion.php");

$conexion=mysqli_connect($db_host,$db_usuario,$db_contra);

if(mysqli_connect_errno()){

echo "Falló al conectar con la BBDD";

exit();

}

mysqli_select_db($conexion,$db_nombre) or die ("No se encuentra la BBDD");

mysqli_set_charset($conexion, "utf8");

$consulta="INSERT INTO PRODUCTOS (CÓDIGOARTÍCULO, SECCIÓN, NOMBREARTÍCULO) VALUES ('AR44', 'DEPORTES', 'RAQUETA BADMINTON')"

$resultados=mysqli_query($conexion,$consulta);

mysqli_close($conexion);
```

Si ejecutamos esto, se debe haber introducido en la base de datos los nuevos registros (aunque ojo!. Al hacer f12 nos sale una página en blanco)

2. El siguiente paso será crear un formulario , ser capaces de introducir la información que aparezca en la base de datos, en el formulario

Creamos el html(unas tablas) y el css, vamos poniendo nombres a cada cuadro

```

</style>
</head>

<body>
<h1>Registro de Artículos</h1>
<form name="form1" method="get" action="Insertar_Registro.php">
 <table border="0" align="center">
 <tr>
 <td>Código Artículo</td>
 <td><label for="c_art"></label>
 <input type="text" name="c_art" id="c_art"></td>
 </tr>
 <tr>
 <td>Sección</td>
 <td><label for="seccion"></label>
 <input type="text" name="seccion" id="seccion"></td>
 </tr>
 <tr>
 <td>Nombre artículo</td>
 <td><input type="text" name="nombre_art" id="nombre_art"></td>
 </tr>
 <tr>
 <td>Precio</td>
 <td><input type="text" name="precio" id="precio"></td>
 </tr>
 <tr>
 <td>Fecha</td>
 <td><input type="text" name="fecha" id="fecha"></td>
 </tr>
 <tr>
 <td>Importado</td>
 <td><input type="checkbox" name="importado" id="importado"></td>
 </tr>
 <tr>
 <td>País de Origen</td>
 <td><input type="text" name="pais_origen" id="pais_origen"></td>
 </tr>
 </table>
</form>

```

3. Seguimos con el documento introducir\_registros.php

Los values tenemos que modificarlos. NO podríamos dejar los valores fijos porque se introducirían siempre los mismos.

- a. Se trata de que estos valores tenemos que extraerlos del formulario anterior.  
Para pasar los del documento formulario a el documento insertar registro, lo haremos mediante `$_GET`.
- b. Por otro lado necesitamos que esos valores que rescatamos del formulario, se almacenen en los value.  
para ello creamos un montón de variables donde se almacene esa información.
  - b.i. Primero la variable cod que va a ser donde almacenemos el código del artículo, mas la función `$_GET` y luego el nombre del cuadro de texto;

a continuación almacenamos en la variable sec, la sección; luego el nombre; el precio....; fecha; campo importado; y la ultima origen

```
//$busqueda=$_GET["buscar"];
$codigo=$_GET["c_art"];
$seccion=$_GET["seccion"];
$nombre=$_GET["n_art"];
$precio=$_GET["precio"];
$fecha=$_GET["fecha"];
$importe=$_GET["importado"];
$pais=$_GET["p_orig"];
require("datos_conexion.php");
$conexion=mysqli_connect($db_host,$db_usuario,$db_contra);
if(mysqli_connect_errno()){
 echo "Fallo al conectar con la BBDD";
 exit();
}
mysqli_select_db($conexion,$db_nombre) or die ("No se encuentra la BBDD");
mysqli_set_charset($conexion, "utf8");
$consulta="INSERT INTO PRODUCTOS (CÓDIGOARTÍCULO, SECCIÓN, NOMBREARTÍCULO, PRECIO, FECHA, IMPORTADO, PAÍSDEORIGEN) VALUES
```

b.ii.

Con esto ya tenemos almacenado en unas cuantas variables , los datos que vienen de la otra página. Gracias al \$\_GET, rescatamos los valores que tenemos de la otra pagina

b.iii. Ahora modificamos el INSERT INTO :

b.iii.1. Primero añadimos todos los campos de la tabla

b.iii.2. Luego añadimos todos los campos fijos del value por las variables correspondientes



Esto nos introduce todos los registros en la base de datos pero conviene que nos salga un mensaje de que se ha realizado correctamente. por lo que se introduce el siguiente if

```

 mysqli_set_charset($conexion, "utf8");

 $consulta="INSERT INTO PRODUCTOS (CÓDIGOARTÍCULO, SECCIÓN, NOMBREARTÍCULO, PRECIO, F
 ('$cod', '$sec', '$nom', '$pre','$fec','$imp','$por')";

 $resultados=mysqli_query($conexion,$consulta);

 if($resultados==false){

 echo "Error en la consulta";

 }else{

 echo "Registro guardado";
 I
 }

```

Mejoramos el echo, registro guardado, para que nos de la información guardada

```

 }

 echo "Registro guardado

";

 echo "<table><tr><td>$cod</td></tr>";
 echo "<tr><td>$sec</td></tr>";
 echo "<tr><td>$nom</td></tr>";
 echo "<tr><td>$pre</td></tr>";
 echo "<tr><td>$fec</td></tr>";
 echo "<tr><td>$imp</td></tr>";
 echo "<tr><td>$por</td></tr></table>";

```

De modo que nos salga lo siguiente



## ELIMINANDO REGISTROS DE UNA BASE DE DATOS DESDE UN FORMULARIO HEMOS VISTO:

- COMO CONSULTAR UNA INFORMACIÓN DE UNA BASE DE DATOS
- COMO INSERTAR NUEVOS REGISTROS
- AHORA VAMOS A VER COMO ELIMINAR REGISTROS

¿CÓMO

CON UNA INSTRUCCIÓN SQL NUEVA: DELETE FROM TABLA WHERE  
CAMPO=CRITERIO

Repasamos el ejercicio anterior y lo reutilizamos

- Copiamos el formulario de registro anterior y le renombramos poniendo formulario eliminar. Y el contenido en lugar de poner registro de artículos le vamos a llamar eliminación de artículos, y el botón en lugar de enviar sea eliminar
- Y hacemos lo mismo con el archivo insertar registro anterior y lo renombrados como eliminar\_registro PHP

Normalmente cuando utilizamos un formulario para eliminar registros deberíamos usar solo el formulario un campo que suele ser un campo clave ( en el ejemplo el código artículo)

El campo clave es aquel que no se repite, que es único para cada registro.

Si utilizamos el resto de campo también va a funcionar pero no es lo que se suele hacer, porque si le decimos que elimine sección deportes nos eleminaria de una tacada todos los artículos de esta sección

Pasos:

1. En el action ponemos eliminar registros
2. En el archivo php de eliminar registro hacemos los siguientes cambios:
  - a. Cambiamos la instrucción SQL: quitamos Insert Into y lo sustituimos por DELETE

```
DELETE FROM PRODUCTOS WHERE CÓDIGOARTÍCULO='&cod';
```

- b. Hacemos un if para que el caso en que haya un error avise, y si ha salido correctamente diga "registro eliminado". Lo probamos y vemos como efectivamente nos devuelve "registro eliminado"

¡OJO! Si decimos de eliminar un archivo que no existe dará también "registro eliminado", para evitar esto, utilizamos la función MySQLi\_affected\_rows (conexión)

pide como parámetro la conexión a la base de datos y lo que hace es informarnos de cuantas filas o registros de la base de datos, se han visto afectados de tipo INSERT,(insertr) de tipo DELETE,(borrar) o de tipo UPDATE(actualizar).

3. Cambiaríamos el else por esto

```
//echo mysqli_affected_rows($conexion);

if (mysqli_affected_rows($conexion)==0){

 echo "no hay registros que eliminar con ese criterio";

}else{

 echo "Se han eliminado " . mysqli_affected_rows($conexion) . " registros";
}

]
```

Solo nos queda ver como actualizar los registros



Imaginemos que tenemos que subir 100 euros todos los artículos toda la tabla de productos. Entonces sería

UPDATE productos SET PRECIO= precio + 100

otro ejemplo

vamos a modificar el nombre martillo, por "el martillo":

- primera

Ejecutar la(s) consulta(s) SQL en la base de datos pruebas:

```
1 UPDATE PRODUCTOS SET NOMBREARTÍCULO='EL MARTILLO' WHERE CÓDIGOARTÍCULO='AR22'
```

- otro ejemplo

la llave inglesa quiero que pase a costar 34,40. Incremento de 10 euros. Si lo tengo en formato texto, tendré que poner el nuevo precio entre comillas. SI fuera un campo numérico, tendríamos que hacer esto PRECIO=PRECIO+10. A continuación cláusula where nombre articulo="llave inglesa"

Ejecutar la(s) consulta(s) SQL en la base de datos pruebas:

```
1 UPDATE PRODUCTOS SET PRECIO='34,40' WHERE NOMBREARTÍCULO='LLAVE INGLESA'
```

si quisiéramos cambiar dos datos de un registro. Ej en traje de caballero queremos modificar por un lado el precio por otro el país de origen

Ejecutar la(s) consulta(s) SQL en la base de datos pruebas:

```
1 UPDATE PRODUCTOS SET PRECIO='350', PAÍSDEORIGEN='ESPAÑA' WHERE CÓDIGOARTÍCULO='AR02'
```