# Gestión de bases de datos

## Sesión 02

**Econ. Leonel Heredia Altamirano**

**Diplomado de Business Intelligence**
**SQL Server 2019**

14 de agosto de 2022

# Creación de una base de datos

Una base de datos en SQL Server es una réplica de la base de datos *model*, ya que esta base proporciona páginas vaícas en las que se puede crear archivos lógicos dentro de la base de datos para una implementación. Se utiliza la sentencia DDL *Create* para la creación de la base de datos, la cual cuenta con el siguiente formato:

```
CREATE DATABASE <NombreBaseDatos>
GO
```

Especificando el archivo maestro:

```
CREATE DATABASE  <NombreBaseDatos>
ON PRIMARY (
NAME = 'NombreArchivoLógico',
FILENAME = 'NombreArchivoFísico',
SIZE = <Tamaño>,
MAXSIZE = <Tamaño>,
FILEGROWTH = <Tamaño>
```

Especificando el archivo de transacciones:

```
CREATE DATABASE  <NombreBaseDatos>
ON PRIMARY (
NAME = 'NombreArchivoLógico',
FILENAME = 'NombreArchivoFísico',
SIZE = <Tamaño>,
MAXSIZE = <Tamaño>,
FILEGROWTH = <Tamaño>
)
LOG ON (
NAME = 'NombreArchivoLogico',
FILENAME = 'NombreArchivoFísico',
SIZE = <Tamaño>,
MAXSIZE = <Tamaño>,
FILEGROWTH = <Tamaño>
)
GO
```

- **NombreBaseDatos:** Es el nombre de la nueva base de datos; deben ser únicos en un servidor y pueden tener hasta 128 caracteres, a menos que no se especifique ningún nombre lógico para el registro.
- **ON PRIMARY:** Especifica que la lista de archivos está asociada al grupo principal. Este grupo contiene toda las tablas del sistema de base de datos.
- **LOG ON:** Especifica que los archivos de registro de la base de datos (archivo de registro) se han definido explícitamente.
- **NAME:** Especifica el nombre lógico del archivo. Este archivo es utilizado para referenciar al archivo en las sentencias del Transact-SQL.
- **FILENAME:** Especifica el nombre de archivo en el sistema operativo. Se debe precisar la ruta de acceso y el nombre del archivo que el sistema operativo reconocerá.
- **SIZE:** Especifica el tamaño para el archivo. Cuando este parámetro no es especificado para un archivo de registro, SQL le asigna específicamente 1 MB, el mínimo predeterminado.

- **MAXSIZE:** Especifica el tamaño máximo de crecimiento del archivo; se pueden utilizar sufijos KB y MB; el valor predeterminado es MB; solo se pueden especificar número enteros.
- **FILEGROWTH:** Especifica el incremento de crecimiento del archivo; este valor no puede exceder el valor de MAXSIZE.

Por buenas prácticas, siempre será bueno validar la existencia de la base de datos usando la siguiente sentencia:

```
IF DB_ID('<NombreBaseDatos>') IS NOT NULL
DROP DATABASE <NombreBaseDatos>
GO
```

La sentencia IF permite condicionar la existencia de la base de datos siempre y cuando la función DB_ID emita el valor FALSE, se le asigna una comparación IS NOT NULL para validar la existencia de la base de datos. Entonces, se podría decir que, si la base de datos existe, se ejecuta la sentencia DROP DATABASE que eliminará la base.

Para agregar, modificar o eliminar archivos contenidos en la base de datos, se usará la sentencia DDL *Alter* para la modificación o alteración de los archivos contenidos en una base de datos.
Por ejemplo, para agregar un archivo:

```
ALTER DABATASE <NombreBaseDatos>
ADD FILE (
    NAME = <NombreLogicoArchivo>,
    FILENAME = <NombreFisicoArchivo>,
    SIZE = <Tamaño>,
    MAXSIZE = <Tamaño>,
    FILEGROWTH = <Tamaño>
)
GO
```

Para eliminar archivos:

```
ALTER DATABASE <NombreBaseDatos>
REMOVE FILE <NombreArchivo>
GO
```

Para modificar las propiedades de los archivos:

```
ALTER DATABASE <NombreBaseDatos>
MODIFY FILE (
    NAME = <NombreArchivo>,
    PROPIEDAD = Valor
)
GO
```

Cuando se ejecutan sentencias, estas siempre se realizan sobre una base de datos activa. Eso quiere decir que dichas sentencias afectarán toda vez a los objetos contenidos en la base de datos. El formato de activación es:

```
USE <NombreBaseDatos>
GO
```

# Tipos de datos en SQL

## Tipo caracter

Es una combinación de letras, número y caracteres.

1. **Char:** Tipo de datos caracter de longitud fija, tiene como requisito indispensable la especificación del total de caracteres permitidos.

```
DNI CHAR(8)
SEXO CHAR(1)
```

2. **Varchar:** Tipos de datos de longitud variable, es decir, el valor asignado no es fijo. SQL administra los espacios en blanco y los optimiza.

```
DIRECCIÓN VARCHAR(50)
EMAIL VARCHAR(60)
```

3. **Text:** Tipo de datos de longitud extensa el cual permite almacenar información masiva inclusive que excedan los 8 KB.

```
COMENTARIO TEXT
```

## Tipo decimal y entero

El tipo decimal consiste en almacenar valores numéricos con fracción decimal. Dentro de este tipo tenemos:

1. **Decimal:** Pueden tener hasta 38 dígitos, podrán estar en el lado derecho del punto decimal.

```
PROMEDIO DECIMAL (5,2)
```

El tipo entero consiste en almacenar valores numéricos sin fracción decimal.

1. **Int:** Permite declarar valores enteros.

```
EVALUACIÓN INT
```

SQL nos permite agregar nuevos tipos de datos llamados *Tipos de datos definidos por el usuario*, los cuales podrán ser usados como un tipo de datos dentro de la creación o modificación de una tabla de base de datos.

# Gestión de tablas

Las sentencias de definición de datos para tablas permitirán agregar, modificar o eliminar tablas dentro de una base de datos activa. Las tablas son consideradas como objetos de la base de datos y los principales componentes dentro del sistema de datos; asimismo son las que guardan información que resulta importante en una base de datos.

La tabla representa una entidad dentro de la base de datos; por lo tanto, se deberán crear las tablas necesarias para un sistema de base; asimismo, no podrá existir duplicidad de tablas y no existirán tablas que no se asocien a otras.

```
CREATE TABLE <NombreTabla> (
    CAMPO1 TIPO,
    CAMPO2 TIPO,
    CAMPO3 TIPO
)
GO
```

Cuando la tabla es creada por primera vez dentro de la base de datos, no habrá problemas de validación, pero si se crea o modifica una tabla, será necesario controlar la existencia:

```
IF OBJECT_ID('NombreTabla')IS NOT NULL
    DROP TABLE 'NombreTabla'
GO
CREATE TABLE <NombreTabla> (
    CAMPO1 TIPO,
    CAMPO2 TIPO,
    CAMPO3 TIPO
)
GO
```

Cuando se ejecuta la sentencia *CREATE TABLE* de manera correcta, se puede encontrar la lista de tablas de la base de datos activa:

```
SP_TABLES
GO
```

Para visualizar la estructura de una tabla, se podrá observar el nombre de la tabla, el nombre de las columnas y la capacidad de almacenamiento de cada campo.

```
SP_COLUMNS 'NombreTabla'
GO
```

La modificación de la estructura de la tabla se realiza con la sintaxis:

```
ALTER TABLE <NombreTabla>
    ALTER COLUMN <Columna TIPO>
GO
```

Para agregar un campo se utiliza la sintaxis:

```
ALTER TABLE <NombreTabla>
    ADD ColumnaNueva1 TIPO,
        ColumnaNueva2 TIPO
GO
```

Para eliminar un campo se utiliza:

```
ALTER TABLE <NombreTabla>
    DROP COLUMN <Columna>
GO
```

Eliminar una tabvla conlleva también eliminar toda la información que contenga. El método básico es:

```
DROP TABLE <NombreTabla>
GO
```

El método de comprobación para eliminar es:

```
IF OBJECT_ID('NombreTabla')IS NOT NULL
    DROP TABLE 'NombreTabla'
GO
```

Cuando se crea una tabla de base de datos, se debe especificar si el campo podrá recibir valores nulos o no. Por ejemplo, los campos o llaves siempre serán de restricción no nula, pues no existen llaves vacías; en otros campos, se debe analizar si será necesario dejarlo vacío o no, como en el caso de que algún cliente no cuente con correo electrónico.

El formato de restricción nulo o no nulo de tablas es:

```
CREATE TABLE <NombreTabla> (
    CAMPO1 TIPO NULL|NOT NULL,
    CAMPO2 TIPO NULL|NOT NULL,
    CAMPO3 TIPO NULL|NOT NULL
)
GO
```

Si modificamos la tabla con restricciones nulos o no:

```
ALTER TABLE <NombreTabla>
    ALTER COLUMN TIPO NULL|NOT NULL
GO
```

Las restricciones de integridad permiten asignar llaves primarias o foráneas a una tabla; es necesario dicha referencia para que las tablas se logren asociar a otros, generando la integridad referencial entre tablas.

La llave principal o PRIMARY KEY restringe el ingreso de valor a datos únicos, el cual permite identificar un registro frente a los demás. Por buenas prácticas, el tipo de datos de un campo clave debe ser número entero (Int) o de caracter (Char).

```
CREATE TABLE <NombreTabla>(
    CAMPOCLAVE TIPO NOT NULL PRIMARY KEY,
    CAMPO2 TIPO NULL | NOT NULL,
    CAMPO3 TIPO NULL | NOT NULL
)
GO
```

Para la asignación de campo clave a partir de la modificación de tablas:

```
ALTER TABLE <NombreTabla>
    ADD PRIMARY KEY (CAMPOCLAVE)
GO
```

El formato de asignación de campo clave compuesto a partir de:

```
CREATE TABLE <NombreTabla>(
    CAMPO1 TIPO NOT NULL,
    CAMPO2 TIPO NULL | NOT NULL,
    CAMPO3 TIPO NULL | NOT NULL,
    PRIMARY KEY (<CAMPOCLAVE1>,<CAMPOCLAVE2>)
)
GO
```

El formato de asignación de campo clave compuesto a partir de la modificación de la tabla:

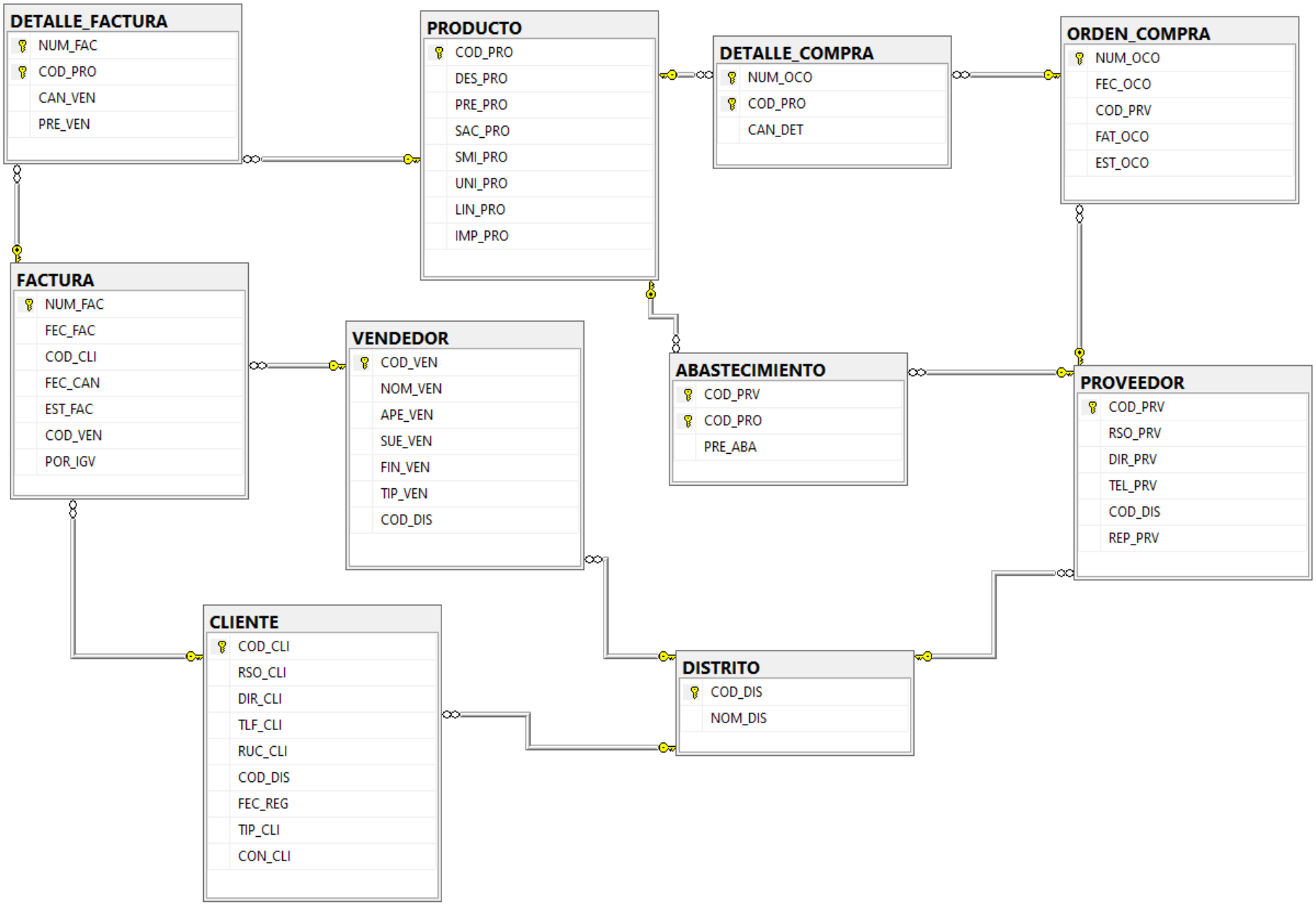
```
ALTER TABLE <NombreTabla>
    ADD PRIMARY KEY (CAMPOCLAVE1, CAMPOCLAVE2)
GO
```

La clave secundaria, campo foráneo o llave foránea, permitirá asociar una tabla a otra, desde la cual se podrá asignar a una columna dicha restricción. El formato de asignación de llave secundaria desde la creación:

```
CREATE TABLE <NombreTabla1>(
    CAMPO1 TIPO NULL | NOT NULL
    CAMPO2 TIPO NOT NULL REFERENCES <NombreTabla2>,
    CAMPO3 TIPO NULL |NOT NULL
)
GO
```

Otra forma de asignación de llave secundaria es:

```
CREATE TABLE <NombreTabla1>(
    CAMPO1 TIPO NULL | NOT NULL
    CAMPO2 TIPO NOT NULL,
    CAMPO3 TIPO NULL |NOT NULL,
    FOREIGN KEY (CAMPOSECUNDARIO)
        REFERENCES <NombreTabla2>
)
GO
```


DETALLE_FACTURA
NUM_FAC
COD_PRO
CAN_VEN
PRE_VEN
PRODUCTO
COD_PRO
DES_PRO
PRE_PRO
SAC_PRO
SMI_PRO
UNI_PRO
LIN_PRO
IMP_PRO
DETALLE_COMPRA
NUM_OCO
COD_PRO
CAN_DET
ORDEN_COMPRA
NUM_OCO
FEC_OCO
COD_PRV
FAT_OCO
EST_OCO
FACTURA
NUM_FAC
FEC_FAC
COD_CLI
FEC_CAN
EST_FAC
COD_VEN
POR_IGV
VENDEDOR
COD_VEN
NOM_VEN
APE_VEN
SUE_VEN
FIN_VEN
TIP_VEN
COD_DIS
ABASTECIMIENTO
COD_PRV
COD_PRO
PRE_ABA
PROVEEDOR
COD_PRV
RSO_PRV
DIR_PRV
TEL_PRV
COD_DIS
REP_PRV
CLIENTE
COD_CLI
RSO_CLI
DIR_CLI
TLF_CLI
RUC_CLI
COD_DIS
FEC_REG
TIP_CLI
CON_CLI
DISTRITO
COD_DIS
NOM_DIS

La propiedad *Identity* solo es aplicable a columnas de tipo numérico, ya que se define un autoincremento de valores que pueden representar una numeración de valores en forma automática por cada registro dentro de la tabla. Por ejemplo, podría tratarse de una tabla de facturas y su columna NUMFACTURA tenga asignado la propiedad *Identity* para identificar el número de factura registrada.

```
CREATE TABLE <NombreTabla>(
    CAMPO1 TIPO NOT NULL PRIMARY KEY IDENTITY(VALOR1,VALOR2),
    CAMPO2 TIPO NULL | NOT NULL,
    CAMPO3 TIPO NULL | NOT NULL
)
GO
```

El primer valor de *Identity* determina el punto de inicio y el segundo valor indica la forma de incremento.

La cláusula *Default* permite asignar un determinado valor por defecto según el tipo de datos; dicho valor debe ser especificado antes de registrar valores a la tabla. El formato de *Default* es:

```
CREATE TABLE <NombreTabla>(
    CAMPO1 TIPO NOT NULL PRIMARY KEY,
    CAMPO2 TIPO NULL | NOT NULL DEFAULT <Valor>,
    CAMPO3 TIPO NULL | NOT NULL DEFAULT <Valor>
)
GO
```

El formato de asignación *Default* en la modificación de tabla:

```
ALTER TABLE <Nombre_Tabla>
    ADD DEFAULT <Valor>
    FOR <Campo>
```

La cláusula *Check* permite restringir el rango de valores que pueden estar permitidos ingresar en una o más columnas de una tabla. Se debe tener en cuenta que *Check* evalúa el valor a registrar en la tabla; por lo tanto, se debe implementar una condición sobre cada campo evaluado.

```
CREATE TABLE <NombreTable>(
    CAMP01 TIPO NOT NULL PRIMARY KEY,
    CAMP02 TIPO NOT NULL CHEK <Condición>,
    CAMP03 TIPO NOT NULL CHEK <Condición>
)
GO
```

El formato de asignación en modificación:

```
ALTER TABLE <NombreTabla>
    ADD CONSTRAINT <Nombre>
    CHECK <Condición>
```

La cláusula *Unique* permite determinar que los valores registrados en una misma columna no sean idénticos, es decir, se mantengan únicos.

```
CREATE TABLE <NombreTabla>(
    CAMPO1 TIPO NOT NULL PRIMARY KEY,
    CAMPO2 TIPO NULL | NOT NULL UNIQUE,
    CAMPO3 TIPO NULL | NOT NULL UNIQUE
)
GO
```

El formato de asignación en modificación:

```
ALTER TABLE <NombreTabla>
    ADD CONSTRAINT <Nombre>
    UNIQUE <Campo>
```

# Ejercicio

De la base de datos PROYECTOSINDUSTRIALES, realizar las siguientes modificaciones:

- En el correo del cliente, mostrar el mensaje “No cuenta” si el usuario no registra dicho dato.
- En el monto del proyecto, el valor debe ser superior a cero.
- En la carrera del encargado, solo se podrán registrar valores como “Jefe-Contador” y “Supervisor-Vendedor”
- El número de RUC del cliente debe ser un valor único por cliente.
- Modificar la capacidad de caracteres a 100 de la descripción del detalle de proyecto.
- Agregar el campo telefónico “TEL_CLi” y número de hijos “HIJ_CLI” a la tabla CLiente.
- Eliminar el campo “HIJ_CLI” de la tabla Cliente.