Si queremos ver un listado con todas las bases de datos que tenemos creadas en nuestro servidor podemos usar el comando

show databases;

```
14 •  show databases;
```

Result Grid | Filter Rows:

Database
abm
abm_socios
classicmodels
commerce
hospital
escuela
gimnasio

Crear nueva base de datos

create database nombre_base;

Antes de comenzar a modificar/insertar datos en la base debemos indicarle a MySQL sobre qué base queremos trabajar

use nombre_base;

Para crear una tabla debemos indicar el nombre de la tabla y seguido los atributos de la tabla con sus respectivos tipos de datos (al final del pdf encontraremos una lista de los tipos de datos más usados)

```
create table ejemplo1 (idcliente mediumint,
                       nombre varchar(30),
                       apellido varchar(30),
                       telefono varchar(10),
                       mail varchar(30),
                       primary key (idcliente));
```

→ no olvidemos indicar cuál es la **PK** indicándole qué atributo será la primary key

```
create table ejemplo2 (IDorden mediumint,
                       -> fecha date,
                       -> idclientes mediumint,
                       -> tipopago varchar(30),
                       -> primary key (idorden));
```

Para ver cómo quedó armada nuestra tabla usamos el comando

describe ejemplo1;

```
12 •  describe ejemplo1;
```

Result Grid | Filter Rows: | Export:

Field	Type	Null	Key	Default	Extra
idcliente	mediumint	NO	PRI	NULL	
nombre	varchar(30)	YES		NULL	
apellido	varchar(30)	YES		NULL	
telefono	varchar(10)	YES		NULL	
mail	varchar(30)	YES		NULL	

Para comenzar a realizar modificaciones sobre las tablas vamos a usar el comando **alter table**

→ Para agregar una **foreing key**:

```
alter table ejemplo1
add foreign key (idcliente)
references ejemplo2 (idclientes);
```

Con este comando le decimos que a la tabla llamada ejemplo1 queremos que el atributo "idcliente" sea la foreing key para relacionar con el atributo "idclientes" de la tabla ejemplo2

! Tener en cuenta que para agregar una FK la tabla a la que hacemos referencia ya debe estar creada, y los atributos de ambas tablas que vamos a relacionar deben tener el mismo tipo de dato.

→ Para cambiar el tipo de dato que habíamos declarado en un principio. Por ejemplo, quiero que teléfono no sea más varchar, ahora quiero que sea int

```
alter table ejemplo1
modify column telefono int;
```

→ Para modificar el nombre de un atributo de la tabla

```
ALTER TABLE ejemplo1
RENAME COLUMN nombre TO nombre_de_pila;
```

→ Para Agregar un nuevo atributo a la tabla, no olviden escribir el tipo de dato

```
alter table ejemplo1
add fecha_nacimiento date;
```

→ Para eliminar un atributo de la tabla

```
alter table ejemplo1
drop column mail;
```

→ Para agregar claves primarias a tablas existentes que no tengan PK

Solo una pk

```
ALTER TABLE ejemplo1
ADD PRIMARY KEY (idcliente);
```

Para más de una pk

```
ALTER TABLE clientes
ADD PRIMARY KEY (idcliente, apellido);
```

Los tipos de datos más usados son:

→ *Numéricos*

- *MEDIUMINT*: El tipo de dato numérico MEDIUMINT permite números desde -8388608 hasta 8388607.
- ***INT:*** El tipo de dato numérico INT permite números desde -2147483648 hasta 2147483647.
- ***BIGINT*****:** El tipo de dato numérico BIGINT permite números desde -9223372036854775808 hasta 9223372036854775807.
- ***FLOAT*****:** El tipo de dato numérico FLOAT permite almacenar pequeños números decimales (de punto flotante). Al ser de punto flotante, sus cálculos son aproximados. Podemos especificar el número máximo de dígitos (tamaño) y el número de decimales (decimal).
- FLOAT(6,2) tendrá 4 dígitos enteros y 2 decimales, por ejemplo: 5467.67
- *DOUBLE*: El tipo de dato numérico DOUBLE permite almacenar grandes números decimales (de punto flotante). DOUBLE(5,1) tendrá 4 digitos enteros y 1 dígito decimal, por ejemplo 5467.1
- *DECIMAL*: El tipo de dato numérico DECIMAL permite almacenar grandes números decimales de punto fijo, por tanto, los cálculos con este tipo DECIMAL son exactos.

→ *De texto*

- *CHAR*: El tipo de dato CHAR sirve para almacenar una cadena de datos de longitud fija. Por ejemplo, CHAR(50), será un campo de longitud fija de 50 posiciones (que son de tipo carácter). La longitud máxima que podemos definir un campo CHAR es de 255.
- ***VARCHAR*****:** sirve para almacenar una cadena de datos (caracteres, números y caracteres especiales) de longitud variable La longitud máxima es de 255 caracteres.
- *TINYTEXT*: El tipo de datos TINYTEXT sirve para almacenar una cadena de datos (sólo caracteres, no admite número ni caracteres especiales) de una longitud máxima de 255 caracteres
- *TEXT*: El tipo de dato TEXT sirve para almacenar una cadena de caracteres de longitud máxima de 65,535 caracteres.
- *BLOB*: El tipo de datos BLOB sirve para almacenar datos de tipo BLOB (Binary Large Object). Admite una longitud máxima de 65,535 bytes de datos.
- *MEDIUMTEXT*: El tipo de dato MEDIUMTEXT sirve para almacenar una cadena con una longitud máxima de 16.777.215 caracteres.
- *MEDIUMBLOB*: El tipo de datos MEDIUMBLOB sirve para almacenar datos de tipo BLOB con una longitud máxima de 16.777.215 bytes.
- *LONGTEXT*: El tipo de dato LONGTEXT sirve para almacenar una cadena de longitud máxima de 4.294.967.295 caracteres
- *ENUM*: El tipo de dato ENUM sirve para introducir una lista de posibles valores. La longitud máxima es de 65.535 posibles valores. Por ejemplo si definimos una columna como ENUM(‘uno’, ‘dos’), entonces en esta columna solo puede almacenar los valores de ‘uno’ o ’dos’. Si queremos insertar cualquier otro valor (por ejemplo ‘tres’), no se grabará ‘tres’ y en su lugar quedara el campo vacío, sin valor (‘’).
- *SET*: El tipo de dato SET es similar a ENUM pero la longitud máxima de valores posibles es de 64, y los valores posibles se pueden combinar.

→ *Fecha*

- ***DATE*****:** Este tipo de datos se utiliza para almacenar fechas sin hora. El formato es 'YYYY-MM-DD', donde YYYY representa el año con cuatro dígitos, MM el mes con dos dígitos y DD el día con dos dígitos.
  Ejemplo: '2023-04-23'
- *TIME*: Este tipo de datos se utiliza para almacenar valores de hora sin fecha. El formato es 'HH:MM:SS', donde HH representa la hora en formato de 24 horas, MM los minutos y SS los segundos. Ejemplo: '13:30:00'
- *DATETIME*: Este tipo de datos se utiliza para almacenar fechas y horas. El formato es 'YYYY-MM-DD HH:MM:SS', donde YYYY representa el año con cuatro dígitos, MM el mes con dos dígitos, DD el día con dos dígitos, HH la hora en formato de 24 horas, MM los minutos y SS los segundos. Ejemplo: '2023-04-23 13:30:00'