# Nociones básicas para crear una base de datos

Comprendimos cómo es la estructura de una base de datos, ahora nos enfocamos en cómo crear una con algunos pasos sencillos.

Comenzaremos por dar permisos necesarios en nuestra base de datos para poder operar, voy a tomar de ejemplo una base de datos MySQL, pero se puede aplicar en otros motores similares relacionales.

## Creando el usuario:

Para acceder a nuestra base de datos, vamos a necesitar que sea por medio de un usuario. La seguridad lo es todo en estos casos, por eso no recomiendo que se utilice el usuario que trae por defecto MySQL (root) y lo mismo se da en otros motores (de base de datos: Ej. MS SQL Server, Oracle, PostgreSQL u otros).

En primera instancia, nos vamos a conectar con el usuario principal (luego de la instalación), para luego poder crear un usuario que nos permitirá trabajar tanto a nosotros como a las aplicaciones con nuestra base de datos.

Ejecutamos el siguiente comando para conectarnos:

```
Para conectarse:
  ./mysql -uroot
```

**Una aclaración importante:** en el curso no veremos lo que refiere a la instalación del motor, ya que no sería del todo conveniente enfocarnos en uno solo. Te recomiendo dirigirte a la página principal del motor de base de datos que elijas para poder acceder a la documentación de instalación en el sistema operativo que vayas a usar. Seguramente, lo vas a encontrar sumamente explicado y en el idioma que requieras.

Antes de continuar con el siguiente comando, también quiero aclarar que estos comandos que muestro pueden ser ejecutados por medio del cliente que trae por defecto el motor o cualquier cliente que elijas. Para la ejecución de los comandos vamos a utilizar un cliente SQL. El cliente es aquel que te va a facilitar la conexión a una base de datos. Los motores suelen traer un cliente por defecto y en muchos casos de accesos por consola (línea de comando de tu sistema operativo) pero existen otros con una interfaz visual que suelen ser más completos y traer funcionalidades adicionales. Por los ejemplos que veremos en el curso, nos alcanzará con los que traen por defecto los motores.

Ahora sí, pasemos a la creación de un usuario:

```
-- Para crear un usuario remoto
CREATE USER 'miweb'@'ip_address' IDENTIFIED BY 'StrongPassword';

-- Para crear un usuario local
CREATE USER 'miweb'@'localhost' IDENTIFIED BY 'StrongPassword';
```

Vemos dos ejemplos distintos donde aparece por primera vez la sentencia reservada "CREATE". Vamos a usarla para la definición de muchos de los objetos de nuestra base de datos.

En este caso "CREATE USER" nos permite crear un usuario en el motor: "miweb", es el nombre que le voy a dar y tenemos la opción de crearlo local usando "localhost" (acompañado de un @) o en una base de datos en alguna dirección remota "ip_address", se debería reemplazar por la dirección IP de nuestra base remota.

No debemos olvidar la contraseña (*password* en inglés) que nos agregará seguridad y será requerida cada vez que ingresemos. La palabra "StrongPassword" es solo un ejemplo, deberemos reemplazarlo por un juego de caracteres que RECORDEMOS luego (aconsejable siempre usar mayúsculas, números y caracteres especiales menos el @).

Si todos nos salió bien ya tendríamos nuestro usuario y nos podríamos conectar.

```
Conectarme con mi usuario:

    mysql -u miweb -p  -- luego me va a solicitar mi contraseña.
```

## Permisos al usuario:

Vimos cómo crear nuestro usuario, pero es necesario revisar los permisos que deberá tener. Por defecto, cuando creamos un usuario se le asignan permisos variados.

A continuación, te dejo sentencias que nos van a permitir setear o, mejor dicho, definir los permisos sobre mi usuario. Esto nos va a servir para poder trabajar aún más sobre la seguridad de nuestra Base de Datos.

### Darle permisos:

- La sentencia:

GRANT permission_type ON database.table TO 'username'@'localhost';

Donde permission_type puede ser INSERT, UPDATE, DELETE, etc. (revisar manual del motor de base de datos que hayamos seleccionado).

**Ejemplo:**

GRANT INSERT ON *.* TO 'username'@'localhost';

- Darle permisos a mi usuario en una base particular, con todos los privilegios:

GRANT ALL PRIVILEGES ON MiBaseDeDatos.* TO 'miweb'@'localhost';

- Ver que permisos tengo:

SHOW GRANTS for username;  -- reemplazo username por mi nombre de usuario

**Ejemplo:**

SHOW GRANTS for 'miweb'@'localhost';

Otros ejemplos:

- SHOW GRANTS;
- SHOW GRANTS FOR CURRENT_USER;
- SHOW GRANTS FOR CURRENT_USER();

Sacarle Permisos:

**Ejemplo:**

REVOKE DELETE ON MiBaseDeDatos.* TO 'miweb'@'localhost';

De todas formas, para esto siempre podemos recurrir al manual de la Base de Datos que hayas elegido.

## Creando mi primera base de datos

Para crear nuestra primera base lo más complejo será elegir el nombre ya que la sentencia presenta una menor complejidad.

```
CREATE DATABASE MiBaseDeDatos;
```

Para ver que se haya creado correctamente, podemos usar el siguiente comando:

```
show databases;  -- Mostrar bases de datos
```

Nos va a mostrar las bases que fueron creadas dentro del motor de Base de Datos.

Y nos resta conectarnos o, mejor dicho, hacer uso de nuestra base de datos.

```
use [nombre_base];  -- conectarse a una base;
```

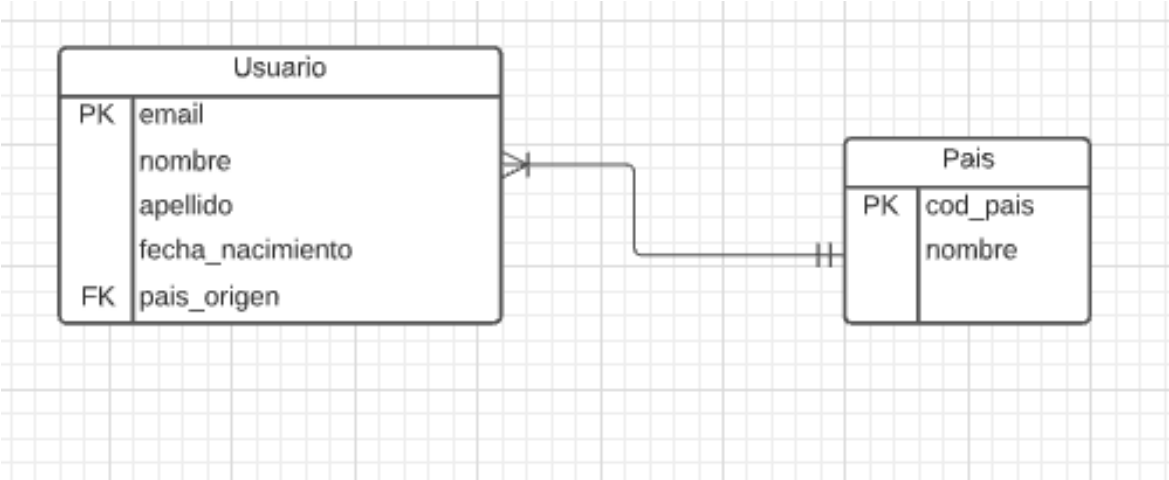
Luego de esta sentencia donde pongamos el nombre de la base de datos creada, todas las sentencias que ejecuten se harán sobre esa base de datos.

## Creando mi primera tabla

Pasemos a uno de los principales objetos con los que vamos a interactuar: las tablas. La sentencia de creación de tablas, en los manuales puede ser muy extensa y quizás hasta algo confusa. Por esto, nos vamos a parar sobre una versión simplificada que va a ser útil para el

curso y para nuestra implementación en un desarrollo web. Y podemos recurrir al manual del motor de Base de Datos que hayamos elegido para más información.

Vamos a recurrir a nuestro ejemplo de Usuario y País.




Y vamos a transformarlo de entidad a tabla en nuestra base de datos.

Vamos a ir analizando cada una de las sentencias que voy a mostrar y entenderlas en detalle.

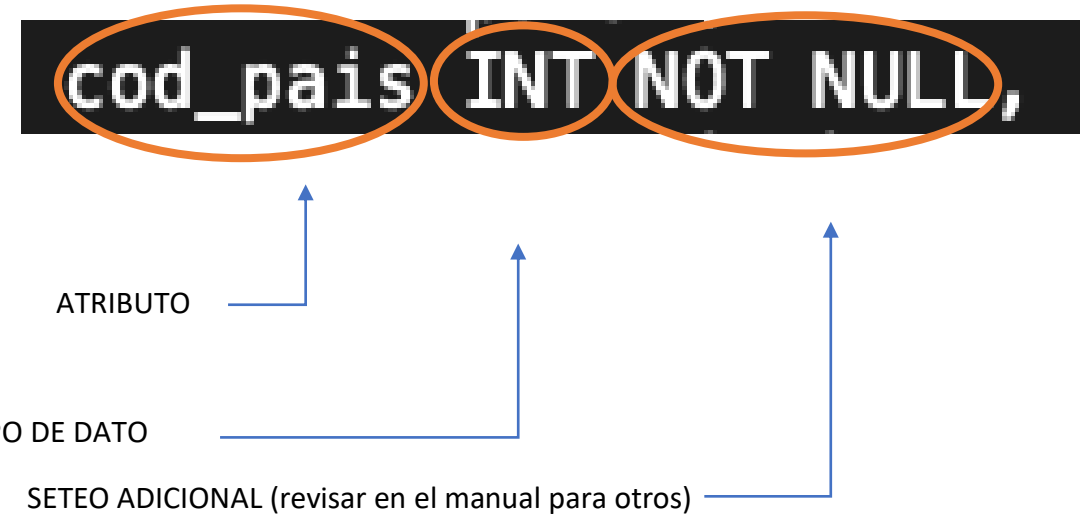
Arrancamos por crear la tabla PAIS:

```
CREATE TABLE pais (
    cod_pais INT NOT NULL,
    nombre VARCHAR(20) NOT NULL,
    PRIMARY KEY(cod_pais)
);
```

Iniciamos por CREATE TABLE nombre_tabla donde nombre_tabla lo reemplazamos por el nombre que queramos darle a nuestra tabla. Inmediatamente abrimos paréntesis y al final vamos a terminar cerrando paréntesis y luego punto y coma “;”. Esto es normal en los motores para decir que nuestra sentencia llega hasta ahí.

Dentro, tenemos en principio, el listado de atributos (columnas) que definimos para nuestra tabla y siempre lo acompañamos del tipo de dato. Separando por comas cada uno de los atributos y seteos (el último va sin coma al final). Cada atributo o columna permite, luego del tipo de dato, agregar algún seteo adicional. En mi caso, agregue "NOT NULL" para avisarle al motor de base de datos que no permita que se ingrese un registro nuevo donde esa columna no esté definida.

Esto último lo vamos a ver en detalle en los próximos módulos, pero en la definición de la tabla debemos aclararlo para evitar que se nos escape un dato sin definir que necesitemos.




Para definir cuál será la clave primaria, utilizamos las palabras reservadas PRIMARY KEY y entre paréntesis definimos, separado por comas, el o los atributos/columnas que serán parte.

```
PRIMARY KEY(cod_pais)
```

Ahora creamos la tabla USUARIO

```
CREATE TABLE usuario (
    email VARCHAR(250) NOT NULL,
    nombre VARCHAR(45),
    apellido VARCHAR(45),
    fecha_nac DATE,
    pais_origen INT NOT NULL,
    PRIMARY KEY (email),
    FOREIGN KEY (pais_origen) REFERENCES pais(cod_pais)
);
```

En este caso repetimos pasos similares, pero incorporamos algunos tipos de datos nuevos referidos a los grupos de Tiempo y Caracteres. Justo en esos casos no usamos seteos adicionales, porque podríamos no querer que el motor restrinja los campos que deban ser completados al momento de incorporar un nuevo registro.

Pero hemos incluido algo nuevo, además de agregar la columna pais_origen, le definimos que se comporte como clave foránea de la tabla PAIS en el campo cod_pais. Es decir, que va a referenciar a esa tabla.

```
FOREIGN KEY (pais_origen) REFERENCES pais(cod_pais)
```

Lo hacemos con la palabra reservada FOREING KEY, seguido de paréntesis e indicando la columna en la tabla USUARIO que va a ser de tipo clave foránea, para luego agregar la palabra reservada REFERENCES e incorporando la tabla a la cual queremos referenciar. En este caso PAIS se abre paréntesis y agrega la columna de la tabla PAIS que referenciamos, cod_pais.

Podríamos no hacer uso de esto último y que seamos nosotros, los creadores de la base, quienes sabemos qué columna es una clave foránea y dónde referencia. Pero de alguna forma, esto implica corromper la integridad referencial de la base (Teorema ACID) y como consecuencia, podría suceder que un registro se borre en una tabla referenciada y el motor no lo evite. Adicionalmente, si alguien más trabaja con nuestra base se pierda la información sobre este comportamiento.

## Visualizando las tablas:

Ya vimos cómo crear tablas, ahora nos falta algo para poder visualizarlas y ver cómo están armadas.

```
show tables; -- Mostrar tablas
```

Este nos permitirá ver todas las tablas creadas en la base que estemos usando (solo funciona cuando hayamos ejecutado previo USE mi_base_de_datos).

Como se puede ver en el siguiente ejemplo:

```
mysql> show tables;
+------------------------+
| Tables_in_mibasededatos |
+------------------------+
| pais                   |
| usuario                |
+------------------------+
2 rows in set (0.01 sec)
```

Y, si quisiéramos ver cómo está definida mi tabla, usamos la siguiente sentencia.

```
desc [nombre_tabla] -- mostrar la definición
```

Donde debemos reemplazar [nombre_tabla] por la que deseamos ver.

```
mysql> desc usuario;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| email       | varchar(250) | NO   | PRI | NULL    |       |
| nombre      | varchar(45)  | YES  |     | NULL    |       |
| apellido    | varchar(45)  | YES  |     | NULL    |       |
| fecha_nac   | date         | YES  |     | NULL    |       |
| pais_origen | int          | NO   | MUL | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

El motor nos mostrará cada uno de los campos/columnas/atributos (se le puede llamar de todas estas formas) con el correspondiente tipo de dato. Si permite que no sea definido (columna Null), si es o no clave primaria (o foránea), si por defecto puede tener algún valor (lo pueden ver en el manual) y algún extra que se pueda configurar.