



Nuestro compromiso es con el *futuro*.

Bases de Datos

Clase 4

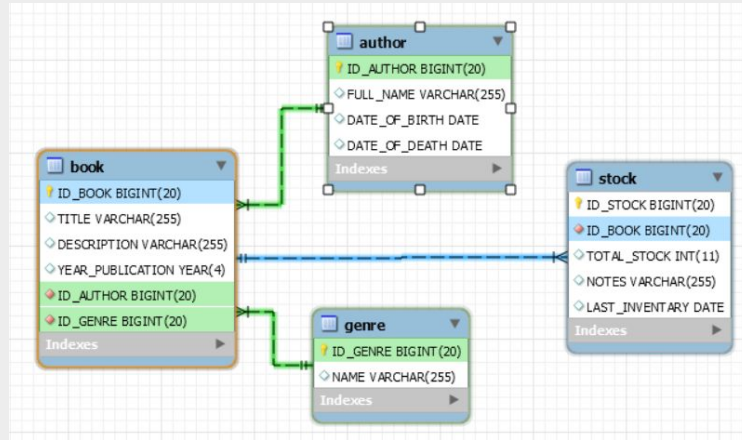
¿Qué veremos hoy?

Hoy vamos a continuar el módulo de **Bases de Datos**, vamos a seguir usando nuestra herramienta **Mysql Workbench**, dónde ya avanzamos con nuestro sistema de **inventario de libros**.

Adicional a lo mencionado, vamos a comenzar a ver que significan conceptos como DML y DDL y de qué forma se pueden leer datos de nuestras tablas.

Esquema a plantear

La clase pasada, trabajamos en el esquema que se muestra a continuación, insertando datos y creando nuestras tablas, entre otros conceptos que se vieron, usando **Mysql Workbench**, veamos el concepto de **DDL** a continuación.



DDL

El lenguaje de definición de datos (en inglés Data Definition Language, o **DDL**), es el que se encarga de la modificación de la estructura de los objetos de la base de datos. Incluye órdenes para modificar, borrar o definir las tablas en las que se almacenan los datos de la base de datos. Dentro de este grupo existen algunas operaciones muy conocidas como:

CREATE, ALTER, DROP

CREATE: Se crea el objeto (tabla), por ejemplo

```
CREATE TABLE table_name (  
    column1 datatype,  
    column2 datatype,  
    column3 datatype,  
    ....  
);
```

ALTER: Este comando permite modificar la estructura de una tabla u objeto, por ejemplo:

```
ALTER TABLE tbl_name  
    [alter_option [, alter_option] ...]  
    [partition_options]
```

DDL

DROP: DROP TABLE table_name, es decir, se especifica el nombre de la tabla, por ejemplo:

```
DROP TABLE 'ALUMNOS';
```

Además de lo que ya hemos visto, DDL, también podemos manipular o en su defecto insertar datos. Para esto, existen otro tipos de instrucciones/sentencias que podemos usar, las cuales forman parte de otro grupo llamado DML. Veremos a continuación de qué tratan las sentencias DML y cuales son.

DML

Las sentencias **DML** permite a los usuarios introducir datos para posteriormente realizar tareas de consultas o modificación de los datos que contienen las Bases de Datos.

Los elementos que se utilizan para manipular los datos, son los siguientes:

- **SELECT**, esta sentencia se utiliza para realizar consultas sobre los datos.
- **INSERT**, con esta instrucción podemos insertar los valores en una base de datos.
- **UPDATE**, sirve para modificar los valores de uno o varios registros.
- **DELETE**, se utiliza para eliminar las filas de una tabla.

A continuacion, veremos algunos ejemplos sobre estas sentencias....

Inserción de datos. INSERT

Vimos la clase pasada como trabajar con nuestro esquema e insertar datos en ella, repasemos como se usaba la sentencia INSERT

```
INSERT INTO `author` (`idauthor`, `date_of_birth`, `date_of_death`) VALUES (1,'1988-02-03',NULL);
```

Cuando usamos la sentencia insert, especificamos el nombre de la tabla, junto con los campos de la misma; dentro de VALUES agregamos que valores tendrán dichas columnas.

Similar con la tabla genre, quedaria :

```
INSERT INTO `genre` (`idGenre`, `name`) VALUES (1,'M'),(2,'F');
```

En este caso se dispone de dos valores, M y F con su respectivo id de valor 1 y 2 respectivamente.

Estos solo son ejemplos de cómo usar la sentencia INSERT, a continuacion

INSERCIÓN DATOS

```
INSERT INTO `book` (`idbook`, `title`, `description`, `year_of_publication`, `id_author`, `id_genre`) VALUES  
(1,'Icaro curso','modulo base de datos','2022-03-03',1,1);
```

La sentencia es similar a la que ya veníamos mostrando, pero al final, los valores **1** y **1**, nos indican la relación que tiene la tabla **book** con las tablas **genre** y **author**; expresando que :

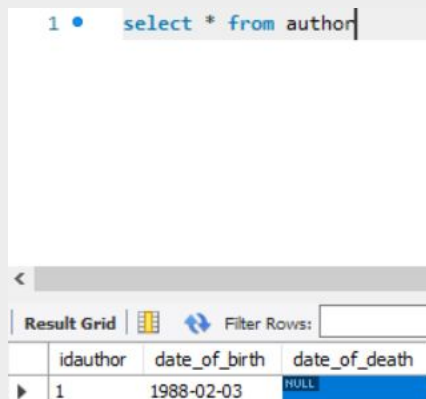
- 1) El registro de *id 1* de nuestra tabla **book** tendrá relación con el registro 1 de la tabla **author** y con el registro de *id 1* de la tabla **genre**

Se animan a insertar más datos para esta tabla?

Y para la tabla **stock**?

Obtención de datos. SELECT

Tomemos por ejemplo, a nuestra tabla author; como podemos hacer para obtener todos los usuarios de nuestra tabla?; si ejecutamos la sentencia **SELECT * FROM author**, obtendremos:

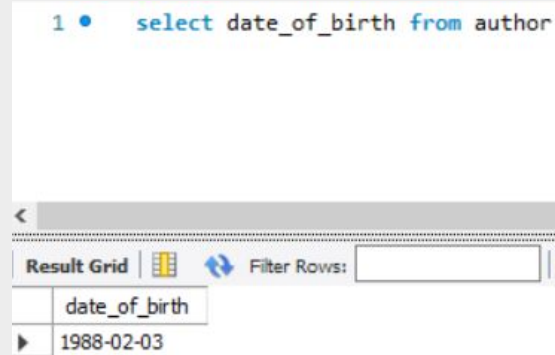


The screenshot shows the MySQL Workbench interface. At the top, the SQL editor contains the query `select * from author`. Below the editor, the 'Result Grid' tab is active, displaying the results of the query. The grid has three columns: `idauthor`, `date_of_birth`, and `date_of_death`. There is one row of data with the values 1, 1988-02-03, and NULL.

| | idauthor | date_of_birth | date_of_death |
|---|----------|---------------|---------------|
| ▶ | 1 | 1988-02-03 | NULL |

La sentencia fue ejecutada en nuestra herramienta Mysql Workbench. Podemos apreciar, que figura un ***** en nuestra sentencia. Esto significa que estamos queriendo visualizar todas las columnas que figuren en nuestra tabla. Como podríamos, entonces, solo obtener el campo 'date_of_birth' o fecha de nacimiento?

Obtención de datos. SELECT







Como podemos apreciar, solo obtenemos el campo solicitado, es decir, que tenemos que especificar el campo deseado dentro de la sentencia **SELECT**

En caso de tener varias filas dentro de nuestra, como podemos, por ejemplo obtener los autores cuyo nombre sea Juan?

Obtención de datos. SELECT

```
1 select * from author where name = 'Juan'
```


Result Grid |   Filter Rows: | Edit:  

| | idauthor | date_of_birth | date_of_death | name | lastname |
|---|----------|---------------|---------------|------|----------|
| ▶ | 1 | 1988-02-03 | NULL | Juan | Perez |
| * | NULL | NULL | NULL | NULL | NULL |

Introducimos una nueva sentencia, dentro de nuestro **SELECT**, el **WHERE**. Seguido del **WHERE** debemos especificar qué condición se debe cumplir; para este planteo en particular, queremos todos los autores cuyo nombre sea 'Juan'.

Ahora, podemos anadir mayor detalle a nuestra consulta. Podemos ordenar nuestros campos?

Obtención de datos. SELECT

```
1 select * from author
```

| | idauthor | date_of_birth | date_of_death | name | lastname |
|---|----------|---------------|---------------|---------|-----------|
| ▶ | 1 | 1988-02-03 | NULL | Juan | Perez |
| | 2 | 1987-03-22 | NULL | Luciano | Lopez |
| | 3 | 1987-03-11 | NULL | Augusto | Tamagnone |
| * | NULL | NULL | NULL | NULL | NULL |

```
1 select * from author order by name
```

| | idauthor | date_of_birth | date_of_death | name | lastname |
|---|----------|---------------|---------------|---------|-----------|
| ▶ | 3 | 1987-03-11 | NULL | Augusto | Tamagnone |
| | 1 | 1988-02-03 | NULL | Juan | Perez |
| | 2 | 1987-03-22 | NULL | Luciano | Lopez |
| | NULL | NULL | NULL | NULL | NULL |

En la imagen de la izquierda obtenemos todos los usuarios como ya hemos ejecutado antes. En la imagen de la derecha, introducimos una nueva sentencia, **ORDER BY**. Con esta última podemos ordenar los nombres, de forma alfabética, por eso, el autor de nombre Augusto, figura en primer lugar.

**No olvidemos que
disponemos de
clases de consulta!!**



ICARO Asociación Civil
CUIT 30716564815
info@icaro.org.ar
www.icaro.org.ar

Muchas gracias!



ICARO Asociación Civil
CUIT 30716564815
info@icaro.org.ar
www.icaro.org.ar