Dia 12/03 clase 4

Tips

💡 Utilizar **;** al final de un bloque de código es una buena práctica.

💡 El uso de **( )** se debe realizar para realizar la declaración de inicio y fin del contenido de una Tabla.

💡 Si bien podemos generar una ventana de Query por cada bloque de código, es buena práctica realizar todos los **bloques de código** en un solo Query pero tomando en cuenta que debes marcar el **bloque de código** que queremos aplicar **Execute**.

Siempre que queramos trabajar sobre una base de datos, tenemos dos forma de hacerlo: 1. Con la siguiente sintaxis: ***use database*** *[Base de Datos]* 2. En la parte superior izquierda se debe indicar la base de datos que se está usando

Cada vez que aplicamos **EXECUTE** a un bloque de códigos y no hay observaciones, en la parte baja al medio aparecerá un mensaje: *"Commands completed successfully"*.

COMENTARIOS EN EL PROGRAMA: -- comentario –

ERRORES. Puntitos rojos a la derecha son indicadores de que algo esta mal. Puedo tener errores al escribir (en la sintaxis), en el orden de las palabras o en la repetición de comandos. Hay cosas que no se pueden volver a repetir.

Graphical user interface

Description automatically generated with medium confidence

CONOCER LO QUE HAY EN UNA BASE DE DATOS (resumen en vista)

Select \* from information\_schema.columns WHERE TABLE\_NAME='estudiantes'

NOTA: no es trabajo del analita de datos, crear bases de datos pero esta bueno tener una idea para poder entender luego como trabajo

* Conocer del lenguaje SQL
* Identificar sus principales objetos.
* Identificar e implementar las sentencias del sub lenguaje DDL.

**Lenguaje de definición de datos**, DDL por sus siglas en inglés. Este es el conjunto de sentencias que se encargan de la **definición de la base de datos y sus objetos**.

# **Creación base de datos**

**Funciones :CREATE, ALTER, DROP**

**Crear una base de datos con nombre “BD01”**

create database BD01

escribis y pones ejecutar. Si apretas ejecutar sin seleccionar nada en particular se ejecuta todo. Sino hay que seleccionarlo o apretar en el renglón con clic derecho y poner ejecutar

Graphical user interface

Description automatically generated with medium confidence

# **Creación de tablas**

**Crear una tabla con nombre “Table01”**

Uso la sentencia CREATE TABLE. Al crear una tabla, todo su contenido va entre paréntesis.

Creamos la tabla con un solo campo (columna) llamado “nombre” que contiene cosas varchar de hasta 100 caracteres

Tengo que indicar DONDE quiero que agregue la tabla porque puede haber mas de una base de datos

create table Table01

(Nombre varchar (100));

Graphical user interface, text, application

Description automatically generated

“Dbo” es la tabla

Puedo elegir la base de datos a mano o escribir la sentencia *use* nombrebasededatos

Graphical user interface, text

Description automatically generated

TIPOS DE DATOS

Graphical user interface, text

Description automatically generated

Creo una tabla con dos columnas. Indico siempre los tipo de datos, NO OLVIDAR. Una de las columnas es primary key. Lo indico también al crearla. Por defecto, para una PK, te pone no nulo. Para cualquier otra columna te pone null (por defecto).

Text

Description automatically generated

create table Table02

( N\_Cliente int PRIMARY KEY not null, Nombre varchar (200) );

Todas las tablas deben contener una PK, en algunos casos pueden tener más de una y estas se llamarían Claves candidatas. No reconoce entre mayúsculas o minúsculas pero es buena practica mayúscula para primary key.

create table Table03

(

ID int PRIMARY KEY not null,

NM\_Cliente int,

sexo varchar (2),

telefono int,

ID\_Edad int,

CONSTRAINT FK FOREIGN KEY (NM\_Cliente)

REFERENCES Table02 (N\_Cliente)

);

Para separar las columnas que va agregando le pone una COMA.

Para que una Tabla puede tener una FK siempre debe contar con una PK anteriormente definida

Text

Description automatically generated

**Crear una VISTA**

Son consultas previamente hechas, guardadas en nuestra base de datos. La vista permite aislar los datos con restricciones de seguridad (porque uno ve el resultado de la consulta sin nada mas, ni sabe como se llego a eso o que se filtro). Gracias a las vistas los datos se disponibilizan para los usuarios finales.

create view Cliente\_M

as

select

NM\_Cliente, sexo

from [dbo].[Table03]

aca le estoy diciendo que me cree una vista con las columnas “NM\_cliente” y “sexo” de la tabla 3. Tomar en cuenta que utilizamos las sentencias ***select*** y ***from***, las cuales generalmente siempre van juntas. Esta indica la tabla a consultar (***from***) y las columnas (***select***). Por otro lado, tomar en cuenta la sintaxis a colocar con ***from***.

Es como guardar un bloque de código ya armado, cuando lo quiero llamar escribo:Graphical user interface, application, Word

Description automatically generated

select \*from Cliente\_M

Si es que no se han creado datos esta visualización solo va a mostrar el encabezado de columnas. Pero en conclusión, te devuelve el resultado de una consulta realizada previamente.

**Borrar la vista Cliente\_M**

drop view Cliente\_M;

**Funcion “create”**

create schema RT01

dbo es el esquema que aparece por defecto, al parecer esto modifica el tipo de esquema

DUDAS; como hago si, una vez ya creada la tabla, quiero agregar otra columna dentro de esa tabla.



**Agregar columnas a tablas ya creadas**

alter table Table02

add Apellido varchar (200);

No olvidar que a la nueva columna se le debe indicar el tipo de dato

**Eliminar columnas en tablas**

**alter table Table02**

**drop column Apellido;**

**Renombrar columnas**

**EXEC sp\_rename**

**"Table02.Nombre", "NM";**

vamos a Ejecutar (EXEC) un procedimiento de SQL (sp\_rename). Se va a renombrar la columna Nombre por NM de la Tabla 02.

Modificar el tipo de dato de una columna **Sexo en la Table03**

alter table Table03

alter column sexo

varchar (20);

En este caso se ha modificado la cantidad de caracteres de 2 a 20 en la columna sexo, pero también puede aplicar para cambiar el tipo de dato.

**Eliminar Tabla**

drop table Table01;

En este caso se borrará la Tabla01 que se ha creado anteriormente. Recordar que para eliminar una Tabla que tiene referenciada una FK, primero se debe eliminar la Tabla que ha hecho referencia y luego la que se quiere.

# Datos en tablas

## **Insertar datos en la Tabla**

Insert into Cliente (N\_Cliente,Nombre,Sexo,telefono,ID\_Edad)

values (562, 'Fernando', 'M', 42568360, 3);

insertar en la tabla cliente, en las siguientes columnas, los siguientes datos. Estan de manera ordenada la columna y el dato a ingresar

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Sino también con orden diferente lo toma

Graphical user interface, application

Description automatically generated

Ingresar datos como vienen sin especificar columna

Insert into Cliente

values (398, 'Christian', 'M', 45786349, 1);

te los agrega en el orden en que estan las columnas armadas

Insert into Cliente

values (591, 'Jose', 'M', 45741242, 2),

581, 'Juan', 'M', 45786845, 4);

Si quieres registrar más de un conjunto de datos, solo le debes colocar una **,** y seguir aumentando valores.

## **Actualizar o modificar registros.**

**Actualizar un dato en la columna N\_Cliente de la Tabla Cliente**

--update nombredetabla set nombrecolumna = nuevodato where otracolumnaquedice = taldato—

The UPDATE statement is used to modify the existing records in a table

Be careful when updating records. If you omit the WHERE clause, ALL records will be updated!

UPDATE table\_name  
SET column1 = value1, column2 = value2, ...  
WHERE condition;

Por ej**:**

UPDATE Customers  
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'  
WHERE CustomerID = 1;

Graphical user interface, text, application

Description automatically generated

Update Cliente set Nombre = 'Carlos' where N\_Cliente = 562;

**Borrar los datos de una fila especificando un campo de la columna en la Tabla Cliente**

delete from Cliente where N\_cliente = 562;

borra toda la fila

delete from cliente

where N\_cliente = 591 or N\_Cliente = 581;

asi borras mas de una fila

delete from cliente;

Borras TODOS los datos de la tabla (pero no la tabla ojo)