MySQL & PHP

eamexicano

Conceptos

Clase 01

eamexicano

Aplicación, Sitio, Página

Página: Documento que puede ser visualizado en un navegador web. El documento puede incluir la manera en que va a ser presentado (estilo del documento) así como la manera de responder a ciertos eventos del usuario (scripts).

Sitio: Conjunto de páginas web relacionadas.

Aplicación: software que puede ser utilizado a través de internet (generalmente con un navegador web).

Lenguajes más utilizados para aplicaciones web

HTML	CSS	JS	PHP	SQL*
Estructura	Presentación	Interacción	Plantillas Conexión	Almacenaje

* SQL como lenguaje - Structured Query Language.

Aunque cada lenguaje tiene una funcionalidad en particular hay ocasiones en que implementa (en conjunto con otros lenguajes) funciones comunes.

Al tener los lenguajes separados y utilizandolos para un propósito en particular resulta más fácil prestar atención a los detalles para la implementación y solucionar algunos problemas que se puedan presentar.

Front End - aplicaciones web

HTML	CSS	JS	PHP	SQL
Estructura	Presentación	Interacción	Plantillas Conexión	Almacenaje

No se necesita un servidor web para desarrollar páginas o sitios que no tienen backend.

“En diseño de software el front-end es la parte del software que interactúa con el o los usuarios...”

http://es.wikipedia.org/wiki/Front-end_y_back-end

Back End - aplicaciones web

HTML	CSS	JS	PHP	SQL*
Estructura	Presentación	Interacción	Plantillas Conexión	Almacenaje

Es necesario configurar una máquina de desarrollo con los programas básicos que se van a utilizar en el sitio en vivo.

La configuracíon del servidor en el que va a ser implementado puede variar.

"...el back-end es la parte que procesa la entrada desde el front-end"

http://es.wikipedia.org/wiki/Front-end_y_back-end

LAMP

Aunque en el stack de LAMP hay 4 componentes
(SO, SW, LP, DBMS) los últimos dos es donde vamos a pasar mayor tiempo.

-> Iniciar por mysql,

-> PHP lo vamos a utilizar como vinculo entre mysql y el usuario.

eamexicano

LAMP / MAMP / WAMP / XAMP

Es un stack (conjunto) de programas que se utilizan en servidores web (enfocados a aplicaciones web).

Está formado por un sistema operativo, servidor web, lenguaje de programación, administrador de bases de datos.

El nombre viene por las siglas de la configuración que se utiliza mas frecuentemente:

Linux

Apache

MySQL

PHP

Dependiendo del S.O. también se le puede encontrar como:

WAMP - Windows
MAMP - Mac (OS X)
LAMP - Linux / Unix
XAMPP - X = Cualquier plataforma

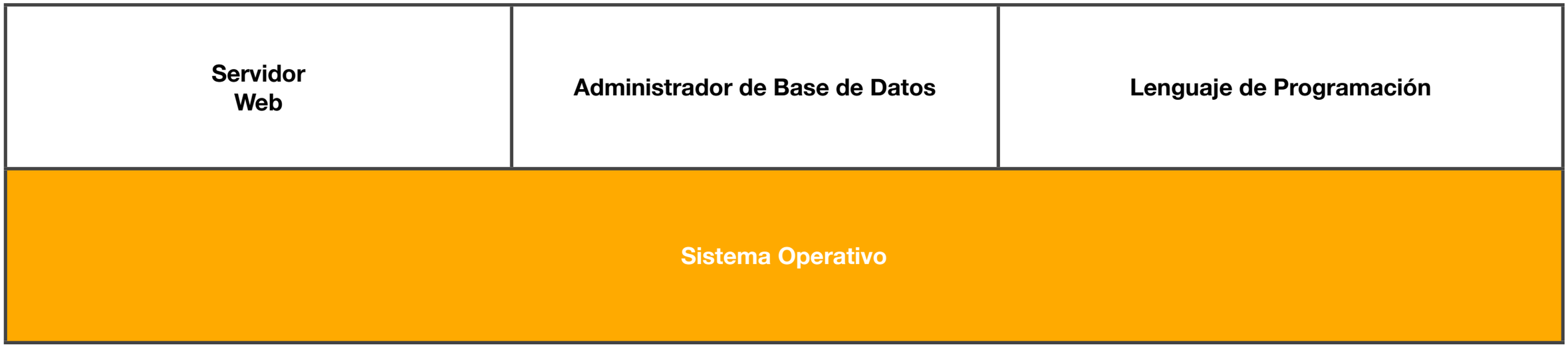
LAMP Stack

Servidor Web	**Administrador de Base de Datos**	**Lenguaje de Programación**
Sistema Operativo		

Las aplicaciones web generalmente utilizan un servidor de correo (postfix / sendmail).

Aunque no se muestra en el diagrama quedaría como una capa sobre el S.O.

LAMP Stack - Sistema Operativo



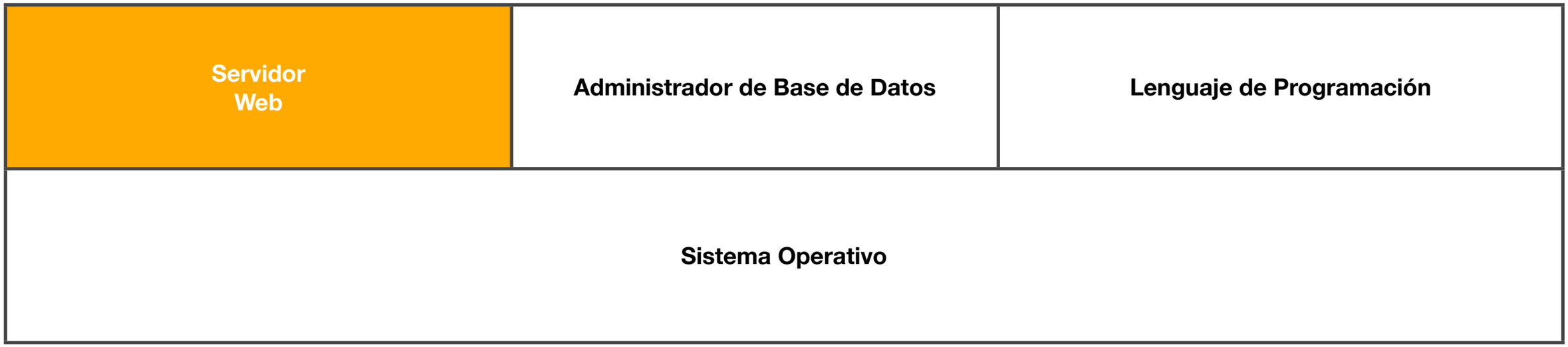
El sistema operativo es la base sobre la cuál está la aplicación.
Algunas de las tareas que se toman en cuenta son:

Configuración de cuentas de usuario para dar acceso al servidor.
Permisos para instalar / desinstalar software.
Configurar programas de conexión (FTP / SSH).

Si el hospedaje es administrado por un tercero ubicar:

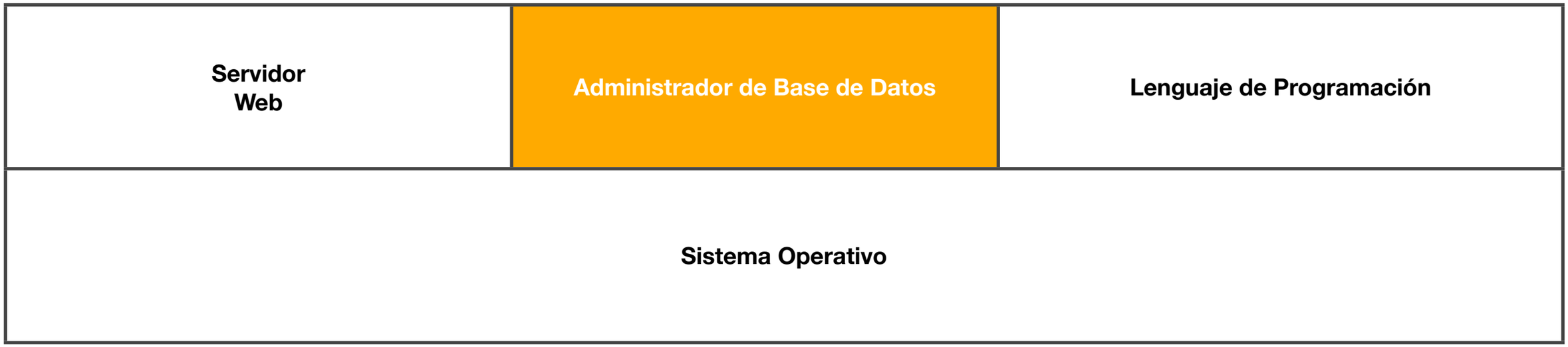
- La carpeta htdocs (o la carpeta donde vayan a estar los proyectos web).
- Acceso a la BD (usuario, contraseña y software de administración disponible)

LAMP Stack - Servidor Web - Apache



El servidor web se encarga de enviar los archivos (html, css, js) al usuario que lo solicite.
Si se trata de una aplicacón donde exista código, el servidor web busca el archivo y lo manda a interpretar con el lenguaje de programación correspondiente (PHP). Una vez que PHP procesa el archivo lo devuelve al servidor web y este lo envía al usuario.

LAMP Stack - DBMS - MySQL



El administrador de bases de datos - DBMS - Database Management System - funciona como un almacén de datos.

LAMP Stack - Lenguaje de Programación - PHP

<table>
<tr><td>Servidor
Web</td><td>Administrador de Base de Datos</td><td>Lenguaje de Programación</td></tr>
<tr><td colspan="3">Sistema Operativo</td></tr>
</table>

En este caso el lenguaje de programación se va a utilizar como conexión entre la BD y la interfaz gráfica.

Conexión a la BD.
Lógica de la aplicación.
Presentación de datos.

Instalar XAMPP

XAMPP - Linux, Mac OS X, Windows

MAMP - Mac OS X

WAMP - Windows

Consideraciones

Dependiendo de la versión de XAMPP* es posible que la configuración cambie:
- Versión de cada uno de los componentes del stack.
- Versión de PHPMyAdmin (se refleja en la interfaz web).
- Tener puertos distintos a los bien conocidos.
- MySQL tenga una contraseña predeterminada.

* XAMPP, MAMP (Mac OS), WAMP (Windows) o instalación independiente de software.

Mac OS

Si se está utilizando “Compartir Web” es posible que el servidor apache de XAMPP no pueda iniciar.
Hay que deshabilitar “Compartir Web” mientras se trabaja con XAMPP.

MySQL

eamexicano

Conceptos Generales

MySQL es un administrador de bases de datos (DBMS - sigas en inglés).

Un administrador de BD se encarga de organizar de las BD, los usuarios (accesos a la BD), modificar el uso de recursos del servidor.

Una BD es un conjunto de datos pertenecientes a un mismo contexto.

¿Para qué sirven las Bases de Datos?
Para tener datos almacenados los cuales puedan ser accesibles de manera fácil para su uso.

No todas las BD son electrónicas.

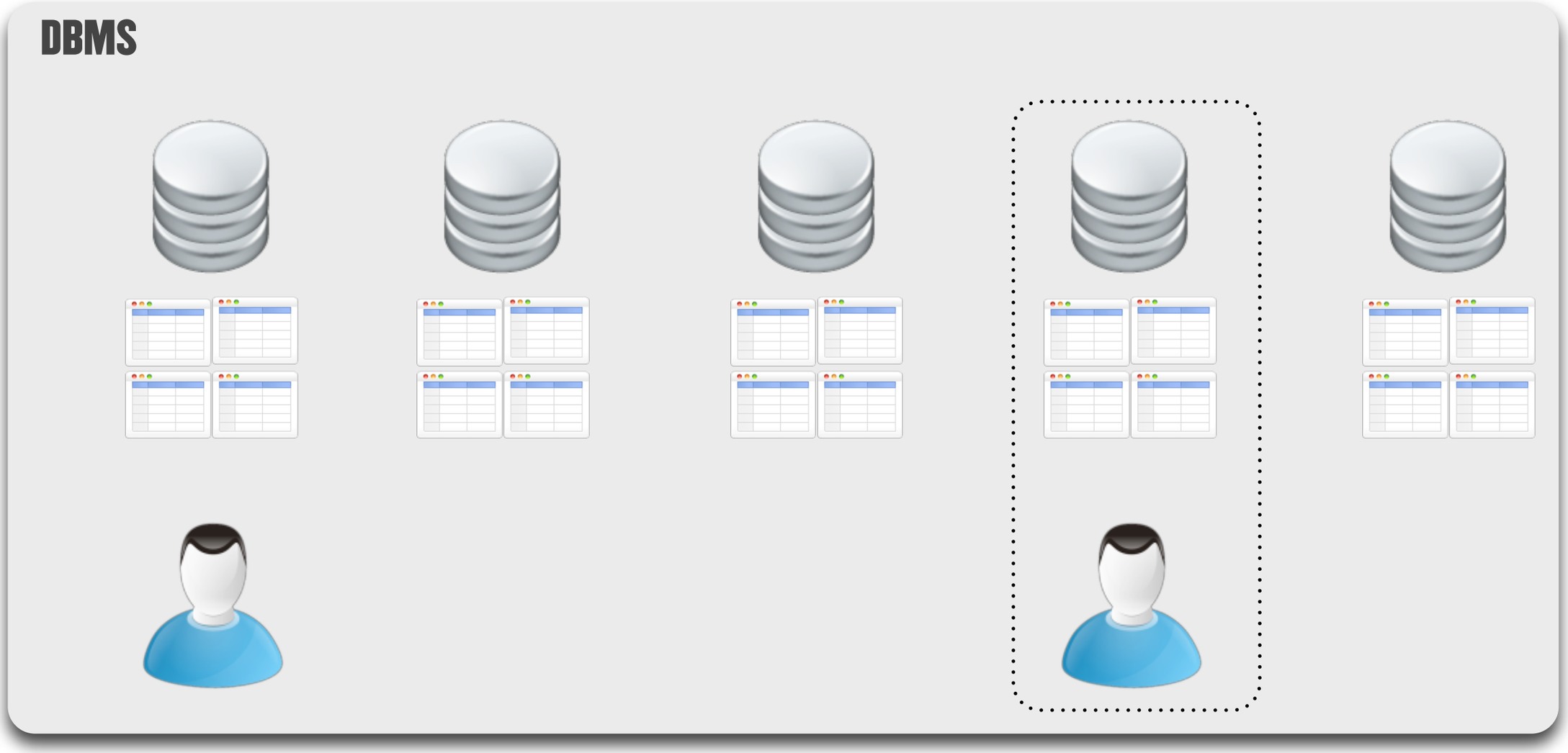
Ejemplo de BD no electrónicas:

Catálogo de libros o de una biblioteca en tarjetas impresas.
Directorio telefónico impreso.

Ejemplos de BD electrónicas:
Contactos del teléfono.
Lista de canciones del reproductor mp3 (iTunes).

Algunos ejemplos de BD en internet son las redes sociales (facebook, twitter), blogs (blogger, tumblr), tiendas en linea (amazon, barnesandnoble, etsy), gobierno (SAT, Finanzas, tesorería).

Representación de DBMS

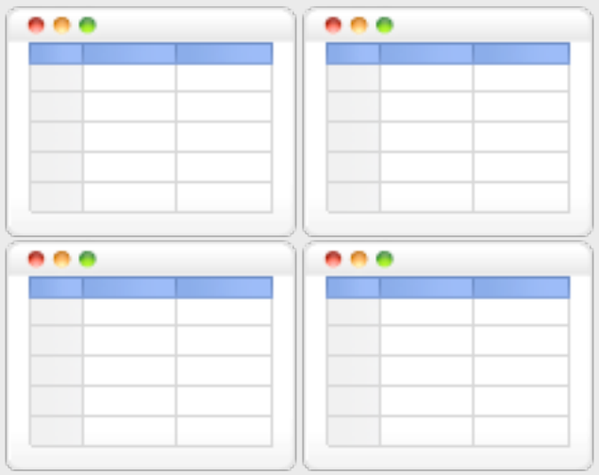


Representación de bd, tabla, columna, fila

BD

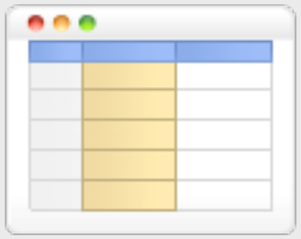
La BD es un contenedor. Ese contenedor tiene otros contenedores llamados tablas donde van a estar almacenados los datos.
Este contenedor sirve para dar contexto general a los datos almacenados en las tablas. Podemos tener una tabla de usuarios en una tienda y en una lista de pendientes.

Tablas



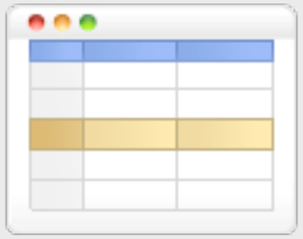
En cada tabla se almacenan los datos pero antes se tiene qué definir ¿qué datos se van a almacenar?
Al crear la tabla se definen los datos que se van a almacenar utilizando columnas.

Columnas (atributos)



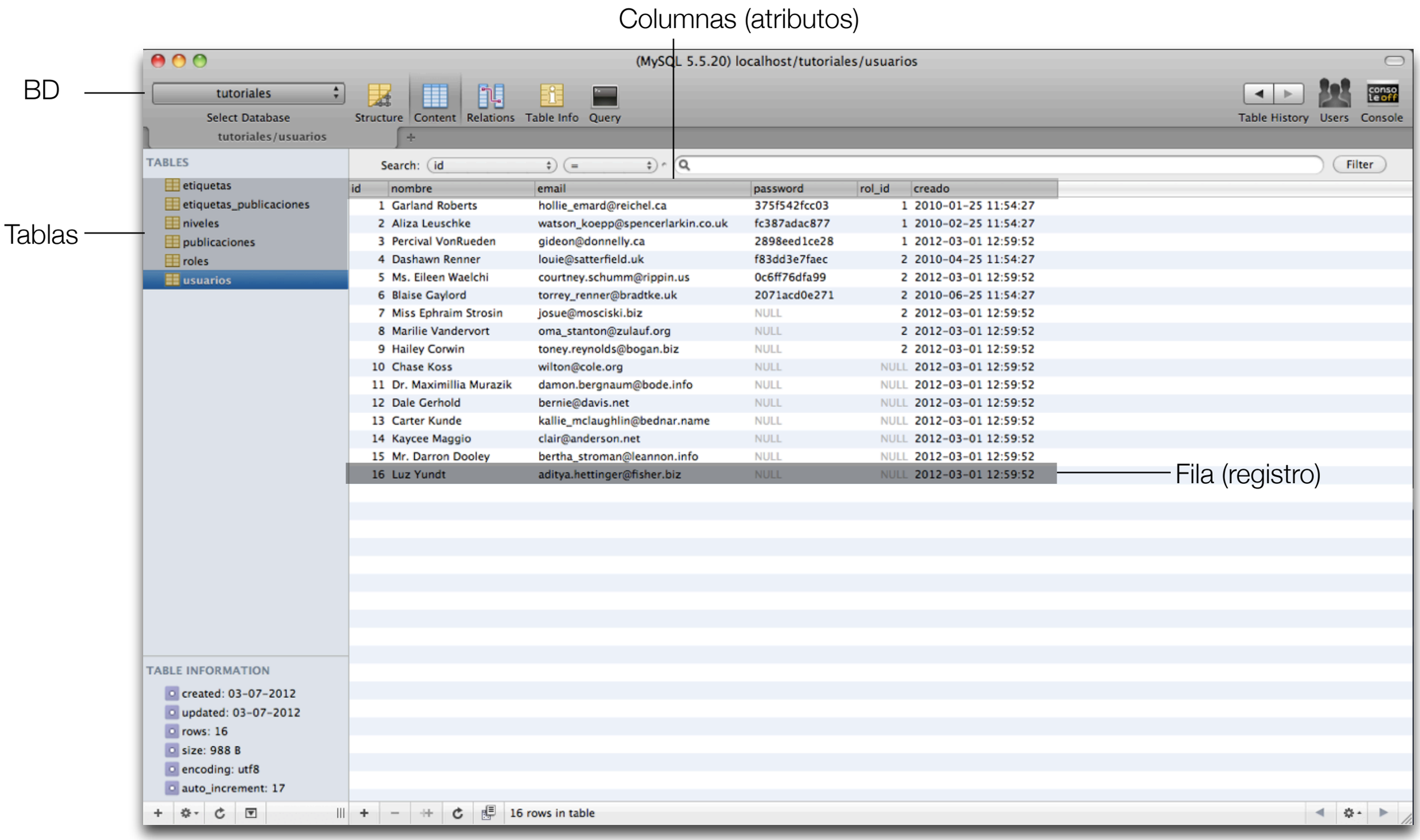
Cada columna representa un atributo que quiere ser almacenado.
Se definen con un nombre (que representa el dato que va a estar almacenado) y cómo va a estar almacenado (texto, número, archivo, etc).

Filas (registros)

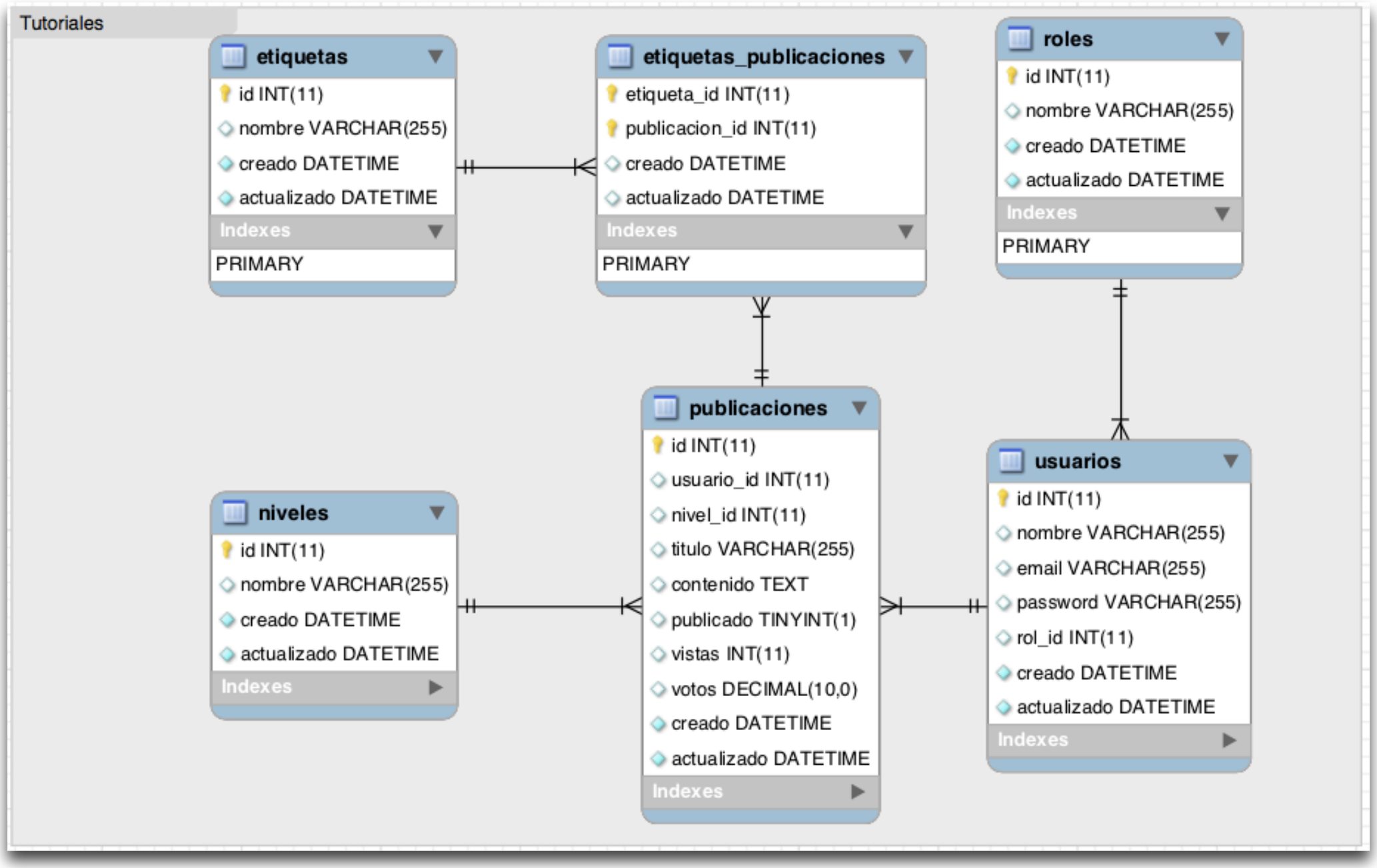


Cada fila o registro es un conjunto de datos almacenados que representan una entidad (usuario, libro, orden de compra, etc).

Visualización de bd, tabla, columna, fila



MySQL - BD Relacional



Se llama relacional porque los datos se guardan en tablas diferentes.
Para asociar los datos entre las distintas tablas, algunos valores de una columna en una tabla se repiten en una columna en otra tabla.

Identificadores

Los identificadores son columnas cuya función principal es:

Diferenciar cada registro.

Cada registro (fila) en las tablas debe ser distinto. Para lograr esto, se utiliza una columna en la cual los valores no se pueden repetir. A esta columna se le conoce como Llave Primaria (PK).

PK - Clave Primaria: Identifica de manera única cada registro en una tabla. Su valor no se puede repetir.

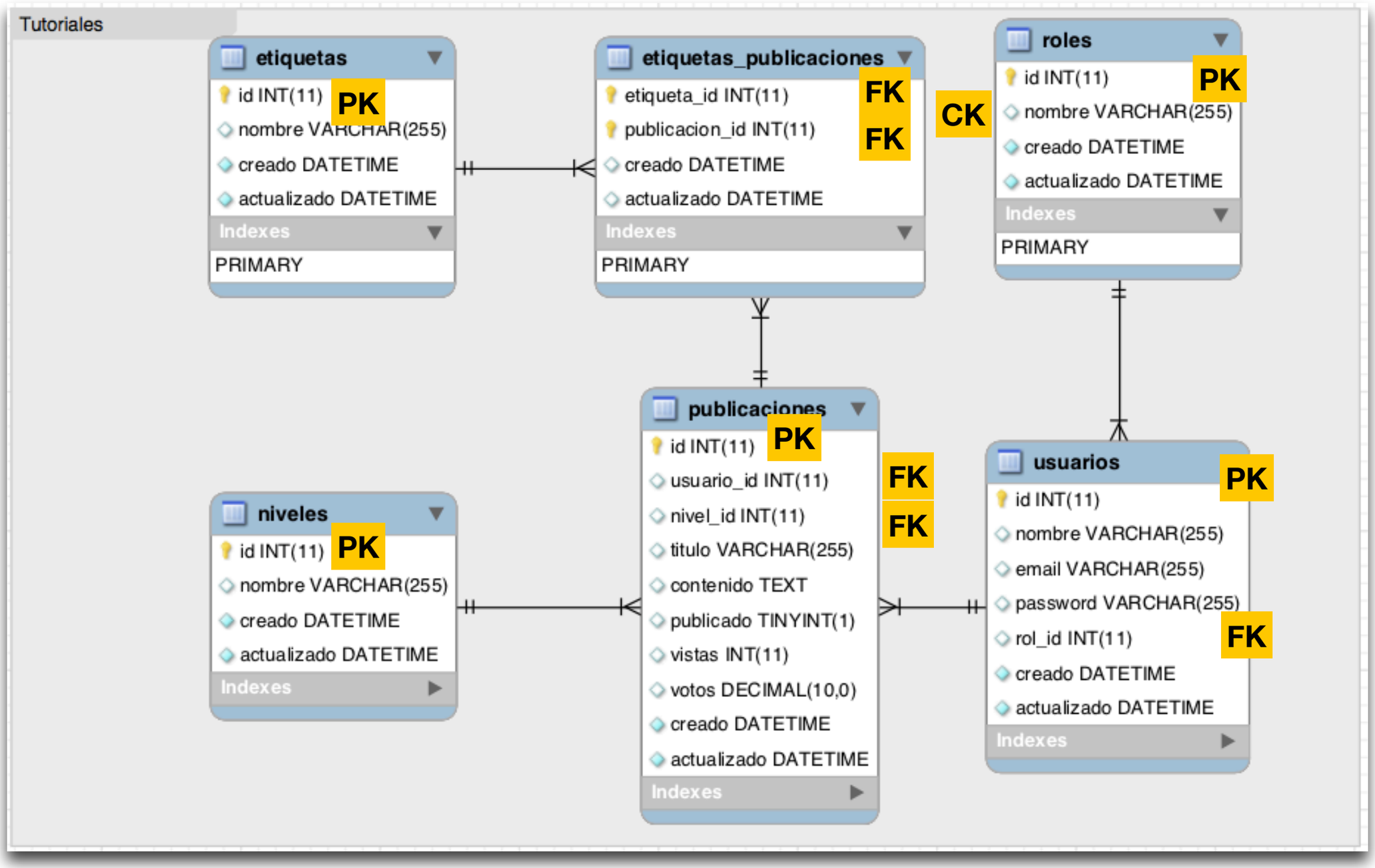
- Asociar registros entre diferentes tablas.

Para asociar datos almacenados en una tabla “A” con datos almacenados en una tabla “B” se crea una columna en la tabla “B” en la cual se va a almacenar un valor que distinga a que registro de la tabla “A” pertenecen. A esta columna se le conoce como llave Foranea (FK). El valor almacenado es el valor de la llave primaria de la tabla “A”.

FK - Clave Foránea: Identifica un registro cuyos datos están en una tabla “A” con datos almacenados en otra tabla “B”.

CK - Clave Compuesta: Este tipo de clave utiliza dos o más columnas para formar una clave primaria. Generalmante se utiliza en tablas de unión cuya función es asociar un dato de una tabla con muchos datos de otra tabla y viceversa. Cuando esto sucede cada una de las columnas funciona como FK de la columna a la cual hace referencia.

MySQL - Identificadores



MySQL - Identificadores

niveles	
id	nombre
1	Fácil
1	Difícil

etiquetas	
id	nombre
4	Cliente
6	FrontEnd

etiquetas_publicaciones	
etiqueta_id	publicacion_id
4	1
6	1

publicaciones		
id	nombre	nivel_id
1	HTML	1
2	PHP	1

Tareas generales - bd

Tarea	Descripción	Sintaxis
Crear	Útil para hacer proyectos nuevos o para importar alguna BD que no tenga dentro de su script la generación de la BD.	**CREATE DATABASE** nombreBD **CHARSET** utf8 **COLLATE** utf8_general_ci**;**
Eliminar	Borra la BD.	**DROP DATABASE** nombreBD**;**
Utilizar	Selecciona la BD en la que se va a trabajar.	**USE** nombreBD**;**
Importar	Útil para transportar las BD de una computadora a otra o, para utilizar respaldos en BD que dejaron de funcionar.	**mysql -u root -p <** nombreBD.sql
Exportar	Útil para hacer respaldo.	**mysqldump -u root -p >** nombreBD.sql

Tareas generales - tablas

Tarea	Descripción	Sintaxis
Crear	Crea una tabla dentro de la BD seleccionada (en la que se está trabajando). Se tiene que especificar el nombre, tipo de dato y opciones* de la(s) columna(s). * Cuando sea necesario. Estos atributos pueden ser un valor predeterminado si el usuario no asigna un valor, no permitir valores vacíos. Entre otros.	**CREATE TABLE** nombreTabla(nombreColumna1 TIPO_DATO OPCIONES, nombreColumna2 TIPO_DATO OPCIONES)**;**
Eliminar	Borra una tabla de la BD.	**DROP TABLE** nombreTabla**;**
Modificar	Cambiar la estructura de la tabla. Las modificaciones pueden ser (se pueden combinar en un solo ALTER TABLE): Agregar columna(s).	**ALTER TABLE** nombreTabla **ADD COLUMN** nombreColumna TIPO_DATO OPCIONES, **ADD COLUMN** nombreColumna TIPO_DATO OPCIONES**;**
	Eliminar columna(s)	**ALTER TABLE** nombreTabla **DROP COLUMN** nombreColumna**;**
	Cambiar tipo de dato que almacena la(s) columna(s).	**ALTER TABLE** nombreTabla **MODIFY** nombreColumna TIPO_DATO OPCIONES)**;**

Tareas generales - registros

Tarea	Descripción	Sintaxis
Crear	Crea un registro dentro de la tabla.	**INSERT INTO** nombreTabla (nombreColumna1, nombreColumna2) **VALUES** ('valor1','valor2');
Leer	Muestra los datos de una tabla. Se pueden especificar las columnas separadas por comas o utilizar * para seleccionar todas las columnas. Si no se especifica ninguna condición muestra todos los datos, si se especifica solo muestra los que cumplan con la condición.	**SELECT** nombreColumna(s) **FROM** nombreTabla(s) **WHERE** condicion(es);
Actualizar	Actualiza la(s) columna(s) de una tabla al valor asignado. Si no se establece una condición **actualiza todos** los registros de la tabla. Al principio es mejor hacer una consulta para asegurar que la condición es válida y regresa los resultados que se quieren modificar.	**UPDATE** nombreTabla **SET** nombreColumna1 = "valor1", nombreColumna2 = "valor2" **WHERE** condicion(es);
Eliminar	Elimina los registros que cumplan con una condición. Si no se establece una condición **elimina todos** los registros de la tabla. Al principio es mejor hacer una consulta para asegurar que la condición es válida y regresa los resultados que se quieren eliminar.	**DELETE FROM** nombreTabla **WHERE** condicion(es);

Software de administración MySQL

Para administrar MySQL se pueden utilizar diferentes programas (incluida la linea de comandos).
Una opción es phpMyAdmin - software de administración de MySQL escrito en PHP - que viene instalado en **XAMPP.**
Para utilizarlo hay que iniciar xampp y en un navegador ir a: http://localhost/phpmyadmin

PHP MyAdmin

Software de administración (aplicaciones independientes que no utilizan un navegador web como base para su funcionamiento).

MySQL Workbench

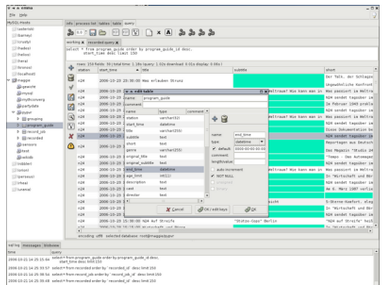
Windows, Linux, Mac OS X
* Gratuito pero se necesita crear cuenta en Oracle para descargarlo.

SequelPro

Mac OS X

HeidiSQL

Windows



Emma

Linux

Valores predeterminados para definir tablas, columnas

Columna	Valor	Valores adicionales
id	int auto_increment not null primary_key	PRIMARY KEY / CLAVE PRIMARIA
‘tabla’_id	int not null	FOREIGN KEY / CLAVE FORÁNEA
texto	varchar(255)	
creado / actualizado	timestamp	
texto	text	
números	int	
decimales (uso en tipo de cambio)	decimal(19,2)	
codificación	utf8	
colación	utf8_general_ci	

Una manera para comenzar a crear bd / tablas / columnas / es utilizar estos valores predeterminados (donde apilquen) como referencia y modificarlos según se necesite.

Si los valores que se quieren almacenar son necesarios para que los datos sean válidos utilizar NOT NULL en la columna.
Si se le quiere dar un valor predeterminado en caso de que no se ingrese el dato se utiliza DEFAULT.

id - Presente en casi todas las tablas*
*excepto tablas de unión - join tables que el id está formado por todos los identificadores que hacen referencia a las tablas que está uniendo.

‘tabla’_id - Llave Foranea. El nombre ‘tabla’ se sustituye por el nombre de la tabla la cuál contiene los datos a los cuales se hace referencia.

Ejercicio

Crear una BD que se llame mis_pendientes_nombre

Crear una tabla que se llame pendientes con los siguientes atributos

id
contenido
creado
actualizado

Exportar la BD como SQL

Borrar la BD del sistema

Importar la BD

CRUD

C	**CREAR**	**INSERT INTO** nombreTabla **(** nombreColumna1**,** nombreColumna2**)** **VALUES (‘**valorColumna1**’,** valorColumna2**)**
R	**LEER**	**SELECT** columna(s) **FROM** tabla(s) **WHERE** (cuando) se cumpla(n) condición(es), si no se indica se selecciona todo.
U	**ACTUALIZAR**	**UPDATE** nombreTabla **SET** nombreColumna1 **=** ’valorColumna1’**,** nombreColumna2 **=** ‘valorColumna2’ **WHERE** (cuando) se cumpla(n) condición(es), si no se indica se selecciona todo.
D	**ELIMINAR**	**DELETE FROM** nombreTabla **WHERE** (cuando) se cumpla(n) condición(es), si no se indica se selecciona todo.

Crear (C)

Elegir la tabla en donde se van a insertar los registros.
Elegir las columnas que van a contener valor. Algunas columnas pueden tener valores predeterminados (como id, creado) que no es necesario incluir al momento de crear el registro para que sean válidos.

Si se va a insertar un solo registro:

INSERT INTO nombreTabla(nombreColumna1, nombreColumna2) **VALUES** (‘valorColumna1’, ‘valorColumna2’)

Si se va a insertar más de un registro:

INSERT INTO nombreTabla(nombreColumna1, nombreColumna2) **VALUES** (‘valorColumna1’, ‘valorColumna2’)**,** (‘valorColumna1’, ‘valorColumna2’)

```
1 SELECT * FROM niveles
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	creado	actualizado
1	Fácil	2013-01-18 23:11:53	0000-00-00 00:00:00
2	Medio	2013-01-18 23:11:59	0000-00-00 00:00:00
3	Difícil	2013-01-18 23:12:05	0000-00-00 00:00:00

```
1 SELECT * FROM niveles
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	creado	actualizado
1	Fácil	2013-01-18 23:11:53	0000-00-00 00:00:00
2	Medio	2013-01-18 23:11:59	0000-00-00 00:00:00
3	Difícil	2013-01-18 23:12:05	0000-00-00 00:00:00
4	Pro	2013-01-23 15:52:48	2013-01-23 15:52:23

Leer (R)

Las consultas son necesarias para actualizar y / o eliminar datos además de consultar los datos que se encuentran almacenados.

Elegir la(s) tabla(s) a consultar.
Elegir la(s) columna(s) a consultar - El caracter * representa todas las columnas de la tabla.
Seleccionar todas las columnas.

SELECT * **FROM** nombreTabla;

Seleccionar algunas columnas.

SELECT nombreColumna1, nombreColumna2 **FROM** nombreTabla;

Filtrar datos.

La consulta anterior devuelve todos los registros que se encuentran en la tabla porque no hay un criterio de comparación para poder filtrar los datos. Para establecer criterios de comparación se utiliza la palabra WHERE y el criterio de comparación (al menos una condición) para mostrar solo los datos que cumplan con ese criterio.
La condición está formada por el nombre de una columna, un operador de comparación y un valor para comparar.

SELECT nombreColumna1, nombreColumna2 **FROM** nombreTabla **WHERE** nombreColumna3 = ‘valor’;

Leer (R)

```
1 SELECT * FROM usuarios
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	email	password
1	Garland Roberts	hollie_emard@reichel.ca	375f542fcc03
2	Aliza Leuschke	watson_koepp@spencerlarkin.co.uk	fc387adac877
3	Percival VonRueden	gideon@donnelly.ca	2898eed1ce28
4	Dashawn Renner	louie@satterfield.uk	f83dd3e7faec
5	Ms. Eileen Waelchi	courtney.schumm@rippin.us	0c6ff76dfa99
6	Blaise Gaylord	torrey_renner@bradtke.uk	2071acd0e271
7	Miss Ephraim Strosin	josue@mosciski.biz	NULL
8	Marilie Vandervort	oma_stanton@zulauf.org	NULL
9	Hailey Corwin	toney.reynolds@bogan.biz	NULL
10	Chase Koss	wilton@cole.org	NULL
11	Dr. Maximillia Murazik	damon.bergnaum@bode.info	NULL
12	Dale Gerhold	bernie@davis.net	NULL
13	Carter Kunde	kallie_mclaughlin@bednar.name	NULL
14	Kaycee Maggio	clair@anderson.net	NULL
15	Mr. Darron Dooley	bertha_stroman@leannon.info	NULL
16	Luz Yundt	aditya.hettinger@fisher.biz	NULL

```
1 SELECT * FROM usuarios WHERE email LIKE "%.uk"
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	email	password	rol_id
2	Aliza Leuschke	watson_koepp@spencerlarkin.co.uk	fc387adac877	1
4	Dashawn Renner	louie@satterfield.uk	f83dd3e7faec	2
6	Blaise Gaylord	torrey_renner@bradtke.uk	2071acd0e271	2

```
1 SELECT nombre, email FROM usuarios WHERE email LIKE "%.uk"
```

Query Favorites ▾ Query History ▾ Run Current Run All

nombre	email
Aliza Leuschke	watson_koepp@spencerlarkin.co.uk
Dashawn Renner	louie@satterfield.uk
Blaise Gaylord	torrey_renner@bradtke.uk

Actualizar (U)

Modificar datos.

Elegir la(s) tabla(s) en donde se encuentran los datos a modificar.
Elegir la(s) columna(s) a modificar y el valor a asignar a la columna.
Especificar la(s) condición(es) para seleccionar los registros a modificar.
Solo se modificará(n) el / los registro(s) que cumpla con la(s) condición(es)
Si no se establece una condición para filtrar los datos a eliminar se modificarán **TODOS** los registros.

UPDATE nombreTabla **SET** nombreColumna1 = 'valorColumna1', nombreColumna2 = ‘valorColumna2’ **WHERE** columna = valor

```
1 SELECT * FROM niveles
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	creado	actualizado
1	Fácil	2013-01-18 23:11:53	0000-00-00 00:00:00
2	Medio	2013-01-18 23:11:59	0000-00-00 00:00:00
3	Difícil	2013-01-18 23:12:05	0000-00-00 00:00:00
4	Pro	2013-01-23 15:52:48	2013-01-23 15:52:23

```
1 SELECT * FROM niveles
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	creado	actualizado
1	Fácil	2013-01-18 23:11:53	0000-00-00 00:00:00
2	Medio	2013-01-18 23:11:59	0000-00-00 00:00:00
3	Difícil	2013-01-18 23:12:05	0000-00-00 00:00:00
4	Profesional	2013-01-23 15:59:15	2013-01-23 15:59:15

Eliminar (D)

Eliminar registros.

Elegir la(s) tabla(s) en donde se encuentran los datos a eliminar.
Especificar la(s) condición(es) para seleccionar los registros a eliminar.
Solo se eliminará(n) el / los registro(s) que cumpla con la(s) condición(es)
Si no se establece una condición para filtrar los datos a eliminar se borrarán **TODOS** los registros.

DELETE FROM nombreTabla **WHERE** columna = valor

Nota:
Eliminar **BORRA** todo el registro.

```
1 SELECT * FROM niveles
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	creado	actualizado
1	Fácil	2013-01-18 23:11:53	0000-00-00 00:00:00
2	Medio	2013-01-18 23:11:59	0000-00-00 00:00:00
3	Difícil	2013-01-18 23:12:05	0000-00-00 00:00:00
4	Pro	2013-01-23 15:52:48	2013-01-23 15:52:23

```
1 SELECT * FROM niveles
```

Query Favorites ▾ Query History ▾ Run Current Run All

id	nombre	creado	actualizado
1	Fácil	2013-01-18 23:11:53	0000-00-00 00:00:00
2	Medio	2013-01-18 23:11:59	0000-00-00 00:00:00
3	Difícil	2013-01-18 23:12:05	0000-00-00 00:00:00

Operadores de comparación

Operador	Descripción
=	Selecciona un valor igual al indicado.
<>, !=	Selecciona un valor distinto al indicado.
<	Selecciona un valor menor que el indicado (no lo incluye).
<=	Selecciona un valor menor o igual que el indicado (lo incluye).
>	Selecciona un valor mayor al indicado (no lo incluye).
<=	Selecciona un valor mayor o igual al indicado (lo incluye).

Operador	Descripción
LIKE	El texto dentro de la columna tiene que parecer al que se encuentra después del operador (entre comillas o apóstrofes). El texto para comprar puede utilizar el caracter % que tiene como función representar cualquier caracter escrito cualquier número de veces.
NOT LIKE	El texto que se encuentra en la columna a consultar **no** debe parecer al texto que se encuentra después del operador. El texto para comprar puede utilizar el caracter % que tiene como función representar cualquier caracter escrito cualquier número de veces.

Operadores de comparación

Operador	Ejemplo
=	id = 10 (El valor almacenado en id tiene que ser igual a 10)
<>, !=	permiso <> ‘administrador’ (El valor almacenado en permiso tiene que ser diferente a la palabra ‘administrador’)
<	calificacion < 6 (no incluye el 6)
<=	calificacion <= 5 (incluye el 5)
>	edad > 17 (no incluye el 17)
<=	edad >= 18 (incluye el 18)

Operador	Ejemplo
LIKE	El uso de % en las búsquedas de texto (LIKE) realiza las siguientes acciones. Igual: email LIKE ‘admin@example.com’ Termine con: email LIKE ‘%.org’ Inicie con: nombre LIKE ‘A%’ Contenga dentro: contenido LIKE ‘%mysql%’
NOT LIKE	El uso de % en las búsquedas de texto (NOT LIKE) realiza las siguientes acciones. Diferente: email NOT LIKE ‘admin@example.com’ No termine con: email NOT LIKE ‘%.edu’ No inicie con: nombre NOT LIKE ‘@%’ No contenga: categoria NOT LIKE ‘%html%’