# Como diseñar eficazmente una base de datos

Hola a todos, primero que nada una disculpa por no haber actualizado estas últimas semanas, hemos estado trabajando en algunas mejoras importantes que aplicaremos en la web en los próximos días.

En esta ocasión, les hablare sobre como diseñar eficazmente una base de datos, este tutorial es la tercera parte del mega tutorial "Desarrollo de Proyectos".

Si no has leído las partes anteriores de este tutorial te recomiendo lo hagas para que no pierdas el hilo y entiendas más fácilmente todo.

Bueno, una vez que ya hemos decidido el nombre de nuestro proyecto, calculado la inversión que vamos a realizar y haber realizado una lluvia de ideas grupal para sacarle el máximo provecho al proyecto, vamos ahora a diseñar eficazmente nuestra base de datos para optimizar los tiempos de ejecución y por consecuente hacer que nuestro sistema sea más rápido, estable y manejable.

Antes de decidir cual sistema gestor de bases de datos vamos a utilizar debemos hacer uso de los diagramas de Entidad-Relación para saber exactamente qué es lo que necesitamos.

El modelo entidad-relación (E-R) se conforma de un conjunto de conceptos que nos van a permitir describir la realidad de nuestro proyecto mediante una serie de gráficos y símbolos.

Para entender más al modelo E-R debemos definir algunos conceptos:

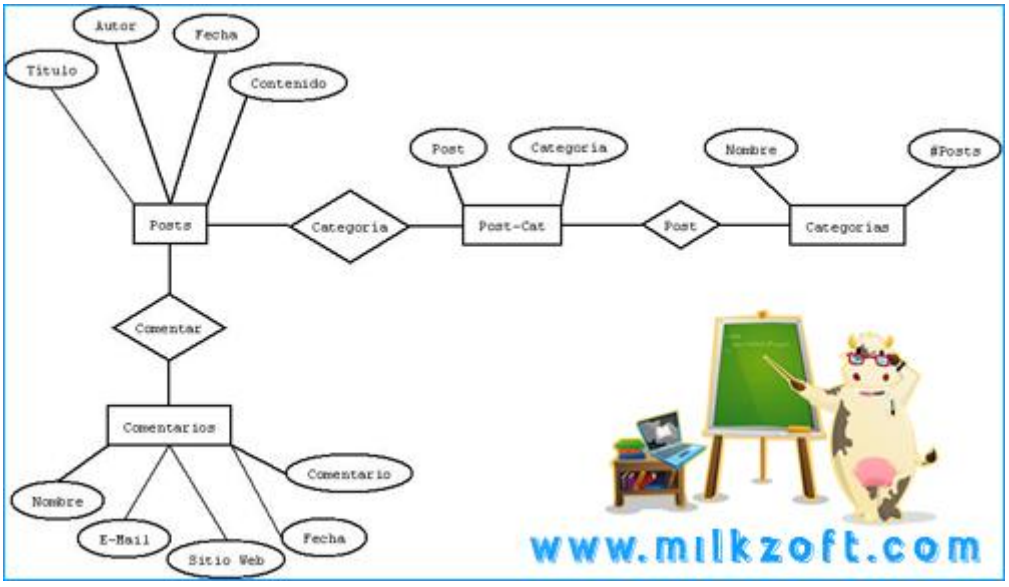
**Entidad (Instancia):** Es cualquier objeto o concepto sobre el cual se obtiene información (ej. Casas, Empresas, Empleados, Automóviles, etc.).

**Conjunto de Entidades (Clase):** Es una colección de objetos muy similares.

**Atributos:** Son las propiedades de las entidades.

**Relaciones:** Es una correspondencia o asociación entre dos o más entidades.


Haremos un pequeño ejemplo sobre como diseñar la base de datos de un blog en el modelo E-R y posteriormente con un sistema gestor de base de datos.



Como se puede observar en la imagen anterior, tenemos 4 entidades: Posts, Categorías, Post-Cat y Comentarios, cada una tiene ciertos atributos y sus relaciones correspondientes. El esquema que maneje es muy sencillo, simplemente se crea un Post (Título, Autor, Fecha y Contenido) y si se relaciona a alguna categoría se hace la relación mediante la entidad Post-Cat, la cual tiene el ID del Post y el ID de la categoría que se va a relacionar, esto nos permite que la relación pueda ser de uno a muchos (1:N) y de la misma manera se relaciona a la entidad Categorías de la cual obtendrá el nombre de la categoría y el número de Posts que se han hecho en ella. Otra relación es la de "Comentar" la cual nos sirve para que las personas que vean los Posts puedan comentarlos dejando su Nombre, E-Mail, Sitio Web, registrar al fecha en que se hizo y el Comentario que dejo.

**¿Cuál sistema de gestión de base de datos es el adecuado para mi proyecto?**

A este punto quería llegar, muchas veces cuando queremos abrir un proyecto casi siempre no le tomamos importancia a cual sistema de gestión de base de datos es el adecuado para nuestro proyecto, por lo general compramos un servidor que tenga Apache, PHP y MySQL instalado y trabajamos sobre él, pero ¿conoces algún otro gestor de base de datos como Oracle, PostgreSQL, SyBase ó Interbase?, sabes ¿Cuáles son las diferencias entre unos y otros?, sus ¿ventajas y desventajas?

Primero tienes que definir la magnitud de tu proyecto, que tanto alcance puede tener y que tanto tráfico de información podrías manejar en un futuro. Si tu proyecto es pequeño (menos de 1 millón de registros) podrías optar por utilizar MySQL, si tu proyecto requiere de mayor seguridad en los datos y un volumen mayor de información (más de 1 millón de registros) podrías utilizar PostgreSQL, estos 2 gestores los mencione porque son de licencia de software libre.

**Diferencias entre MySQL y PostgreSQL**

Bueno en mi experiencia la principal diferencia que he notado con MySQL respecto a PostgreSQL es su velocidad a la hora de hacer consultas, la facilidad que tiene de adaptarse con plataformas como Windows, Linux, Solaris, Mac, entre otros, la gran adaptabilidad que tiene con el lenguaje de programación PHP y que no tiene límite en el tamaño de sus registros.

Por otro lado PostgreSQL aunque presenta un menor rendimiento respecto a MySQL (normalmente es 2 o 3 veces más lento que MySQL), en proyectos de mayor volumen de información presenta una mayor estabilidad y escalabilidad, de igual forma se tiene una mayor seguridad en nuestra información y tiene mejor soporte para triggers.

**Pasando del Modelo E-R al comando SQL**

Bueno, si ya has decidido cual sistema gestor de base de datos vas a utilizar para tu proyecto, solamente resta convertir el diagrama de E-R a comandos SQL para crear nuestra base de datos con las tablas necesarias. Crearemos las bases de datos y sus tablas en MySQL y en PostgreSQL.

## Comandos SQL para MySQL

Primero vamos a crear nuestra base de datos y nuestras tablas para utilizarla en MySQL.

```
CREATE DATABASE Mi_Proyecto;
```

El comando para crear una base de datos en MySQL es **CREATE DATABASE** y enseguida escribimos el nombre que le vamos a otorgar.

Para las tablas, tenemos contempladas 4 (Posts, Categorías, Post2Cats y Comentarios), veamos cómo se hace:

**Tabla Posts:**

```
CREATE TABLE Posts (
ID mediumint(8) unsigned default null auto_increment,
Titulo varchar(100) not null,
Autor varchar(50) not null,
Fecha datetime not null,
Contenido text not null,
PRIMARY KEY (ID)
);
```

**Tabla Categorías:**

```
CREATE TABLE Categorias (
ID mediumint(8) unsigned default null auto_increment,
Nombre varchar(100) not null,
Posts mediumint(8) unsigned default 0 not null,
PRIMARY KEY (ID)
);
```

**Tabla Post2Cats:**

```
CREATE TABLE Post2Cats (
ID mediumint(8) unsigned default null auto_increment,
ID_Post mediumint(8) unsigned default 0 not null,
ID_Category mediumint(8) unsigned default 0 not null,
PRIMARY KEY (ID)
);
```

**Tabla Comentarios:**

```
CREATE TABLE Comentarios (
ID mediumint(8) unsigned default null auto_increment,
Nombre varchar(100) not null,
Email varchar(50) not null,
Web varchar(50) not null,
Fecha datetime not null,
Comentario text not null,
PRIMARY KEY (ID)
);
```

Como puedes observar crear tablas en MySQL es muy sencillo, simplemente escribes el comando **CREATE TABLE** seguido del nombre de la tabla, y posteriormente el nombre de los campos el tipo que será y otros atributos que serán explicados enseguida.

**Mediumint:** El tipo mediumint es un valor entero (integer) el cual va de -8388608 a 8388607 registros y si se utiliza el atributo unsigned va desde 0 hasta 16,777,215. Si se requieren de más registros se puede utilizar el tipo int o el bigint el cual va desde -9,223,372,036,854,775,808 a 9,223,372,036,854,775,807 registros.

**Auto Increment:** Esta propiedad sirve para hacer que ese campo sea auto incrementable, es decir que cada registro nuevo que se inserte en la tabla su ID incremente en 1 automáticamente.
Varchar: El tipo varchar es una cadena con un máximo de 255 caracteres.

**Text:** El tipo text es al igual que el varchar una cadena con un máximo de 65,535 caracteres.
Datetime: El tipo Datetime nos creara un campo de tipo fecha con un formato de tipo "YYYY-MM-DD HH:MM:SS" por ejemplo: "2009-08-01 18:05:35".

**PRIMARY KEY:** La clave principal o Primary Key nos va a servir para definir un campo como único, es decir, que no habrá registros repetidos en ese campo, normalmente se utiliza para los IDs de las tablas.

**Comandos SQL para PostgreSQL**

En PostgreSQL cambia un poco la forma de crear las tablas, pero básicamente es muy similar.

```
CREATE DATABASE Mi_Proyecto WITH OWNER = Usuario ENCODING = 'UTF8';
```

Para crear las tablas utilizaremos los siguientes comandos:

**Tabla Posts:**

```
CREATE TABLE Posts (
"ID" serial,
"Titulo" character varying(100) NOT NULL,
"Autor" character varying(50) NOT NULL,
"Fecha" timestamp without time zone DEFAULT ('now'::text)::timestamp(6) with
time zone NOT NULL,
"Contenido" text NOT NULL,
CONSTRAINT Posts_pkey PRIMARY KEY ("ID")
);
```

**Tabla Categorías:**

```
CREATE TABLE Categorias (
"ID" serial,
"Nombre" character varying(100) NOT NULL,
"Posts" integer NOT NULL DEFAULT 0,
CONSTRAINT Categories_pkey PRIMARY KEY ("ID")
);
```

**Tabla Post2Cats:**

```
CREATE TABLE Post2Cats (
  "ID" serial,
  "ID_Category" integer NOT NULL DEFAULT 0,
  "ID_Post" integer NOT NULL DEFAULT 0,
  CONSTRAINT Post2Cats_pkey PRIMARY KEY ("ID")
);
```

**Tabla Comentarios:**

```
CREATE TABLE Comentarios (
"ID" serial,
"Nombre"  character varying(100) NOT NULL,
"Email" character varying(50) NOT NULL,
"Web" character varying(50) NOT NULL,
"Fecha" timestamp without time zone DEFAULT ('now'::text)::timestamp(6) with
time zone NOT NULL,
"Comentario" text NOT NULL,
CONSTRAINT Comentarios_pkey PRIMARY KEY ("ID"));
```

Como puedes observar en PostgreSQL los campos se escriben dentro de comillas dobles (“”), los ID son de tipo serial los cuales crean unas secuencias que irán incrementando en 1 al igual que ocurre con el auto increment de mysql.

Para finalizar este curso les proporcionare enlaces de herramientas que les pueden ser de gran ayuda a la hora que vayan a diseñar e implementar su base de datos.

**Herramientas para el manejo de bases de datos**

MySQL – www.mysql.com
PostgreSQL – www.postgresql.org
Apache – www.apache.org
PHP – www.php.net
AppServ – www.appservnetwork.com
XAMPP – www.apachefriends.org
PgAdmin III – www.pgadmin.org
phpMyAdmin – www.phpmyadmin.net
phpPgAdmin – http://phppgadmin.sourceforge.net
Dia (Para hacer diagramas E-R) – http://dia-installer.de

Un Saludo.

**Atentamente**

Carlos Santana Roldán

***NOTA:*** *Este tutorial fue creado por Carlos Santana Roldán para MilkZoft, si deseas tomar total o parcialmente el contenido publicado te pedimos nos envíes un mensaje utilizando el formulario de contacto y agregando un enlace a MilkZoft como fuente de los datos (puedes insertar el siguiente código en tu blog o página web).*

**Fuente:** MilkZoft