**Base de datos relacionales**: Colección de elementos de datos organizados en un conjunto de tablas conformadas por filas y columnas a través de las cuales se puede acceder a la información para leerla, cambiarla, borrarla o incluso insertar nuevos datos.

**SQL**: un lenguaje de consulta estructurada utilizado para administrar y recuperar información desde sistemas de gestión de bases de datos relacionales. Un tipo de **lenguaje de programación** que te permite manipular y descargar datos de una base de datos. Tiene capacidad de hacer cálculos avanzados y álgebra.

Tipo de bases de datos SQL:

1. Orace
2. MySQL
3. Microsoft SQL Server
4. PostgreSQL

**Datos estructurados**: Son archivos de tipo texto que se suelen mostrar en filas y columnas con títulos. Son datos que pueden ser ordenados y procesados fácilmente por todas las herramientas de minería de datos. Lo podríamos ver como si fuese un archivador perfectamente organizado donde todo está identificado, etiquetado y es de fácil acceso.

**Datos no estructurados**: Los datos no estructurados, generalmente son datos binarios que no tienen estructura interna identificable. Es un conglomerado masivo y desorganizado de varios objetos que no tienen valor hasta que se identifican y almacenan de manera organizada. El 80 % de la información relevante para un negocio se origina en forma no estructurada, principalmente en formato texto.

Ejemplos: correos electrónicos, imágenes digitales, audio, vídeo, publicaciones en medios sociales etc.

**Sistema de Gestión de Base de Datos:** software, interfaz entre usuario, base de datos y aplicaciones que lo utilizan. Define, manipula, inserta datos.

Los SGBD más conocidas son:

1. Microsoft SQL Server
2. PostgreSQL
3. MySQL
4. MariaDB
5. Oracle

**NoSQL**: se trata de una amplia clase de [sistemas de gestión de bases de datos](https://es.wikipedia.org/wiki/Sistemas_de_gesti%C3%B3n_de_bases_de_datos) que difieren del modelo clásico de [SGBDR](https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos_relacionales) (Sistema de Gestión de [Bases de Datos Relacionales](https://es.wikipedia.org/wiki/Bases_de_Datos_Relacionales)) en aspectos importantes, siendo el más destacado que no usan [SQL](https://es.wikipedia.org/wiki/SQL) como [lenguaje principal de consultas](https://es.wikipedia.org/wiki/Lenguaje_de_consulta). Los datos almacenados no requieren estructuras fijas como [tablas](https://es.wikipedia.org/wiki/Tabla_(base_de_datos)), normalmente no soportan operaciones [JOIN](https://es.wikipedia.org/wiki/Sentencia_JOIN_en_SQL), ni garantizan completamente [ACID](https://es.wikipedia.org/wiki/ACID) (atomicidad, consistencia, aislamiento y durabilidad) y habitualmente escalan bien [horizontalmente](https://es.wikipedia.org/wiki/Escalabilidad#Escalabilidad_horizontal). Los sistemas NoSQL se denominan a veces "no solo SQL" para subrayar el hecho de que también pueden soportar lenguajes de consulta de tipo SQL.

**Tipos de sentencia SQL en una BBDD:**

1. DDL: definición de datos (referido a objetos)🡪CREATE, DROP, ALTER.   
   Por ejemplo, para crear una tabla:

CREATE TABLE Clients (

ID int,

FirstName nvarchar(50),

LastName nvarchar(50),

BirthDate date);

Si queremos eliminar una tabla:

DROP TABLE Clients

Añadir un nuevo campo a la tabla

ALTER TABLE Clients ADD Address nvarchar(250)

1. DML: manipulación de datis🡪SELECT, INSERT, DELETE, UPDATE

Por ejemplo, para introducir valores en la tabla

INSERT INTO Clients (ID, FirstName, LastName, BirthDate, Address)

VALUES(1, ‘JIM’, ‘Morrison’, ’12-08-1943’, ‘2100 Vernon Place’)

Para consultar la tabla:

SELECT \* FROM Clients

SELECT FirstName, LastName, Address FROM Clients

SELECT \* FROM Clients

WHERE ID > 2

Actualizar un registro de la tabla:

UPDATE Clients SET Address = ‘Somewhere in the UK’ WHERE Id = 3

Eliminar un registro de la tabla:

DELETE FROM Clients WHERE ID = 4

**Sentencias SQL utilizadas**: "Un cracker ha accedido a una información privada y tú, que eres "El inspector SQL", debe tomar una serie de decisiones mediante sentencias SQL para resolver el caso”.

1. ***SELECT \* FROM members;***

Se visualizan todos los campos de la tabla members.

1. ***SELECT \* FROM suscribers;***

Se visualizan todos los campos de la tabla suscribers.

1. ***SELECT Email, LastName, FirstName FROM mailing\_list;***

Se consultan los campos Email, LastName, FirstName de la tabla mailing list

1. ***SELECT MemberSince, UserName FROM members;***

Se consultan los campos MemberSince, UserName de la tabla members.

1. ***SELECT DISTINCT MemberSince FROM members;***

Valores no repetidos del campo MemberSince de la tabla members.

1. ***SELECT \* FROM users ORDER BY EmailAddress ASC;***

Muestra todos los campos de la tabla users ordenador, de manera ascendente, por el campo EmailAddress.

1. ***SELECT \* FROM users ORDER BY Surname DESC;***

Muestra todos los campos de la tabla users ordenador, de manera descendente, por el campo Surname.

1. ***SELECT DISTINCT FirstName, Surname, Posts FROM users ORDER BY Surname DESC;***

Muestra registros únicos de los campos FirstName, Surname, Posts, ordenados por el campo Surname de manera descendente.

1. ***SELECT EmailAddress, GivenName FROM mailing\_list ORDER BY EmailAddress DESC, GivenName DESC;***

Devuelve los campos EmailAddress, GivenName de la tabla mailing\_list ordenador por el campo EmailAddress de manera descendente y por el campo GivenName de manera descendente.

1. ***SELECT \* FROM users ORDER BY FamilyName ASC, EmailAddress DESC LIMIT 5;***

Selecciona los cinco primeros registros de la tabla users al ordenarlo por los campos FamilyName y EmailAddress, de manera ascendente y descendente, respectivamente.

1. ***SELECT DISTINCS LastAccess, Email, FamiliName FROM users ORDER BY Email ASC, FamilyName ASC LIMIT 20;***

Selecciona los 20 primeros registros únicos de los campos LastAccess, Email, FamiliName de la tabla users ordenados por los campos Email y FamiliName de manera ascendente.