**Clase 03 - Lenguaje SQL**

# Concepto

Es un **lenguaje de consultas estructuradas** que responde a las siglas en inglés ***Structured Query Language.***

Nos permite acceder y manipular bases de datos.

Es popular por su facilidad de uso y efectividad para convertir grandes volúmenes de datos en información útil.

¿Qué nos permite hacer?

* **Ejecutar** consultas para recuperar datos.
* **Insertar**, **modificar** y **eliminar** registros.
* **Crear** bases de datos, tablas, procedimientos o vistas.
* Establecer **permisos en tablas**, **procedimientos** y **vistas.**

# Sentencias

Las **sentencias** (*también denominadas comandos o cláusulas*), son las **palabras reservadas** **para ejecutar acciones sobre la base de datos.**

En los [lenguajes informáticos](https://es.wikipedia.org/wiki/Lenguaje_inform%C3%A1tico), una **palabra reservada** es una palabra que tiene un significado [gramatical](https://es.wikipedia.org/wiki/Gram%C3%A1tica) especial para ese lenguaje y no puede ser utilizada como un identificador de objetos en códigos del mismo, como pueden ser las **variables**.

Por ejemplo, en [SQL](https://es.wikipedia.org/wiki/SQL), un usuario no puede ser llamado "*group*", porque la palabra group es usada para indicar que un identificador se refiere a un grupo, no a un usuario. Al tratarse de una palabra clave su uso queda restringido.

# Sintaxis

**Conjunto de reglas** **que deben seguirse** al escribir el código SQL para considerarse como **correctas** y así **completar la ejecución** exitosamente.

Las operaciones en SQL siguen una **estructura** que **describe** la **operación** que deseamos **realizar**. Si bien encontraremos consultas muy sencillas y otras más complejas, existen algunos elementos que aparecen con mayor frecuencia:

👉 **Acción** *(keyword)*: crear, seleccionar, insertar, actualizar, eliminar, etc.

👉 **Porción donde operaremos**: puede ser uno o más campos, una o más tablas, o un asterisco (\*) para todas.

👉 **Tabla/s a la cual/es queremos acceder**: la identificamos escribiendo el/los nombre/s.

👉 **Condiciones**: podemos establecer criterios para operar sólo sobre los registros que los cumplan.

Las consultas SQL nos permiten traer todos los campos de una tabla de datos, o determinados campos. Si deseamos traer determinados campos de una tabla, simplemente debemos mencionar los mismos uno al lado del otro, separándolos por una coma. Tal como se muestra en la primera línea de código de este ejemplo.

SELECT id\_class, description FROM class;

donde se seleccionan sólo 2 campos de la tabla de clases de video juegos del esquema que trabajaremos en todo el curso.

A tener en cuenta

Las **sentencias** SQL **no** son **sensibles** a las **mayúsculas y minúsculas**. No obstante, es importante **respetarlas** al colocar el **nombre** de un **campo o tabla**

Cada consulta finaliza con punto y coma (;)

# Consulta de Selección

## Uso de SELECT - FROM

La sentencia ***SELECT*,** como lo indica su nombre, permite **seleccionar información** a extraer y gracias a esto visualizar el resultado.

La cláusula **FROM** complementa al *SELECT*. Esta declara la/s **tabla/s desde la/s cual/es se va a extraer** la información.

Ejemplo:

**SELECT** id\_class, description

**FROM** class;

### Selección de determinados Campos

👉 **El orden de los campos en el SELECT es irrelevante.**

👉 Podemos definir el que necesitemos en primer lugar, independientemente de la posición donde éste haya sido definido cuando se creó la tabla.

👉 Al visualizar el resultado, el orden de los campos será tal como lo coloquemos en la consulta.

### Selección de todos los campos

El símbolo *asterisco* ( **\* )** juega el mismo papel que cuando lo usamos para buscar archivos o carpetas. Representa a “*todos*” los que existan.

Ejemplo:

SELECT \* FROM system\_user;

## SELECT-DISTINCT

La cláusula **DISTINCT** funciona en conjunción con **SELECT**. Permite filtrar de una consulta aquellos registros repetidos del resultado de la misma.

Para utilizar **la sentencia** **SELECT DISTINCT**, debemos incorporar siempre el nombre de al menos un campo de la tabla.

Con ese dato, SQL podrá resolver cómo aplicar correctamente la distinción de los registros recuperados.

Ejemplo:

**SELECT DISTINCT** id\_system\_user, first\_name

**FROM** system\_user;

Operadores SQL

Operadores de comparación

Los **operadores de comparación** en SQL nos **permiten evaluar una condición y determinar si el resultado es verdadero**, **falso o desconocido** (TRUE, FALSE o UNKNOWN)

### Sentencia WHERE

Cuando se aplica la cláusula WHERE en una consulta SQL se retornan todas las filas cuyo valor de verdad de dicha sentencia evalúa VERDADERO.

El objetivo principal es **aplicar un filtro sobre los datos almacenados en la tabla**, que cumplan con una cierta condición.

Esta es la lista de los principales operadores que podemos utilizar junto a la sentencia **WHERE**.

| **=** | *igual a* | **IS [NOT] NULL** | *no es nulo* | **BETWEEN** | *entre* |
| --- | --- | --- | --- | --- | --- |
| **<** | *menor a* | **NOT** | *NOT lógico* | **[NOT] BETWEEN** | *no esta entre* |
| **>** | *mayor a* | **LIKE** | *es como* | **IN** | *en (lista)* |
| **<=** | *menor o igual a* | **[NOT] LIKE** | *no es como* | **[NOT] IN** | *no esta en (lista)* |
| **=>** | *mayor o igual a* | **IS [NOT] TRUE** | *no es verdadero* | **IS [NOT] FALSE** | *no es falso* |
| **!= ó <>** | *distinto de* | **AND** | *AND lógico* | **OR** | *OR lógico* |

Importante

Las **sentencias** tienen un **orden** para su correcto funcionamiento

Cuando tenemos una **sintaxis** incorrecta, el **SGBD** nos presenta el error y en la mayoría de los casos lo hace de forma explícita

**1ºSELECT** (campos...)

**2ºFROM** (tabla)

**3ºWHERE** (condición/es)