	UNIVERSIDAD DON BOSCO **FACULTAD DE ESTUDIOS TECNOLOGICOS** **ESCUELA DE COMPUTACION**
CICLO 01-2016	**GUIA DE LABORATORIO Nº 4** **Nombre de la practica:** Creación de Base de datos relacional con Transact – SQL **Lugar de ejecución:** Laboratorio de Informática **Tiempo estimado:** 2 horas y 30 minutos **Materia:** Base de datos

I. Objetivos

1. Crear una base de datos con Lenguaje SQL
2. Crear tablas y definir tipos de datos con lenguaje SQL
3. Implementar la integridad referencial en una base de datos

II. Introducción Teórica

Lenguaje SQL

SQL (Structured Query Language), Lenguaje Estructurado de Consulta es el lenguaje utilizado para definir, controlar y acceder a los datos almacenados en una base de datos relacional.

Como ejemplos de sistemas gestores de bases de datos que utilizan SQL podemos citar DB2, SQL Server, Oracle, MySql, Sybase, PostgreSQL o Access.

El SQL es un lenguaje universal que se emplea en cualquier sistema gestor de bases de datos relacional. Tiene un estándar definido, a partir del cual cada sistema gestor ha desarrollado su versión propia.

En SQL Server la versión de SQL que se utiliza se llama TRANSACT-SQL.

Se puede ejecutar directamente en modo interactivo, pero también se suele emplear embebido en programas escritos en lenguajes de programación convencionales. En estos programas se mezclan las instrucciones del propio lenguaje (denominado anfitrión) con llamadas a procedimientos de acceso a la base de datos que utilizan el SQL como lenguaje de acceso. Como por ejemplo en Visual Basic, Java, C#, PHP .NET, etc.

Las instrucciones SQL se clasifican según su propósito en tres grupos:

- El DDL (Data Description Language) Lenguaje de Descripción de Datos.
- El DCL (Data Control Language) Lenguaje de Control de Datos.
- El DML (Data Manipulation Language) Lenguaje de Manipulación de Datos.

El DDL, es la parte del SQL dedicada a la definición de la base de datos, consta de sentencias para **definir la estructura de la base de datos**, permiten crear la base de datos, crear, modificar o eliminar la estructura de las tablas, crear índices, definir reglas de validación de datos, relaciones entre las tablas, etc. Permite definir gran parte del nivel interno de la base de datos. Por este motivo estas sentencias serán utilizadas normalmente por el administrador de la base de datos.

El DCL (Data Control Language) se compone de instrucciones que permiten:

Ejercer un control sobre los datos tal como la asignación de privilegios de acceso a los datos (GRANT/REVOKE).

El DML se compone de las **instrucciones para el manejo de los datos**, para insertar nuevos datos, modificar datos existentes, para eliminar datos y la más utilizada, para recuperar datos de la base de datos. Veremos que una sola instrucción de recuperación de datos es tan potente que permite recuperar datos de varias tablas a la vez, realizar cálculos sobre estos datos y obtener resúmenes.

El DML interactúa con el nivel externo de la base de datos por lo que sus instrucciones son muy parecidas, por no decir casi idénticas, de un sistema a otro, el usuario sólo indica lo que quiere recuperar no cómo se tiene que recuperar, no influye el cómo están almacenados los datos.

Es el lenguaje que utilizan los programadores y los usuarios de la base de datos.

La gestión de transacciones (COMMIT/ROLLBACK).

Una transacción se puede definir como un conjunto de acciones que se tienen que realizar todas o ninguna para preservar la integridad de la base de datos.

Por ejemplo supongamos que tenemos una base de datos para las reservas de avión. Cuando un usuario pide reservar un lugar en un determinado vuelo, el sistema tiene que comprobar que queden lugares libres, si quedan lugares reservará el que quiera el usuario generando un nuevo boleto y marcando el lugar como ocupado. Aquí tenemos un proceso que consta de dos operaciones de actualización de la base de datos (crear una nueva fila en la tabla de boletos y actualizar el lugar reservado en el vuelo, poniéndolo como ocupado) estas dos operaciones se tienen que ejecutar o todas o ninguna, si después de crear el boleto no se actualiza el lugar porque se cae el sistema, por ejemplo, la base de datos quedaría en un estado inconsistente ya que el lugar quedaría como libre cuando realmente habría un boleto emitido para este lugar. En este caso el sistema tiene el mecanismo de transacciones para evitar este error. Las operaciones se incluyen las dos en una misma transacción y así el sistema sabe que las tiene que ejecutar las dos, si por lo que sea no se pueden ejecutar las dos, se encarga de deshacer los cambios que se hubiesen producido para no ejecutar ninguna.

Las instrucciones que gestionan las autorizaciones serán utilizadas normalmente por el administrador mientras que las otras, referentes a proceso de transacciones serán utilizadas también por los programadores.

No todos los sistemas disponen de ellas.

Referencia de Transact-SQL (motor de base de datos)

Transact-SQL es fundamental para trabajar con SQL Server. Todas las aplicaciones que se comunican con SQL Server lo hacen enviando instrucciones Transact-SQL al servidor, independientemente de la interfaz de usuario de la aplicación.
A continuación se proporciona una lista de las aplicaciones que pueden generar Transact-SQL:

- Aplicaciones generales de productividad en oficinas.
- Aplicaciones que utilizan una interfaz gráfica de usuario (GUI) para permitir al usuario seleccionar las tablas y columnas cuyos datos desea ver.

- Aplicaciones que utilizan instrucciones del lenguaje general para determinar los datos que el usuario desea ver.
- Aplicaciones de la línea de negocios que almacenan sus datos en bases de datos SQL Server. Estas aplicaciones pueden incluir aplicaciones de otros proveedores o escritas internamente.
- Scripts Transact-SQL que se ejecutan con herramientas tales como **sqlcmd**.
- Aplicaciones creadas con sistemas de desarrollo tales como Microsoft Visual C++, Microsoft Visual Basic o Microsoft Visual J++, y que usan API de base de datos tales como ADO, OLE DB y ODBC.
- Páginas web que extraen datos de bases de datos SQL Server.
- Sistemas de bases de datos distribuidos desde los que se replican datos SQL Server en varias bases de datos o se ejecutan consultas distribuidas.
- Almacenamientos de datos en los que los datos se extraen de los sistemas de procesamiento de transacciones en línea (OLTP) y se resumen para el análisis dirigido a la toma de decisiones.

Instrucción CREATE

Vamos a examinar la estructura completa de la sentencia CREATE empezando con la más general. Descubrirá que las instrucciones CREATE empiezan de la misma forma y después dan paso a sus especificaciones. La primera parte de CREATE será siempre igual:

CREATE <tipo de objeto> <nombre del objeto>

A esta parte le seguirán los detalles, que variarán según la naturaleza del objeto que estemos creando. A continuación se presenta un listado de sintaxis más completa de CREATE:

CREATE DATABASE

Crea una nueva base de datos y los archivos que se utilizan para almacenar la base de datos

Sintaxis:

```
CREATE DATABASE <nombre de base de datos>
 [ON [PRIMARY]
   ([NAME = <nombre lógico del archivo>,]
    FILENAME = <'nombre del archivo'>
   [, SIZE = <tamaño en Kilobytes, megabytes, gigabytes, o terabytes>]
   [,  MAXSIZE = <tamaño en Kilobytes, megabytes, gigabytes, o terabytes>]
   [, FILEGROWTH = <tamaño en Kilobytes, megabytes, gigabytes, o terabytes| porcentaje>] ) ]
 [LOG ON
([NAME = <nombre lógico del archivo>,]
    FILENAME = <'nombre del archivo'>
   [, SIZE = <tamaño en Kilobytes, megabytes, gigabytes, o terabytes>]
   [,  MAXSIZE = <tamaño en Kilobytes, megabytes, gigabytes, o terabytes>]
   [, FILEGROWTH = <tamaño en Kilobytes, megabytes, gigabytes, o terabytes| porcentaje>] ) ]
 [ COLLATE <nombre de intercalación> ]
```

Tenga en cuenta que algunas de las opciones anteriores son mutuamente excluyentes (por ejemplo, si está creando para anexar, la mayoría de las opciones que no sean ubicaciones de archivo no serán válidas). En esta sintaxis hay mucho que explicar, por lo que vamos a desglosarla en sus elementos.

ON:

ON se utiliza en dos sitios para definir la ubicación del archivo donde se almacenan los datos (Archivo .MDF) y para definir la misma información para el lugar donde se guarda el registro (Log de transacciones, archivo .LDF). Advertirá aquí la inclusión de la palabra clave PRIMARY, que indica que lo que sigue es un grupo de archivos primarios (o principales) en el que se guardan físicamente los datos. También podemos guardar datos en los denominados grupos de archivos secundarios.

NAME:

Este es el nombre del archivo que estamos definiendo, pero sólo es un nombre lógico, es decir, el nombre que va a utilizar SQL Server internamente para hacer referencia a dicho archivo.

FILENAME:

Este es el nombre físico del disco del archivo del sistema operativo real en el que se van a guardar los datos y el registro (Log de transacciones). El valor predeterminado dependerá de si estamos tratando con la propia base de datos o con el Log de transacciones. De forma predeterminada, el archivo se ubicará en el siguiente subdirectorio DATA dentro del directorio C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\MSSQL\DATA o el directorio que haya establecido como principal para SQL Server en la instalación. Si estuviésemos utilizando el archivo de base de datos físico, se denominaría igual que nuestra base de datos con una extensión .mdf y si estuviésemos utilizando el registro, tendría el mismo nombre que el archivo de base de datos, pero con un sufijo _log y una extensión .ldf.

SIZE:

Es el tamaño de la base de datos, de forma predeterminada el tamaño se proporciona en megabytes, pero también podemos hacer que se proporcione en kilobytes utilizando KB en lugar de MB tras el valor numérico del tamaño; también podemos usar un tamaño mucho mayor con GB (gigabytes) o incluso TB (terabytes). Tenga en cuenta que este valor debe ser, al menos, tan alto como el de la base de datos model y debe ser un número entero (sin decimales); en caso contrario recibiremos un error. Si no suministramos un valor para SIZE, la base de datos tendrá inicialmente el mismo tamaño que el de la base de datos model.

MAXSIZE:

Este parámetro es una pequeña variante del parámetro SIZE. SQL Server tiene un mecanismo que permite a nuestra base de datos asignar automáticamente un espacio en disco adicional (para crecer) cuando sea necesario. MAXSIZE es el tamaño máximo al que puede crecer la base de datos. Una vez más, de forma predeterminada se proporciona en megabytes, como SIZE, podemos utilizar KB, GB o TB para emplear cantidades de incremento diferentes. La pequeña variante es que no existe un valor predeterminado firme.

Si nuestra base de datos (el archivo .mdf) llega al valor establecido en el parámetro MAXSIZE, nuestros usuarios empezarán a recibir errores indicando que sus inserciones no se pueden ejecutar. Si nuestro registro (Log de transacciones .ldf) llega a su tamaño máximo, no podremos ejecutar ninguna actividad de inicio de sesión en la base de datos.

FILEGROWTH:

Mientras SIZE establece el tamaño inicial de la base de datos y MAXSIZE determina exactamente el tamaño máximo que puede llegar a tener el archivo de la base de datos, FILEGROWTH determina básicamente el incremento del crecimiento con que se puede llegar a dicho máximo. Para ello, proporcionamos un valor indicando por cuantos bytes (en KB, MB, GB, o TB) a la vez deseamos aumentar el archivo. Por ejemplo, si establecemos un archivo de base de datos para que cuando llegue a 1GB incremente en un valor FILEGROWTH de un 20%, la primera vez que se expanda, aumentará hasta a 1.2GB, la segunda vez hasta 1.44GB y así sucesivamente.

LOG ON:

La opción LOG ON nos permite establecer que deseamos que nuestro registro (Log de transacciones) se dirija a un conjunto específico de archivos y dónde se deben ubicar exactamente dichos archivos. Si no proporcionamos esta opción, SQL Server creará el registro (Log de transacciones) en un solo archivo y lo predeterminará para que tenga un tamaño igual al 25% del tamaño del archivo de datos.

COLLATE:

Esta opción tiene que ver con el problema de la ordenación, las mayúsculas y minúsculas y los acentos. Al instalar su SQL Server, habrá decidido sobre cuál es la intercalación predeterminada, pero puede sobrescribir este parámetro a nivel de base de datos y a nivel de columna.

Recomendaciones

Es muy recomendable que guarde sus archivos de registro (.ldf) en una unidad de disco diferente a la de sus archivos de datos principales (.mdf). Al hacerlo, evitará que los archivos de datos principal y de registro compitan por la E/S del disco además de proporcionar una seguridad adicional si falla una unidad.

CREATE TABLE

Crea una nueva tabla en SQL Server

Sintaxis:

```
CREATE TABLE nombre_tabla
(
nombre_campo_1 tipo_1
nombre_campo_2 tipo_2
nombre_campo_n tipo_n
Key(campo_x,...)
)
```

RESTRICCIONES EN UNA TABLA

Limitar el tipo de dato que puede ingresarse en una tabla. Dichas restricciones pueden especificarse cuando la tabla se crea por primera vez a través de la instrucción **CREATE TABLE**, o luego de crear la tabla a través de la instrucción **ALTER TABLE**.

Los tipos comunes de restricciones incluyen las siguientes:

- NOT NULL
- UNIQUE
- CHECK
- Clave primaria
- Clave externa

NOT NULL

De forma predeterminada, una columna puede ser NULL. Si no desea permitir un valor NULL en una columna, deberá colocar una restricción NOT NULL en esta columna especificando que en esa columna es obligatorio introducir un dato.

Por ejemplo, en la siguiente instrucción,

```
CREATE TABLE Cliente
(Codigo integer NOT NULL,
Apellido varchar (30) NOT NULL,
Nombre varchar(30));
```

Con la instrucción NOT NULL en las columnas “Codigo” y “Apellido”, estas no aceptan valores nulos (vacíos), mientras que el campo “Nombre” si puede contener valores nulos.

UNIQUE

La restricción UNIQUE asegura que todos los valores en una columna sean únicos.

Por ejemplo, en la siguiente instrucción,

```
CREATE TABLE Cliente
(Codigo integer UNIQUE,
Apellido varchar (30),
Nombre varchar(30));
```

La columna “Codigo” no puede incluir valores duplicados, dicha restricción no se aplica para columnas “Apellido” y “Nombre”.

Nota: esta restricción no sustituye la clave primaria

CHECK

La restricción CHECK asegura que todos los valores en una columna cumplan ciertas condiciones.

Por ejemplo, en la siguiente instrucción:

```
CREATE TABLE Cliente
(Codigo integer CHECK (Codigo > 0),
Apellido varchar (30),
Nombre varchar(30));
```

La columna “Codigo” sólo acepta datos enteros mayores que cero.

ALTER TABLE

Modifica una definición de tabla al alterar, agregar o quitar columnas y restricciones, reasignar particiones, o deshabilitar o habilitar restricciones y desencadenadores.

Algunas instrucciones pueden ser:

Agregar un nuevo campo

Para agregar un nuevo campo a la tabla digitamos la siguiente sentencia:

```
ALTER TABLE CONTACTOS
ADD estado VARCHAR(8)
```

Esta sentencia nos permite agregar el campo **estado** con un tipo de dato VARCHAR de 8 caracteres a la tabla contactos

Eliminar un campo

Para eliminar un campo de la tabla digitamos la siguiente sentencia:

```
ALTER TABLE CONTACTOS
DROP COLUMN estado
```

AGREGAR UNA RESTRICCIÓN

Con la instrucción ALTER TABLE se pueden agregar las diferentes tipos de restricciones mencionadas anteriormente, por ejemplo:

Restricción Check

```
ALTER TABLE CONTACTOS
ADD CONSTRAINT exd_check CHECK (codigo> 1);
```

Restricción Default

```
ALTER TABLE VENTA
ADD CONSTRAINT default_fecha
DEFAULT getdate() FOR fechaventa;
```

Nota: getdate= fecha actual del sistema

Llave Primaria (PRIMARY KEY)

```
ALTER TABLE CONTACTOS
ADD CONSTRAINT primary_codigo PRIMARY KEY CLUSTERED (Codigo);
```

Llave Foránea (FOREIGN KEY)

```
ALTER TABLE VENTA
ADD CONSTRAINT fK_venta FOREIGN KEY (IdVenta)
    REFERENCES CONTACTOS (Codigo)
```

III. Requerimientos

- Máquina con SQL Server 2012
- Guía Número 5 de base de datos

IV. Procedimiento

Parte 1: Iniciando sesión desde SQL Server Managment Studio

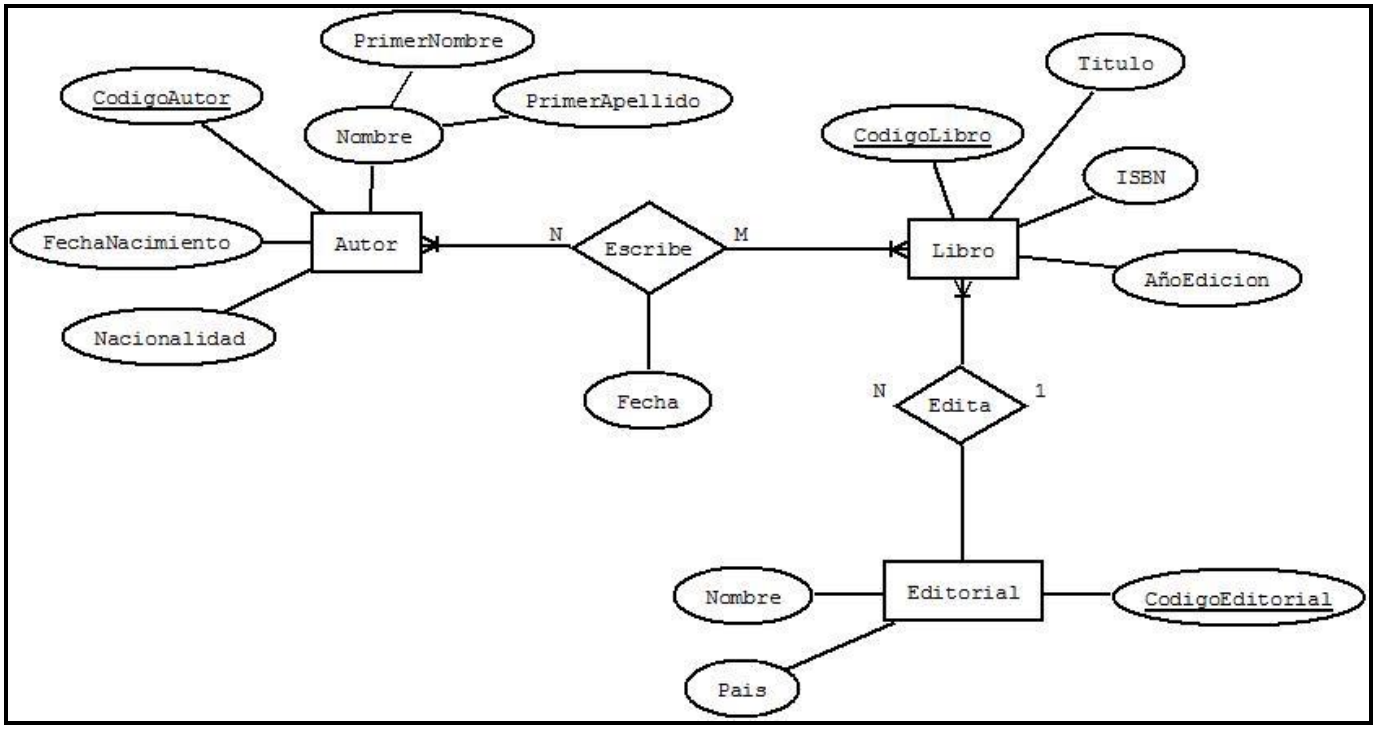
1. Hacer clic en el botón **Inicio**
2. Hacer clic en la opción **Todos los programas** y hacer clic en **Microsoft SQL Server 2012**

Para conectarse con el servidor de base de datos elija los siguientes parámetros de autenticación:

- **Tipo de servidor:** Database Engine
- **Nombre del servidor:** Colocar el nombre del servidor local, por ejemplo PC**NumMaquina-SALA2**
  Nota: NumMaquina es el número de la maquina local
- **Autenticación**: SQL Server Authentication
- **Login:** sa
- **Password:** 123456

Parte 2. Diseño de una base de datos relacional con Transact SQL

Para esta parte de la guía se tomará el siguiente modelo E-R:

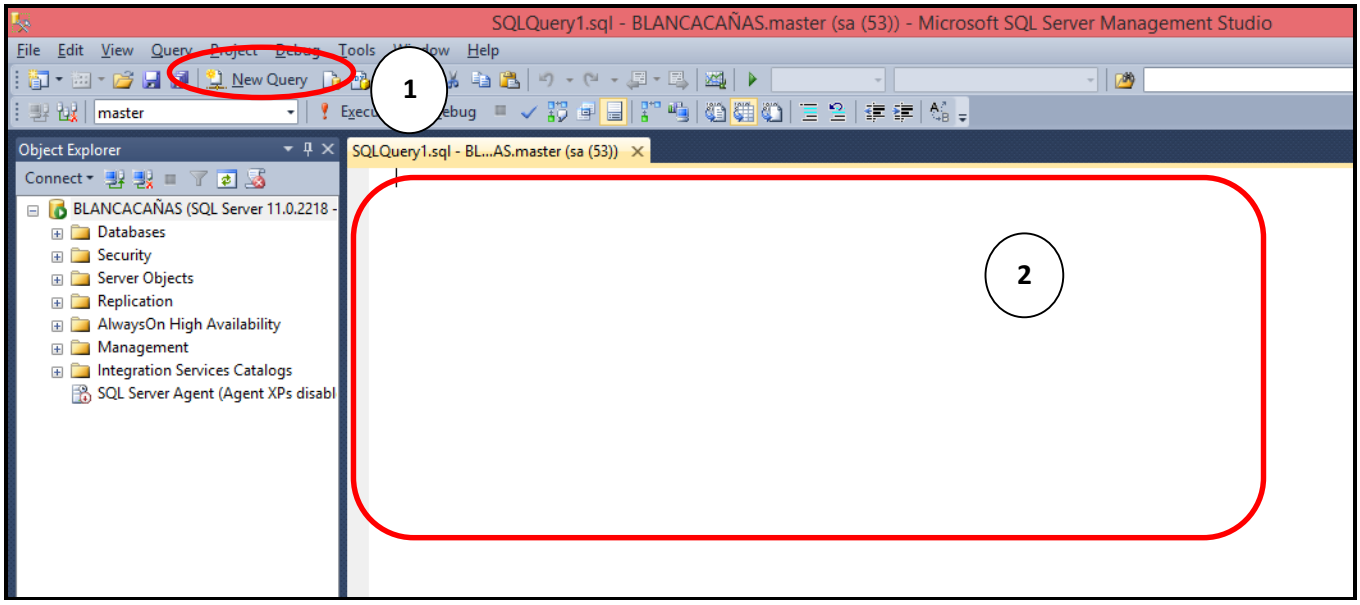


Interpretación del modelo E-R:

- Un Autor puede escribir muchos libros y un libro puede ser escrito por muchos autores
- Una Editorial puede editar muchos libros pero un libro solo puede ser editado por una editorial especifica

Ejercicio 1. Crear la base de datos

1. Luego de estar dentro de Management Studio, ubique el botón en la opción **New Query** (Nueva consulta) (1), hacer clic sobre él para que se habilite un espacio en blanco(2) en donde se pueden ingresar o digitar las sentencias SQL



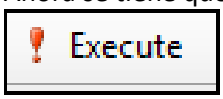
2. Crear una carpeta en la **UNIDAD C** con el nombre **Guia5_SuCarnet,** para que pueda verificar la creación de los archivos .mdf y .ldf de la base de datos del procedimiento de la guía.

3. Crear la base de datos **BasedeDatos_SuCarnet**, en la área de edición de **consultas SQL**
Recuerde que debe sustituir la palabra **SuCarnet** con su número de carnet

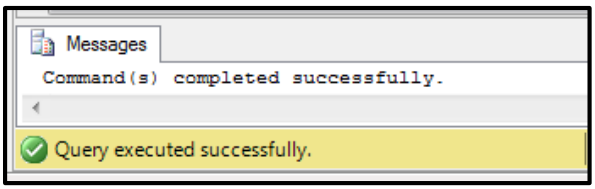
4. Digitar la siguiente consulta:

```
--Creando la Base de datos
--Colocando un comentario
USE master;--Hacer uso de la base de datos MASTER
GO --Comando que indica el final de un lote de instrucciones Transact-SQL.

CREATE DATABASE BasedeDatos_SuCarnet
ON
( NAME = BasedeDatos_SuCarnet_dat,
    FILENAME = 'C:\Guia5_CA002324\BasedeDatos_SuCarnetdat.mdf',
    SIZE = 5,
    MAXSIZE = 20,
    FILEGROWTH = 5 )
LOG ON
( NAME = BasedeDatos_SuCarnet_log,
    FILENAME = 'C:\Guia5_CA002324\BasedeDatos_SuCarnetlog.ldf',
    SIZE = 2,
    MAXSIZE = 10,
    FILEGROWTH = 2 ) ;
GO
```

5. Ahora se tiene que **ejecutar la consulta**, hacer clic en la opción Execute (Ejecutar)

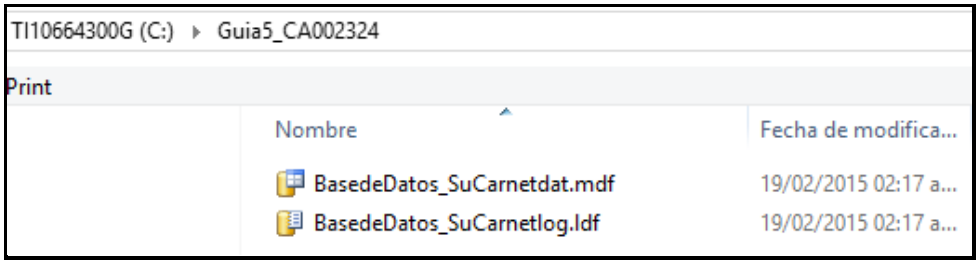


6. Si aparece el siguiente mensaje:

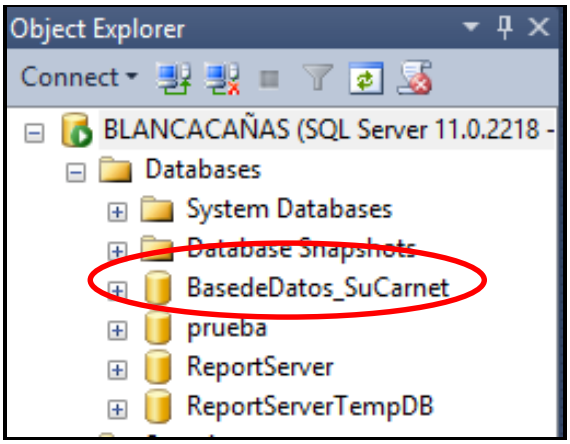


Quiere decir que la consulta se ejecutó correctamente.

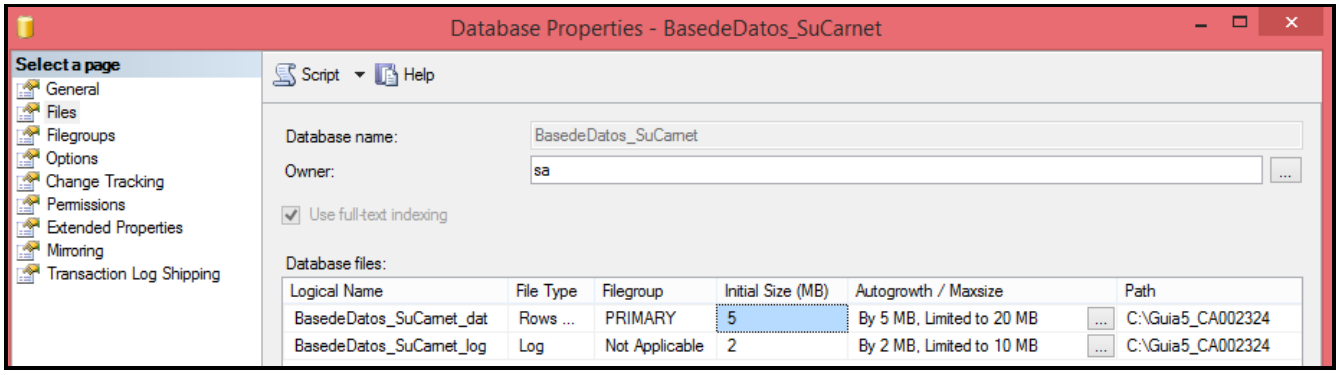
7. Revise la carpeta en la Unidad C y deberá tener los archivos .mdf y .ldf



8. Para verificar si se creó la base de datos, actualice la carpeta DataBase (Base de datos) y observará que se creó la nueva base de datos



9. Verifique las propiedades de la base de datos si son las que se colocaron cuando se creó la base de datos.



Ejercicio 2. Crear las tablas de la base de datos

Las tablas a crear en este ejercicio son:

Tablas	Campos
Autor	CodigoAutor Nombre (PrimerNombre y Primer Apellido) FechaNacimiento Nacionalidad
Libro	CodigoLibro Titulo ISBN AñoEdicion CodigoEditorial
Editorial	CodigoEditorial Nombre País
Detalle_AutorLibro	CodigoAutor CodigoLibro Fecha

Nota:

- A la tabla Libro se agregó el campo CodigoEditorial por la relación que existe entre esta tabla y la tabla Editorial

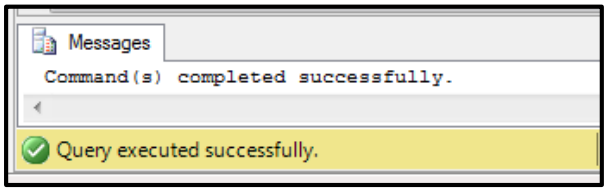
- Se creó la tabla Detalle_AutorLibro ya que esta tabla intermedia rompe la relación de muchos a muchos que existe entre Autor y Libro, se le agregaron los campos CodigoAutor y CodigoLibro para crear la relación de uno a muchos.

1. Después de la consulta que utilizó para crear la base de datos, digite la siguiente consulta SQL:

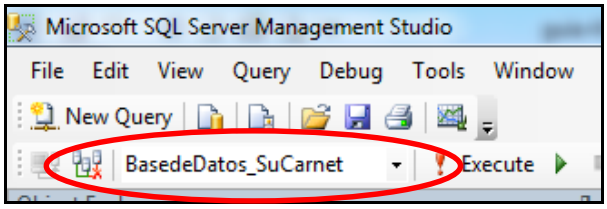
```
--Seleccionando la Base de datos del ejecicio
Use BasedeDatos_SuCarnet
GO
```

2. Seleccione desde el comando USE hasta el comando GO y haga clic en la opción Ejecutar (Execute)

3. Si aparece el siguiente mensaje, se creó correctamente la tabla



Con la instrucción **USE**, hemos seleccionado la base de datos en la cual deseamos trabajar



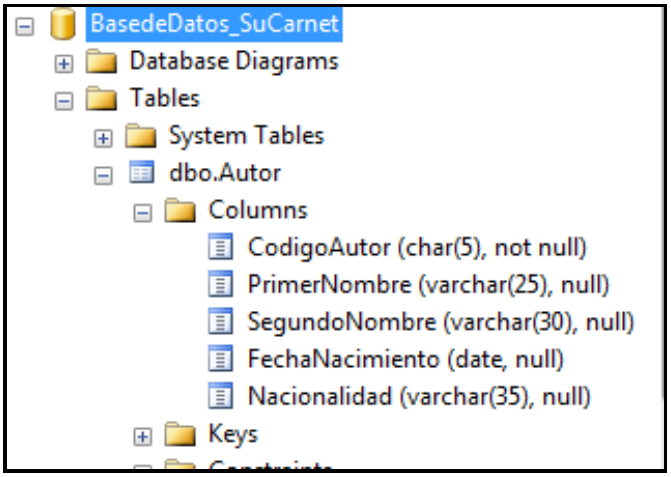
Nota: La instrucción **USE** se usará las veces que nos queremos cambiar de base de datos, sino solo se utiliza una vez

4. Crear la **tabla Autor**, digitar después de la última instrucción GO, la siguiente consulta:

```
--Creando las tablas
--Tabla Autor
CREATE TABLE Autor
(CodigoAutor char(5)NOT NULL,
 PrimerNombre varchar(25),
 SegundoNombre varchar(30),
 FechaNacimiento date,
 Nacionalidad varchar(35),
)
GO
```

5. Seleccionar desde el comando CREATE hasta la instrucción GO, hacer clic en la opción Execute

6. Actualice (Refresh) su Base de datos y abra la carpeta Tables y deberá observar la tabla **Autor** creada



7. Ahora se tiene que crear la tabla **Libro**, digite la siguiente consulta después de la última instrucción GO:

```
--Tabla Libro
CREATE TABLE Libro
(CodigoLibro char(10)NOT NULL,
 Titulo varchar(max),
 ISBN varchar(20) NOT NULL,
 AñoEdicion char(4),
 CodigoEditorial char(5),
)
GO
```

8. Seleccione la consulta desde la instrucción CREATE hasta GO y ejecútela, actualice la base de datos y deberá observar la tabla agregada.

9. Crear la tabla Editorial, digite la siguiente consulta después de la última instrucción GO:

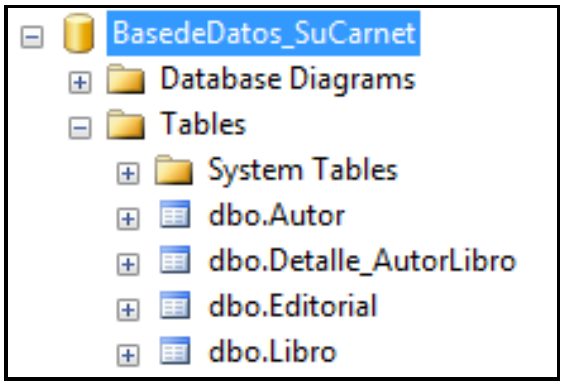
```
--Tabla Editorial
CREATE TABLE Editorial
(CodigoEditorial char(5)NOT NULL,
 Nombre varchar(45),
 Pais varchar(50),
)
GO
```

10. Ejecutar la consulta y actualice la base de datos

11. Crear la tabla **Detalle_AutorLibro**, después de la consulta anterior, digitar la siguiente:

```
--Tabla Detalle_AutorLibro
CREATE TABLE Detalle_AutorLibro
(CodigoAutor char(5),
 CodigoLibro char(10),
 Fecha date,
)
GO
```

12. Ejecutar la consulta y actualice la carpeta Tables, al final deberá tener en la base de datos BasedeDatos_SuCarnet, las cuatro tablas creadas:



Ejercicio 3. Creando restricciones con la instrucción ALTER TABLE

Ahora se creará las relaciones entre las tablas tomando siempre el modelo E-R

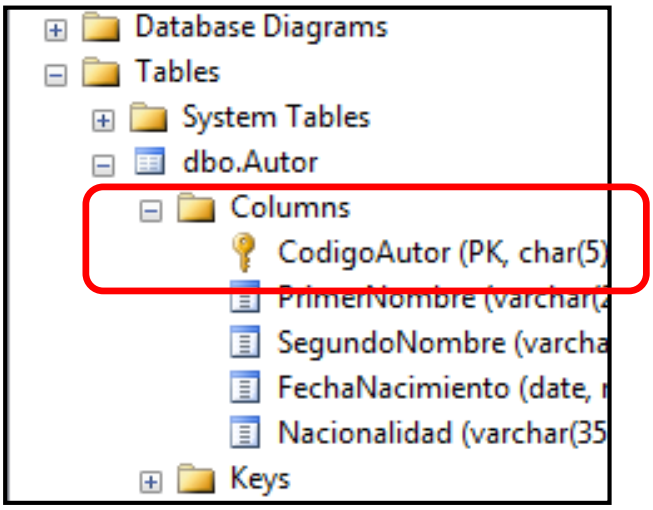
Estableciendo las claves principales o primarias:

1. Después de la última consulta SQL de la parte de creación de tablas, digite la siguiente consulta:

```
--Creando las restricciones

--Creando llaves primarias
--Tabla Autor

ALTER TABLE Autor
ADD CONSTRAINT PK_CodigoAutor
PRIMARY KEY (CodigoAutor)
GO
```

2. Seleccione la consulta desde la instrucción ALTER hasta el comando GO y ejecútela, si no hay ningún error actualice su tabla

3. Expanda la carpeta dbo.Autor y luego la carpeta Columns, observará los campos de la tabla y la asignación de la clave principal en el campo CodigoAutor



4. Asigne la clave principal a las siguientes tablas:

Tabla	Campo
Libro	CodigoLibro
Editorial	CodigoEditorial

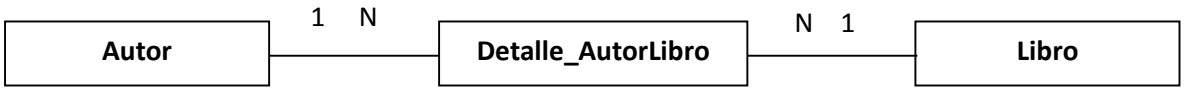
5. Digite las siguientes consultas:

```
--Tabla Libro
ALTER TABLE Libro
ADD CONSTRAINT PK_CodigoLibro
PRIMARY KEY (CodigoLibro)
GO


--Tabla Editorial
ALTER TABLE Editorial
ADD CONSTRAINT PK_CodigoEditorial
PRIMARY KEY (CodigoEditorial)
GO
```

6. Ejecute cada consulta y verifique que se han creado las claves principales en las tablas

Estableciendo las claves externas o foráneas

Siempre tomando en cuenta el modelo E-R, se observa que existe una relación de Muchos a Muchos entre las entidades Autor y Libro, pero como ya se definió la tabla intermedia Detalle_AutorLibro ahora la relación queda de la siguiente manera:



Y se lee de la siguiente manera:
Un autor puede escribir muchos libros y un libro puede ser escrito por muchos autores, por lo tanto en la tabla Detalle_AutorLibro se almacena por separado el código del libro las veces que se quiera como también el código del autor las veces que sea necesario.

Relaciones a crear:

Claves primarias	**Claves foráneas**
Tabla: Autor Campo: CodigoAutor Tabla: Libro Campo: CodigoLibro	Tabla: Detalle_AutorLibro Campo: CodigoAutor Hace referencia a la clave primaria CodidoAutor de la tabla Autor Tabla: Detalle_AutorLibro Campo: CodigoLibro Hace referencia a la clave primaria CodidoLibro de la tabla Libro

7. En el editor de consultas digite la consulta la cual va a crear la relación entre la tabla **Autor** y **Detalle_AutorLibro**

```
--Creacion de llaves foraneas
--Relacion entre Autor y Detalle_AutorLibro

ALTER TABLE Detalle_AutorLibro
ADD CONSTRAINT FK_CodigoAutor
FOREIGN KEY (CodigoAutor)
REFERENCES Autor(CodigoAutor)
GO
```

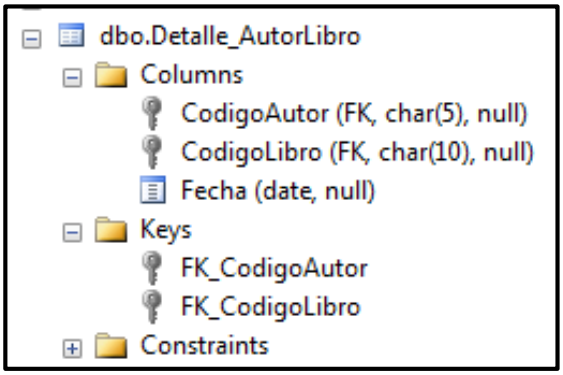
8. Seleccione la consulta y ejecútele

9. Digite la siguiente consulta, la cual creará la relación de Libro y Detalle_AutorLibro

```
--Relacion entre Libro y Detalle_AutorLibro

ALTER TABLE Detalle_AutorLibro
ADD CONSTRAINT FK_CodigoLibro
FOREIGN KEY (CodigoLibro)
REFERENCES Libro(CodigoLibro)
GO
```

10. Seleccione la consulta y ejecútele

11. Ahora despliegue las carpetas **Columns** y **Keys** de la tabla **Detalle_AutorLibro** y observará la creación de las llaves foráneas en la tabla y las cuales están se han utilizado para crear la relación entre la tabla:



12. Crear la relación entre las **tablas Editorial y Libros**

13. Digite después de la última consulta:

```
--Relacion entre Editorial y Libro

ALTER TABLE Libro
ADD CONSTRAINT FK_LibroEditorial
FOREIGN KEY (CodigoEditorial)
REFERENCES Editorial(CodigoEditorial)
GO
```

14. Actualice la base de datos y observará los cambios en las tablas Libro y Editorial

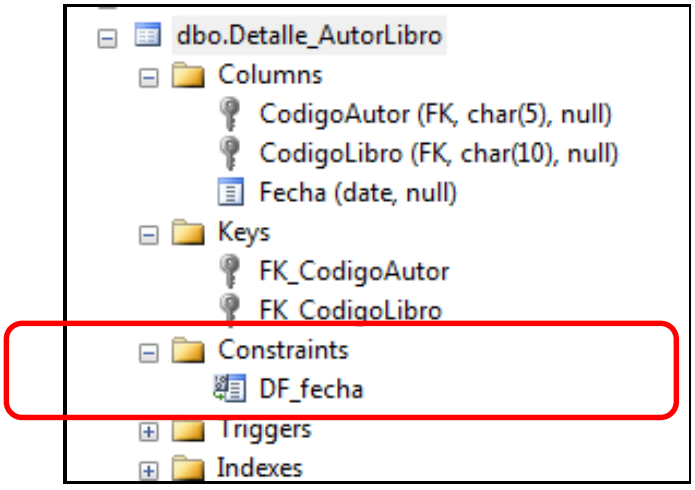
Estableciendo restricciones: Default, Check y Unique

Restricción Default

1. Se creará una restricción Default en el campo Fecha para la tabla Detalle_AutorLibro, en la cual si el usuario no digita nada para esta fecha que se introduzca la fecha del sistema.

```
--Restriccion Default

ALTER TABLE Detalle_AutorLibro
ADD CONSTRAINT DF_fecha
DEFAULT getdate() FOR Fecha;
GO
```

2. Actualice su tabla Detalle_AutorLibro y verifique en la carpeta Constraints la creación de la restricción Default



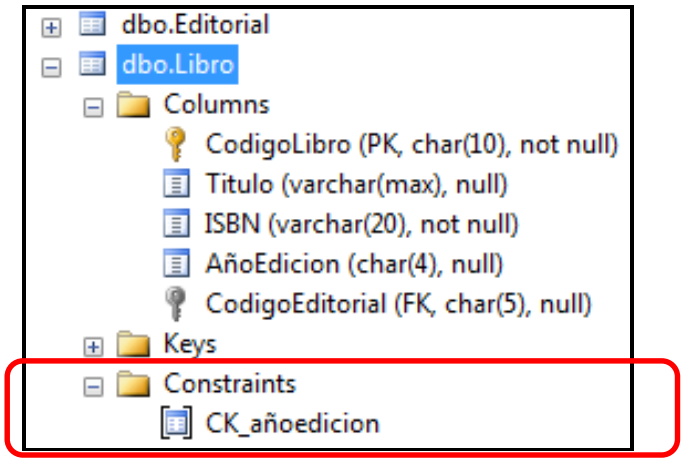
Restricción Check

3. Digite la siguiente consulta:

```
--Restriccion Check

ALTER TABLE Libro
ADD CONSTRAINT CK_añoedicion
CHECK (AñoEdicion> 2010);
GO
```

4. La cual agrega una restricción Check , para el campo AñoEdicion de la tabla Libro, los datos que se introduzcan para este campo deberán ser mayores del 2010

5. Ejecute la consulta y verifique los cambios que le hizo a la columna



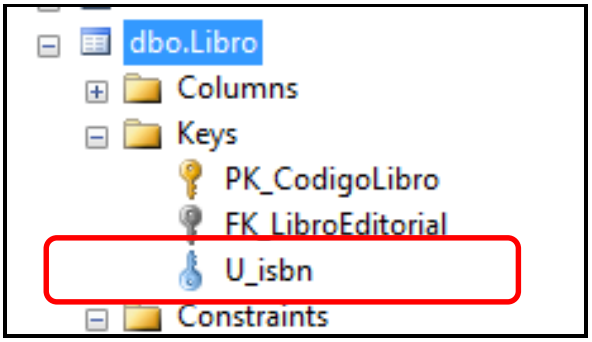
Restricción Unique

6. La consulta que se digitará a continuación, crea una restricción Unique para el campo ISBN de la tabla Libro, el cual se puede tomar ese campo como dato único, pero no es una clave principal ya que ese campo no se utiliza para crear relaciones entre tablas

7. Digite la siguiente consulta:

```
--Restriccion Unique

ALTER TABLE Libro
ADD CONSTRAINT U_isbn
UNIQUE(ISBN)
GO
```

8. Actualice la tabla Libro y vea que agrego la restricción Unique

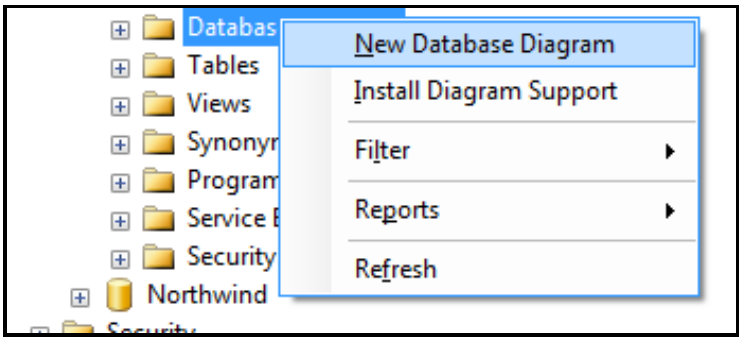


9. Hasta aquí ha creado las relaciones entre las tablas y las restricciones asegurando la calidad de los datos

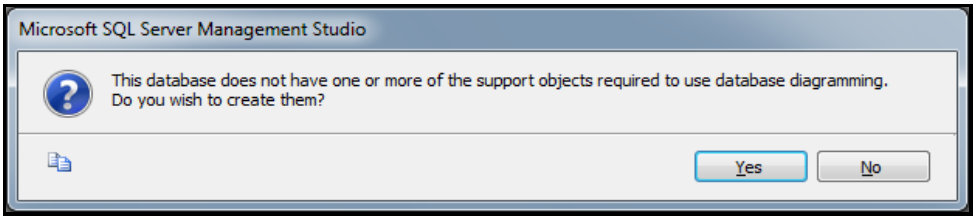
Ejercicio 4. Creando el diagrama de base de datos

1. Haga clic derecho sobre la opción Diagrama de Base de datos (Database Diagrams)

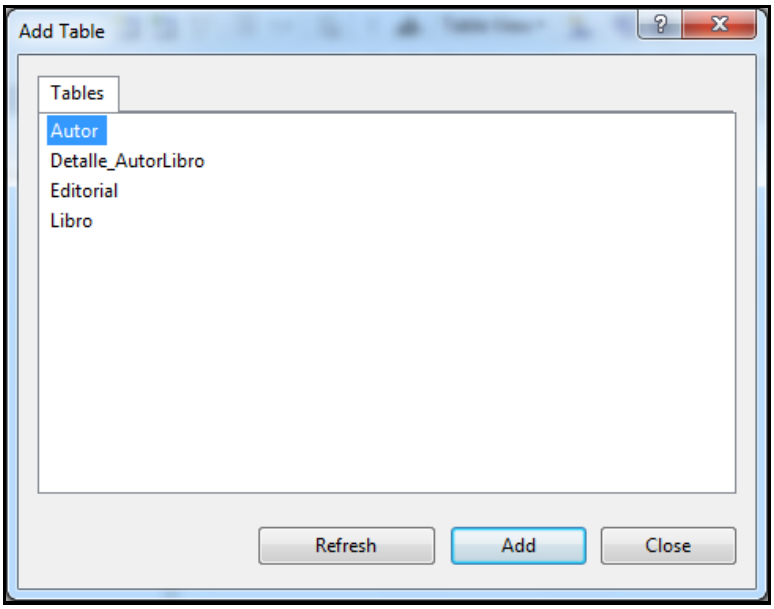
2. Seleccione la opción **Nuevo Diagrama de Base de datos (New Database Diagram)**



3. En la ventana emergente haga clic en Si (Yes)



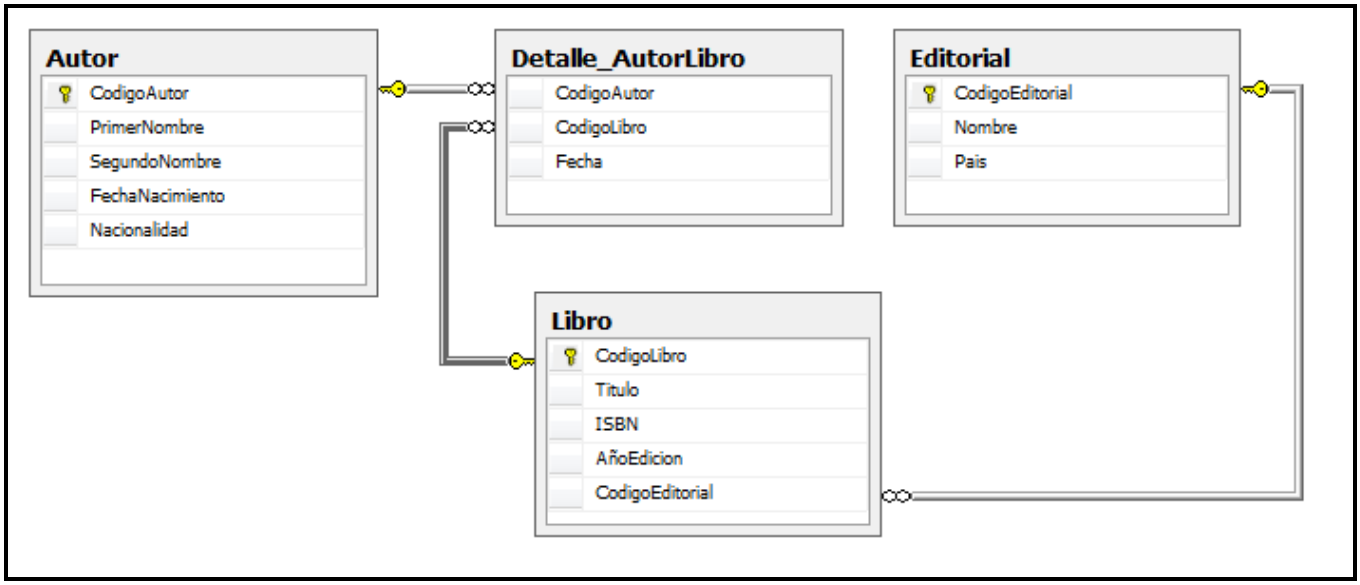
4. Se habilita la venta Agregar tabla (Add Table)



5. Seleccione cada tabla y haga clic en Agregar (Add), realice este paso hasta que termine de agregar todas las tablas

6. Haga clic en Cerrar (Close)

7. Y observará que se ha creado el siguiente diseño de base de datos



8. Guarde el diseño de la base de datos

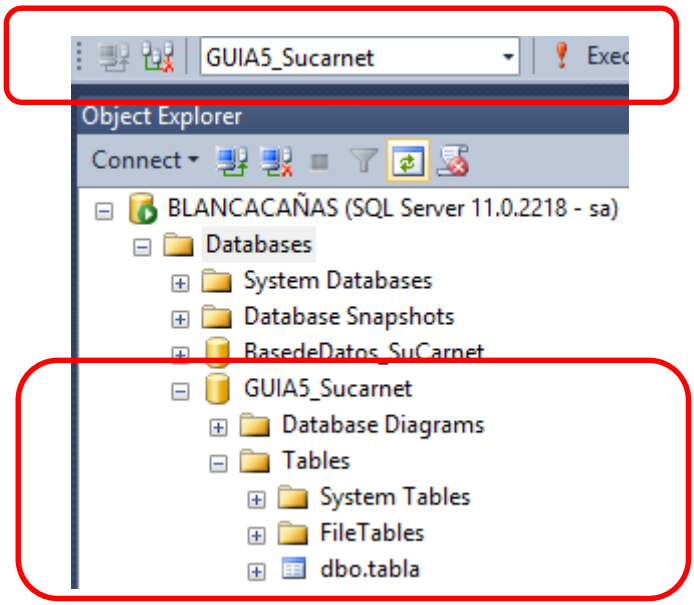
Ejercicio 5. Usando los comandos DROP DATABASE y DROP TABLE

El comando DROP se utiliza para quitar objetos existentes como base de datos, tablas, usuarios, vistas, triggers, procedimiento almacenado etc.

1. Después del último comando GO, digitar la siguiente consulta:

```
--Uso del Comando DROP

CREATE DATABASE GUIA5_Sucarnet

GO

USE GUIA5_Sucarnet

GO

CREATE TABLE tabla
(campo1 int,
 campo2 varchar(5)
)
GO
```

2. Verifique que creo la base de datos y la tabla dentro de la base de datos **Guia5_Sucarnet**, como también que ya está haciendo uso de ella



3. Ahora vamos a eliminar la tabla de la base de datos

4. Digitar la siguiente consulta

```
--Eliminando la tabla

DROP TABLE tabla
GO
```

5. Actualice su base de datos y ya no debe tener la tabla

6. Vamos a eliminar ahora la base de datos

7. Digite la siguiente consulta:

```
--Eliminando la base de datos

USE Master
GO

DROP DATABASE Guia5_Sucarnet
GO
```

8. Actualice la carpeta Databases y ya no debe estar la base de datos que creo en este ejercicio

Ejercicio 6. Guardar las consultas SQL

1. Ahora vamos a guardar las consultas en un archivo que tiene la extensión .sql esto para en cualquier momento haga uso de las consultas

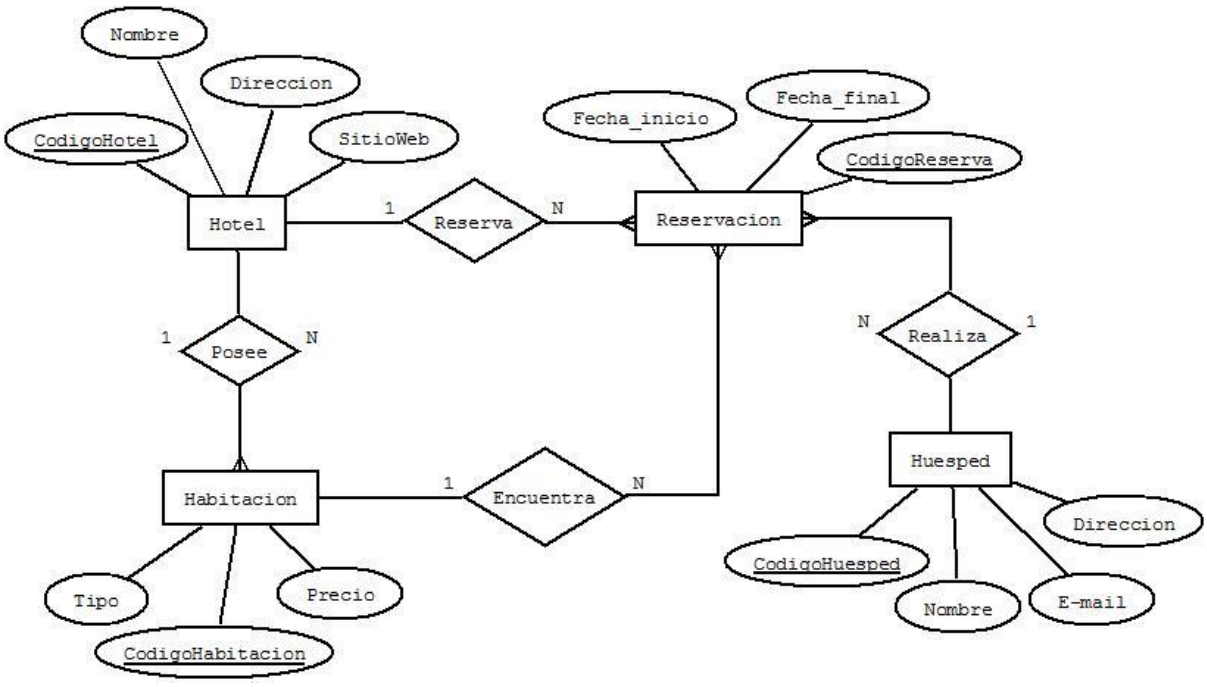
2. Hacer clic en el menú **File** y seleccione la opción **Save SQLQuery1.sql As…** busque la carpeta que creo al inicio del ejercicio 1

3. Y guarde el archivo como Guia4_SuCarnet.sql

4. Verifique que creo el archivo .sql, ese es el archivo que le enviará al docente.

V. Ejercicio complementario

En una nueva área de edición de consultas crear la siguiente base de datos en SQL Server:



Nombre de la base de datos: **Hotel_SuCarnet**

Haciendo uso de TRANSACT – SQL crear:

1. La base de datos
2. Las tablas con sus correspondientes campos y propiedades
3. Las relaciones entre las tablas
4. Crear las siguientes restricciones en los campos:
    a. Unique:
        i. Tabla Hotel (Nombre, SitioWeb)
        ii. Tabla Huesped (E-mail)
    b. Check:
        i. Tabla Habitacion (Precio mayor que 25, Tipo: Doble, individual)
        ii. Tabla Reservacion (Fecha final tiene que ser mayor a la fecha de inicio)
    c. Default:
        i. Tabla Reservacion (Fecha inicio por defecto puede ser la fecha actual del sistema)

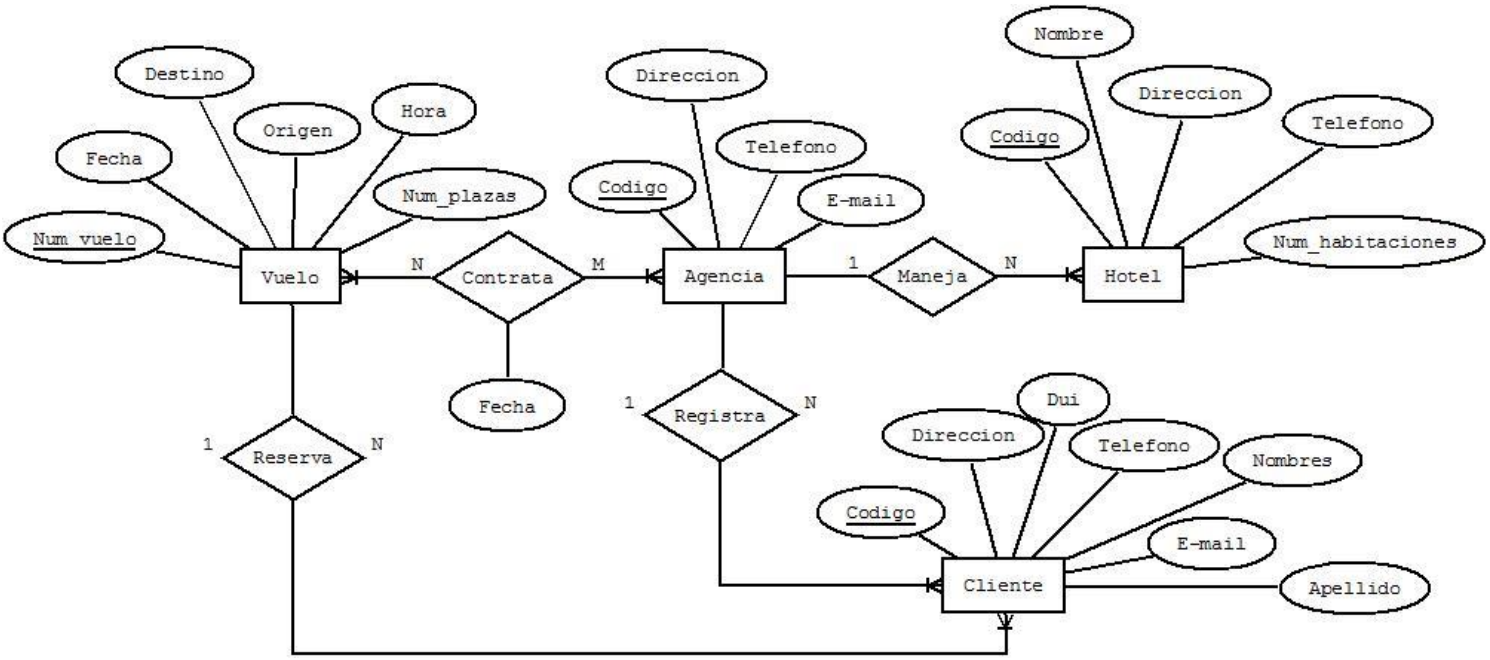
5. Crear el diagrama de la base de datos

Guarde el archivo de las consultas con el nombre **Consultas_Hotel_SuCarnet.sql,** y enviarlo al docente

VI. Análisis de resultados

Tarea en parejas:

Crear la siguiente base de datos en SQL Server:



Nombre de la base de datos: **Agencia_Carnet1_Carnet2**

Haciendo uso de TRANSACT – SQL crear:

1. La base de datos
2. Las tablas con sus correspondientes campos y propiedades
3. Las relaciones entre las tablas
4. Restricciones en todos los campos que crea conveniente
5. Crear el diagrama de la base de datos

Guarde el archivo de las consultas con el nombre **Consultas_Agencia.sql,** en este archivo al inicio de las consultas digitar en comentario los nombres y carnets de cada pareja.

VI. Fuente de consulta

1. La Biblia de SQL Server 2005
Madrid, España: Anaya, 2006
Autor: Mike Gundelerloy y Joseph L. Jorden
Biblioteca UDB – Clasificación: 005.361 G975 2006

2. Microsoft SQL Server 2008: Guía del Administrador
Madrid, España: ANAYA, 2009
Autor: William Stanek
Biblioteca UDB – Clasificación: 005.361 S784 2009

3. Libros en pantalla de SQL Server 2012

https://msdn.microsoft.com/es-es/library/ms176061(v=sql.110).aspx

https://msdn.microsoft.com/es-es/library/bb510741(v=sql.110).aspx