**Clase 06 - Sublenguaje DDL**

**Fundamentos de Data Definition Language**

En la primera clase aprendimos que SQL está conformado por cuatro sublenguajes aplicables en diferentes ámbitos de uso en la DB: **DDL, DML, DCL, TCL**

Hoy profundizaremos **DATA DEFINITION LANGUAGE - DDL**, para sacar partido de todos sus beneficios.

El **Lenguaje de Definición de Datos** se ocupa de **modificar la estructura de objetos de una DB**

Lo conforman **diferentes sentencias** que nos **permiten crear, modificar, borrar o definir la estructura** de las tablas que almacenan datos.

**Sentencias DDL**

Las sentencias disponibles a través de DDL, son:

* **CREATE**
* **ALTER**
* **DROP**
* **TRUNCATE**

Con ellas creamos, modificamos, alteramos y eliminamos objetos.

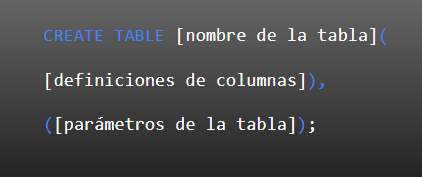
Si bien estas tareas se pueden, y suelen realizar a través de una herramienta de gestión como **Mysql Workbench**, el DDL permite en aquellos casos donde la DB oficia de motor de sistemas web o de gestión, generar muchas o todas estas tareas de forma automatizada.

**DDL: CREATE**

La sentencia **CREATE** cumple la función de **crear nuevos objetos en la base de datos**.

Los tipos de objetos a crear pueden ser: *tablas*, *índices*, *stored procedures* y hasta nuevas *bases de datos*, además, *usuarios* específicos.

Enfoquémonos en la creación de una tabla. Su sentencia sería 👇



**[nombre de la tabla]**: definimos el nombre distintivo de la tabla a crear. Ej: **friend**.

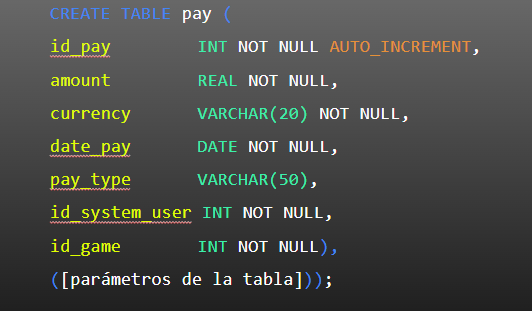
**[definiciones de columnas]**: definimos las columnas o campos y sus propiedades o tipo de datos.

**[parámetros de la tabla]**: definimos otras particularidades de la tabla como por ejemplo, los índices.

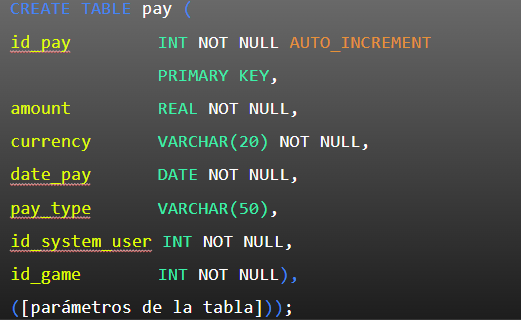
**Definición de columnas o campos**

Para **definir los campos** o columnas, **debemos indicar el nombre** que queremos para éste, **el tipo de dato** que contendrá (con o sin límite en cantidad de caracteres), y **si acepta o no valores nulos**.

**Definición de campos**



**Parámetros de tabla**

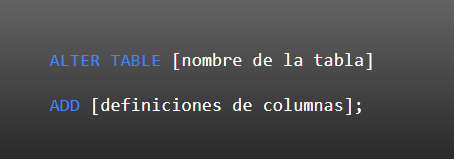
****

**DDL: ALTER**

La sentencia **ALTER** , del inglés “*alterar*”, permite modificar la estructura de una tabla u objeto de base de datos.

Con ella podemos agregar/quitar campos, modificar el tipo de datos de un campo y agregar o quitar índices.

Veamos a continuación cómo podemos agregar un campo a la una tabla ya creada 👇



**[nombre de la tabla]**: nombre de la tabla a alterar. Ejemplo: **friend**

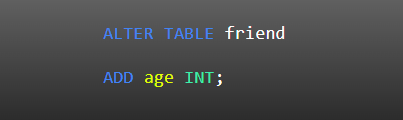
**ADD**: es la acción que realizaremos sobre la tabla. En este caso, agregar un nuevo campo.

**[definición de columna]**: Definimos la o las nuevas columnas, tal como hicimos con la sentencia **CREATE**.

**Parámetros de la tabla**

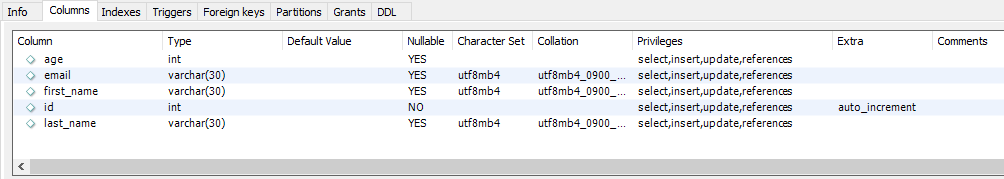
Agreguemos a continuación un campo en nuestra tabla **friend**.

El mismo se llamará **age** y será del tipo **INT**.



**Verificación de datos**

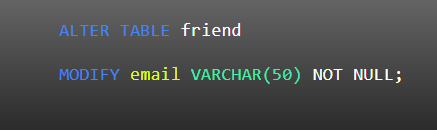
Verifiquemos a continuación mediante **TABLE INSPECTOR** que el nuevo campo se ha creado satisfactoriamente.



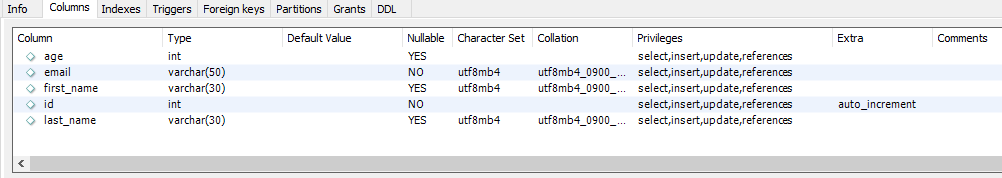


**ALTER TABLE [MODIFY]**

Si deseamos modificar los valores para un campo existente, debemos reemplazar la instrucción **ADD** por **MODIFY**.



**Verificación de datos**

****

Nuevamente podemos ver mediante **TABLE INSPECTOR**, que el campo **email** fue modificado satisfactoriamente

**Otras operaciones con Alter Table**

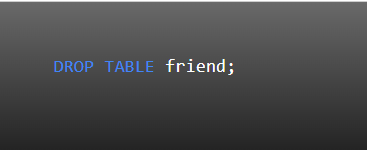
**CHANGE COLUMN**: podemos cambiar el nombre de una columna previamente definida.

**RENAME TO**: podemos cambiar el nombre inicial de una tabla por uno nuevo.

**DROP COLUMN**: podemos eliminar una columna o campo.

**DDL: DROP**

La sentencia **DROP**, del inglés “*borrar*”, nos permite eliminar una tabla

****

**Precauciones a tener en cuenta**

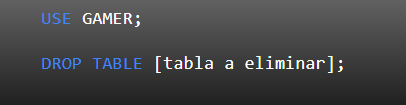
**DDL: DROP (TABLE)**

Dado que este tipo de actividad es poco frecuente, recomendamos siempre contemplar estas dos opciones:

* utilizar la sentencia USE “schema”.
* clonar la tabla si tiene datos y luego eliminarla.

**DDL: USE SCHEMA**

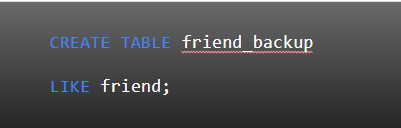
En entornos de trabajo muchas veces saltamos entre diferentes ambientes de DB de forma constante, lo que puede ocasionar que nos encontremos en el ambiente erróneo al momento de eliminar la tabla.

****

**DDL: CLONAR TABLA**

Si no estamos 100% seguros de si eliminar la tabla no afectará la necesidad de su información, podemos clonar, previamente, usando la sentencia:

**CREATE TABLE ... LIKE …**

****

**Tablas y relaciones**

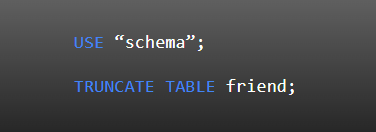
Las relaciones entre tablas son un factor dominante al momento del uso de **DROP TABLE**. Si la tabla tiene definida una o más relaciones con otras, primero deberemos eliminarlas para luego proceder con DROP.

*El motor de base de datos impedirá que podamos borrarla a menos que utilicemos la cláusula de DROP CASCADE*

**DDL: TRUNCATE**

**Sentencia DDL TRUNCATE**

La sentencia TRUNCATE TABLE elimina todos los datos que estén almacenados dentro de la tabla definida. Si bien existe la sentencia DELETE, propia de DML, TRUNCATE garantiza una mayor velocidad de borrado de datos.



**Tablas y relaciones**

Al igual que lo visto con la sentencia **DROP**, las relaciones entre tablas son un factor dominante al momento del uso de **TRUNCATE TABLE.**

No podremos borrar datos que estén vinculados a otros datos de otra tabla

**BUENAS PRÁCTICAS PERFORMANCE TUNING**

**1. Utilizar tipo de datos adecuado**

Debemos utilizar tipos de datos basados ​​en la naturaleza de los datos, ya que **si se utilizan tipos de datos irrelevantes, se puede consumir más espacio o generar errores.**

Ejemplo: el uso de varchar (20) para almacenar valores de fecha y hora en lugar del tipo de datos datetime dará lugar a errores durante los cálculos relacionados con la fecha y la hora y también existe un posible caso de almacenamiento de datos no válidos.

**2. Usar CHAR (1) sobre VARCHAR (1)**

Si escribimos un solo carácter, debemos usar CHAR (1) en lugar de VARCHAR (1) porque VARCHAR (1) tomará un byte adicional para almacenar información.

**3. Usar el tipo de datos CHAR para almacenar solo datos de longitud fija**

Ejemplo: el uso de char (1000) en lugar de varchar (1000) consumirá más espacio si la longitud de los datos es inferior a 1000.

**4. Evitar el uso de formatos de fecha regionales**

Cuando utilicemos el tipo de datos DATETIME o DATE, utilizar siempre el formato de fecha AAAA-MM-DD o el formato de fecha ISO que se adapte a su motor SQL. Otros formatos regionales como DD-MM-YYY, MM-DD-YYYY no se almacenarán correctamente.

**FUNCIONES ESCALARES**

Al igual que los lenguajes de programación en general, SQL incluye una serie de funciones denominadas **Funciones Escalares.**

**Permiten manipular datos cuando los recuperamos o antes de guardarlos**, mediante operaciones predeterminadas, devolviendo un resultado específico acorde a lo esperado.

Algunas ventajas de implementarlas, son:

* Reducir el re-trabajo de la lógica comercial.
* Evitar la inconsistencia de datos que provenga de un software.
* Ayudar a reducir el tráfico de red de aplicaciones cliente/servidor.
* Mejorar en gran medida el rendimiento de los sistemas.

Existen dos tipos de funciones escalares en Mysql:

* Funciones integradas.
* Funciones almacenadas.

Trabajaremos en esta instancia con las **funciones integradas.**

**Funciones Integradas**

Se clasifican bajo las siguientes categorías:

* Funciones de cadenas.
* Funciones numéricas.
* Funciones de fecha.
* Funciones agregadas.

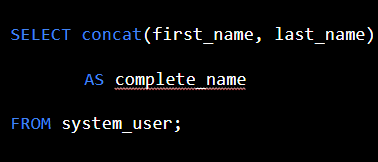
**Funciones de cadena**

Nos **permiten operar** con cualquier tipo de **cadena de caracteres** almacenada en una tabla (*o por almacenarse*). Podemos, entre otras cosas: *convertir el texto a mayúsculas, minúsculas, concatenar strings, cortar una porción del texto, eliminar espacios, revertir el texto, contar caracteres*, entre decenas de más funciones

**Ejemplo: Concat()**

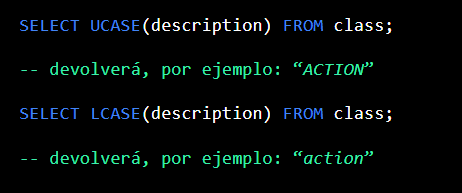
Fusiona cadenas de caracteres en un único bloque de datos.

Podemos, por ejemplo, unificar en un campo llamado **complete\_name**, los campos **first\_name** y **last\_name** de la tabla **SYSTEM\_USER.**

****

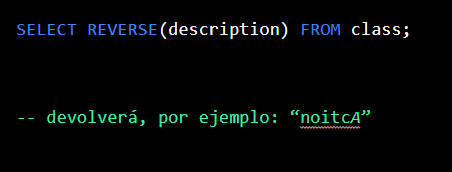
**Ejemplos: UCase() / LCase()**

Convierte a mayúsculas o minúsculas (*respectivamente*) una cadena de texto.

****

**Ejemplos: Reverse()**

Revierte el orden de los caracteres de una cadena de texto.

****

**Funciones de cadena**

Aquí tienes otras opciones para el manejo de caracteres:

* **TRIM()**: elimina los espacios vacíos en los extremos de un texto.
* **SPACE()**: cuenta la cantidad de espacios en un bloque de texto.
* **CHAR\_LENGTH()**: cuenta los caracteres de un bloque de texto.
* **SUBSTRING()**: extrae uno o más caracteres de un bloque de texto.
* Y muchas más…

**Funciones numéricas**

Nos **permiten operar** con cualquier tipo de **número**, a su vez, se subdividen en dos segmentos:

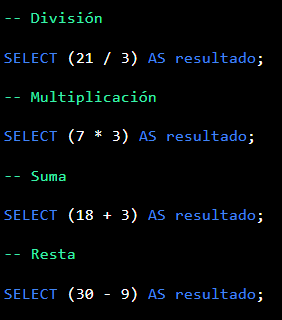
* **operadores aritméticos:** para realizar operaciones matemáticas básicas.
* **funciones matemáticas:** para realizar conversiones y otras operaciones con números de mayor complejidad.

**Ejemplo: operaciones aritméticas**

Podemos realizar operaciones aritméticas, utilizando la simbología común, a través de la estructura:

(**número** ***operador*** **número**).

El resultado se verá en un campo calculado.



**Ejemplos: funciones matemáticas**

* log(), log2(), log10(): cálculo de logaritmos, base 2 y base 10.
* round(): redondeo estándar de un número.
* floor(): redondeo de un número hacia abajo.
* ceiling(): redondeo de un número hacia arriba.
* truncate(): elimina los decimales de un número.

Más otras tantas funciones más. [Puedes consultarlas aquí](https://dev.mysql.com/doc/refman/8.0/en/mathematical-functions.html).

**Funciones de fecha**

Podemos manipular cualquier tipo de cálculo con fechas:

* Obtener los días ocurridos entre determinadas fechas.
* El número del día de un año.
* Extraer el mes, el año, o día de la fecha actual.
* Saber qué día de la semana fue una determinada fecha.

Algunas funciones de fecha:

* **curdate()**: devuelve la fecha actual.
* **curtime()**: devuelve la hora actual.
* **now()**: combina los dos anteriores en un resultado.
* **datediff()**: obtiene la diferencia de tiempo entre dos fechas.
* **dayname()**: Retorna el nombre del día de semana de una fecha determinada.

Más otras tantas funciones más que [puedes consultar aquí.](https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html)