Introducción a SQL

SQL significa lenguaje estructurado de consultas.

Es un lenguaje que permite ejecutar sentencias de código llamadas **consultas** (o *queries* en inglés) sobre una base de datos relacional para obtener y manipular datos.

Las principales sentencias se dividen en dos grandes grupos: **DDL** (Data Definition Language) y **DML** (Data Manipulation Language).

- **DDL:** es el lenguage de definición de datos. Permite crear tablas, modificarlas, borrarlas. También permite crear otro tipo de objetos como: vistas, disparadores, índices, etc.
- **DML:** es el lenguage de manipulación de datos. Sirve para hacer consultas a la base de datos, insertar, modificar o borrar registros de una tabla.

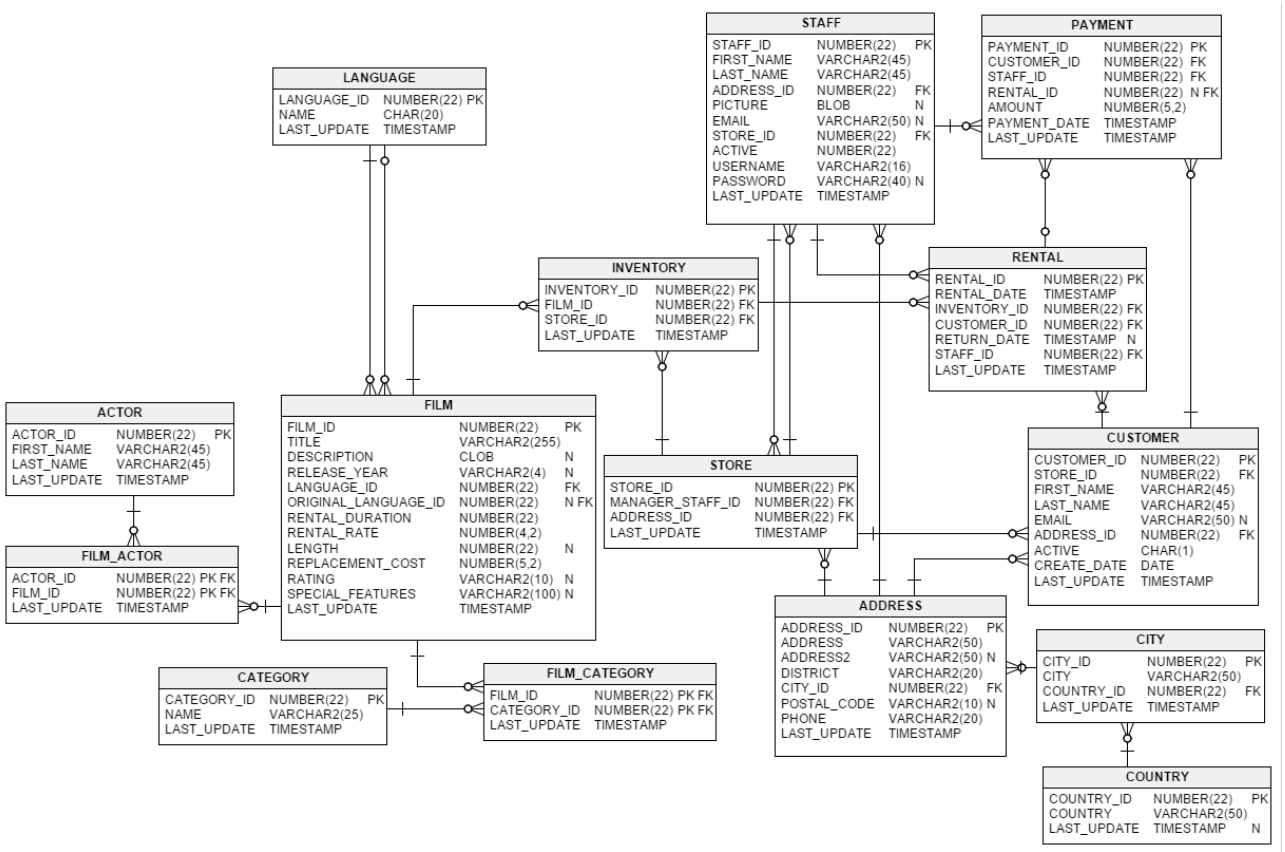
Es un lenguaje *"case insensitive"*, lo que significa que importa si escribimos en minúscula o mayúscula las consultas que ejecutemos. No obstante sí podría importar para los datos almacenados.

Es recomendable tener un explorador de bases de datos a mano.

- https://sqlitebrowser.org (solo para SQLite)
- https://www.dbvis.com

Vamos a usar https://sqliteonline.com/

El *esquema de la base de datos* que se cargó es el siguiente:



1. Sentencia `SELECT` + `FROM`

Permite seleccionar las columnas de una tabla específica de nuestra BD. Es decir, muestra todos los registros de la tabla, pero solo las columnas seleccionadas.

```
%%sql

SELECT <columnas separadas por coma>
FROM <nombre de la tabla>
```

Si queremos obtener todas las columnas de una tabla usamos un asterisco $*$ en lugar de poner los nombres de cada columna.

```
%%sql

SELECT *
FROM <nombre de la tabla>
```

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y verifique los resultados presentados:*00

```
%%sql

SELECT *
FROM film_list
LIMIT 3
```

El resultado esperado es:

FID	title	description	category	price	length	rating	actors
19	AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	Action	0.99	113	PG	JOHNNY LOLLOBRIGIDA
19	AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	Action	0.99	113	PG	JULIA MCQUEEN
19	AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	Action	0.99	113	PG	VAL BOLGER

2. Filtrando resultados con la cláusula `WHERE`

Si queremos filtrar de una tabla los registros que cumplan con una determinada condición debemos usar la cláusula WHERE dentro de una sentencia SQL.

Para eso debemos saber cómo expresar una condición que sea verdadera o falsa. Por ejemplo: quiero quedarme con todos los registros de una tabla que cumplan que el campo EDAD sea mayor a 18 años. Los registros que cumplan esta condición (registros con edades mayores a 18 años) se mostrarán dentro de los resultados.

```
%%sql

SELECT <columnas seperadas por coma>
FROM <nombre de la tabla>
WHERE <condición verdadera o falsa>
```

Tipos de condiciones para usar en la cláusula WHERE.

Condicionales

- > mayor
- >= igual o mayor
- = igual
- < menor

- <= menor o igual
- <> distinto

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%% sql

SELECT rental_id, amount, customer_id
FROM payment
WHERE customer_id = 599
LIMIT 3
```

El resultado esperado es:

rental_id	amount	customer_id
1008	4.99	599
2272	1.99	599
3043	6.99	599

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT title, description, price
FROM film_list
WHERE price <= 3
LIMIT 3
```

El resultado esperado es:

title	description	price
AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	0.99
AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	0.99
AMADEUS HOLY	A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon	0.99

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT title, description, category
FROM film_list
WHERE category = 'Family'
LIMIT 3
```

El resultado esperado es:

title	description	category
AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico	Family
AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico	Family
AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef And a Dentist who must Pursue a Forensic Psychologist in The Gulf of Mexico	Family

Un **rango** puede calcularse con **`BETWEEN a AND b`**

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT rental_id, return_date
FROM rental
WHERE return_date BETWEEN '2005-05-26' AND '2005-05-27'
LIMIT 3
```

El resultado esperado es:

rental_id	return_date
1	2005-05-26 22:04:30.000
14	2005-05-26 02:56:15.000
16	2005-05-26 04:42:11.000

La **inclusión** puede calcularse con `'IN (a, b, c)`.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT *
FROM sales_by_film_category
WHERE category IN ('Drama','Comedy')
LIMIT 3
```

El resultado esperado es:

category	total_sales
Comedy	4383.579999999894
Drama	4587.389999999873

Se puede definir una detección basada en **Patrón** con `LIKE expresión'`, usar `%` dentro de la expresión como comodín.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT customer_id, first_name, last_name
FROM customer
WHERE first_name LIKE 'John%'
LIMIT 3
```

El resultado esperado es:

customer_id	first_name	last_name
300	JOHN	FARNSWORTH
395	JOHNNY	TURPIN
571	JOHNNIE	CHISHOLM

¿Qué pasa si quiero filtrar varias condiciones al mismo tiempo?

Operaciones lógicas

- AND
- OR
- NOT
- XOR

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT film_id, title, rental_rate, replacement_cost
FROM film
WHERE rental_rate > 2 AND replacement_cost < 20
LIMIT 3
```

El resultado esperado es:

film_id	title	rental_rate	replacement_cost
2	ACE GOLDFINGER	4.99	12.99
3	ADAPTATION HOLES	2.99	18.99
6	AGENT TRUMAN	2.99	17.99

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT film_id, title, rental_rate
FROM film
WHERE NOT rental_rate > 2
LIMIT 3
```

El resultado esperado es:

film_id	title	rental_rate
1	ACADEMY DINOSAUR	0.99
11	ALAMO VIDEOTAPE	0.99
12	ALASKA PHANTOM	0.99

3. Algunas funciones especiales

Nombre	Función
`DISTINCT`	Permite obtener los valores distintos de una o más columnas.
`LOWER` / `UPPER`	Convierte una columna en minúsculas/mayúsculas.
`SUBSTR`	Sirve para extraer una parte de una cadena de texto.
`LENGTH`	Calcula el largo de los valores de una columna.
`DATE_FORMAT`	Para trabajar con fechas.
`SUM`	Sumatoria de los valores de una columna.
`COUNT`	Cuenta filas.
`AVG`	Promedio.
`MAX` / `MIN`	Máximo y mínimo.
`+`, `-`, `*`, `/`	Suma, resta, multiplicación, división.
`POWER`	Potencias.
`SQRT`	Raíz cuadrada.
`CONCAT`	Concatena cadenas de texto.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT COUNT(*) AS copias
FROM film_list
```

El resultado esperado es:

copias
5462

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT COUNT(DISTINCT(title)) AS películas_únicas
FROM film_list
```

El resultado esperado es:

películas_únicas
997

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT AVG(price) AS precio_promedio, SUM(length) AS duración_total
FROM film_list
```

El resultado esperado es:

precio_promedio	duración_total
2.957411204686741	629968

4. Ordenamiento

Para ordenar los resultados debemos utilizar la cláusula **`ORDER BY`** al final de la consulta, seguida de la/s columna/s que queramos ordenar.

También podemos especificar si el ordenamiento es ascendente (`ASC`) o descendente (`DESC`). Por default el orden es ascendente, por lo que no hace falta especificarlo.

```
%%sql

SELECT <columnas a seleccionar>
FROM nombre_de_tabla
ORDER BY <columnas para ordenar>
```

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%%sql

SELECT title, rental_duration, rental_rate
FROM film
ORDER BY rental_rate, rental_duration DESC
LIMIT 3
```

El resultado esperado es:

title	rental_duration	rental_rate
ANONYMOUS HUMAN	7	0.99
ARGONAUTS TOWN	7	0.99
BOONDOCK BALLROOM	7	0.99

Otra forma de ordenar los resultados de una consulta es especificando la posición de las columnas.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
%% sql

SELECT title, price, length
FROM film_list
ORDER BY 2, 3 DESC
LIMIT 3
```

El resultado esperado es:

title	price	length
THEORY MERMAID	0.99	184
THEORY MERMAID	0.99	184
THEORY MERMAID	0.99	184

5. SQL en Python

Primero hace falta crear un conector a la base datos.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
import sqlite3

conexión = sqlite3.connect('sakila.db')
```

Cada consulta se realiza por medio de un cursor.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
cursor = conexión.cursor()
cursor.execute('SELECT title, description FROM film_list LIMIT 7')
```

Luego hay que recorrer el cursor. Funciona como un casete: al recorrerlo, se agota. Para volver a usarlo es necesario volver a ejecutar la consulta.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
for fila in cursor:
    print(fila)
```

El resultado esperado es:

```
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
('AMERICAN CIRCUS',
'A Insightful Drama of a Girl And a Astronaut who must Face a Database Administrator in A Shark Tank')
```

El cursor tiene métodos para devolver todas las filas (`fetchall`) o devolverlas de a una (`fetchone`).

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
cursor.execute('SELECT title, description FROM film_list LIMIT 7')
cursor.fetchall()
```

El resultado esperado es:

```
[('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon'),
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon'),
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon'),
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon'),
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon'),
('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon'),
('AMERICAN CIRCUS',
'A Insightful Drama of a Girl And a Astronaut who must Face a Database Administrator in A Shark Tank')]
```

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
cursor.execute('SELECT title, description FROM film_list LIMIT 2')

cursor.fetchone()
```

El resultado esperado es:

```
[('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
```

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
cursor.fetchone()
```

El resultado esperado es:

```
[('AMADEUS HOLY',
'A Emotional Display of a Pioneer And a Technical Writer who must Battle a Man in A Baloon')
```

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
cursor.fetchone()
```

Devuelve `None` porque el cursor se agotó.

Es prolijo cerrar la conexión al terminar el programa.

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
conexion.close()
```

Tipos de datos en SQLite

- `None` → NULL
- `int` → INTEGER
- `float` → REAL
- `str` → TEXT
- `bytes` → BLOB

6. SQL con Pandas

Ejercicio de Inducción: *Pruebe las siguientes líneas de código y observe los resultados:*

```
import pandas as pd
import sqlite3

conexión = sqlite3.connect('sakila.db')

df = pd.read_sql('SELECT film_id, title, description FROM film', conexión)

df.head(3)
```

El resultado esperado es:

film_id	title	description
1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist...
2	ACE GOLDFINGER	A Astounding Epistle of a Database Administrat...
3	ADAPTATION HOLES	A Astounding Reflection of a Lumberjack And a ...