

Universidad Autónoma de Nuevo León
"Escuela Industrial y Preparatoria Técnica "Álvaro Obregón"

Etapas 4

Reporte: Base de datos

Nombres:

MIRANDA MATINEZ VELAZQUEZ-2092832

Grupo: 5L2/514

Especialidad: Programación Web

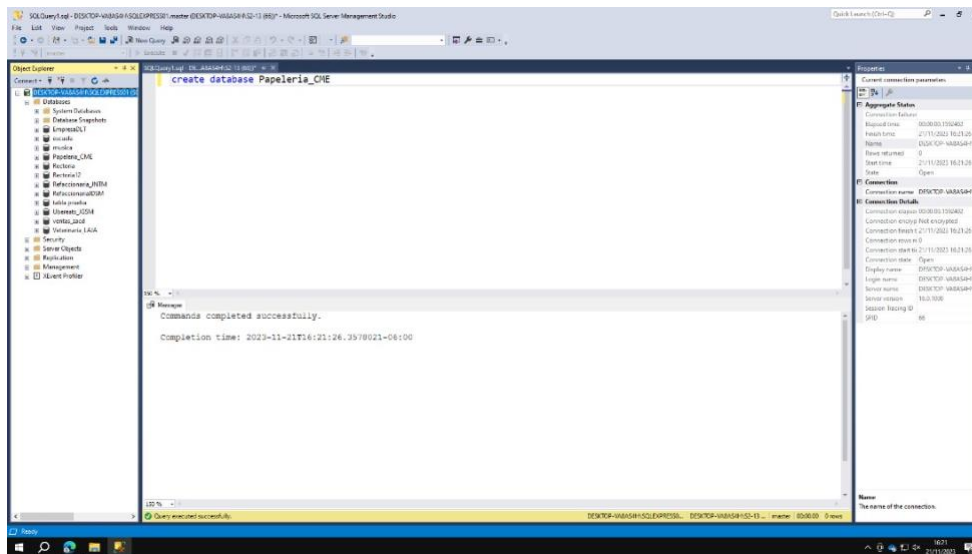
Turno: Vespertino

Docente: Mayra Edith Martínez Arellano

Crea una base de datos en SQL sobre una papelería, con siete tablas: Clientes, Proveedores, Productos, Pedidos, Ventas, Inventario y Entrada_Producto. En cada una de las tablas añadirás 10 registros. Deberás aplicar un store procedure, triggers, funciones y consultas dentro de la base de datos, así como establecer llaves primarias y foráneas para poder crear tablas relacionales. Agrega el diagrama.

Create database

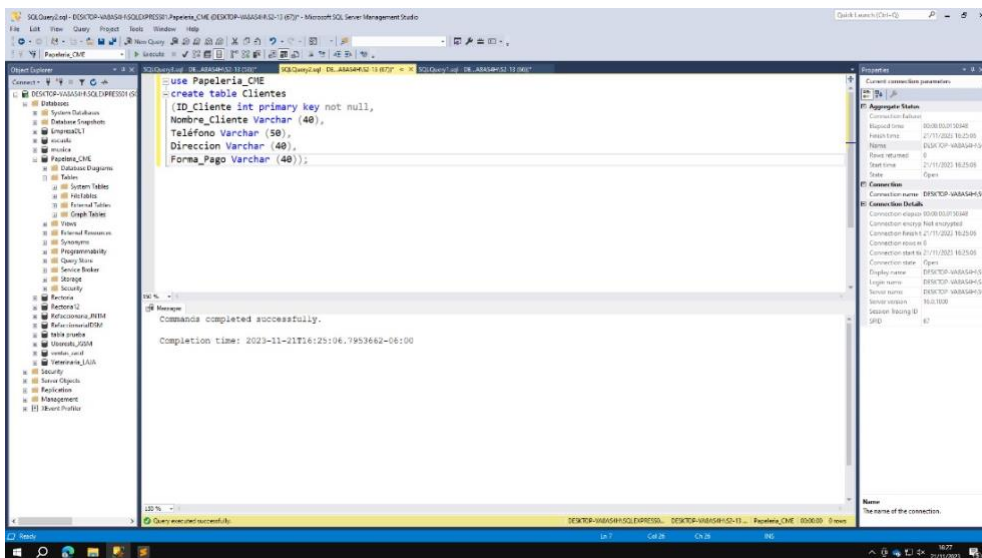
Abrimos SQL y creamos la base de datos, la cual llamaremos “Papeleria_CME”, usamos la opción “create database ____”, la ejecutamos desde la parte de arriba y una vez hecho eso nos aseguramos que se haya cargado correctamente, refrescamos varias veces la base y verificamos si se encuentra en el apartado de la izquierda.



TABLAS:

Tabla: Clientes

Luego de haber creado la base de datos, abrimos un nuevo Query en el cual crearemos la tabla, entonces, mandaremos a llamar la base de datos que creamos anteriormente con “Use____”, y en el siguiente párrafo creamos la tabla con “Create table____”, esta tabla tendrá el nombre de “Clientes”.



Registros

Después de haber creado la tabla la vamos a ejecutar, y verificamos que se haya ejecutado correctamente, después deberemos revisar que al refrescar en la parte de la izquierda aparezca la tabla dentro de la base de datos. Después en un nuevo Query vamos a agregar el contenido de la tabla, los registros (10). Para ello, nuevamente seleccionamos la base de datos de la que tomaremos los datos, y en el siguiente párrafo tomamos la tabla en la que añadiremos los datos, entre paréntesis vamos a poner los campos que anteriormente pusimos en la creación de la tabla, y seguido de ello, con el dato “Value” comenzaremos a llenar los registros en orden, separando cada dato entre comas ‘,’ y paréntesis().

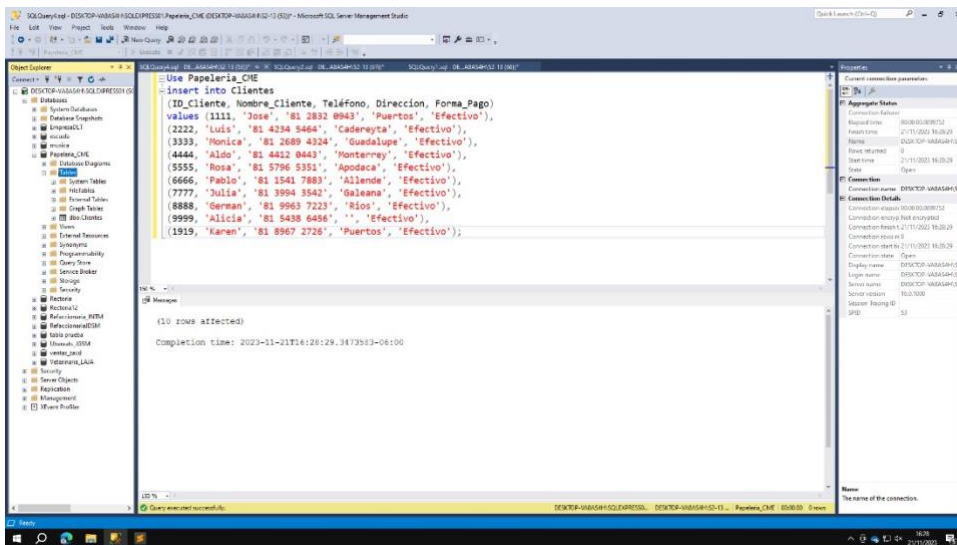


Tabla terminada

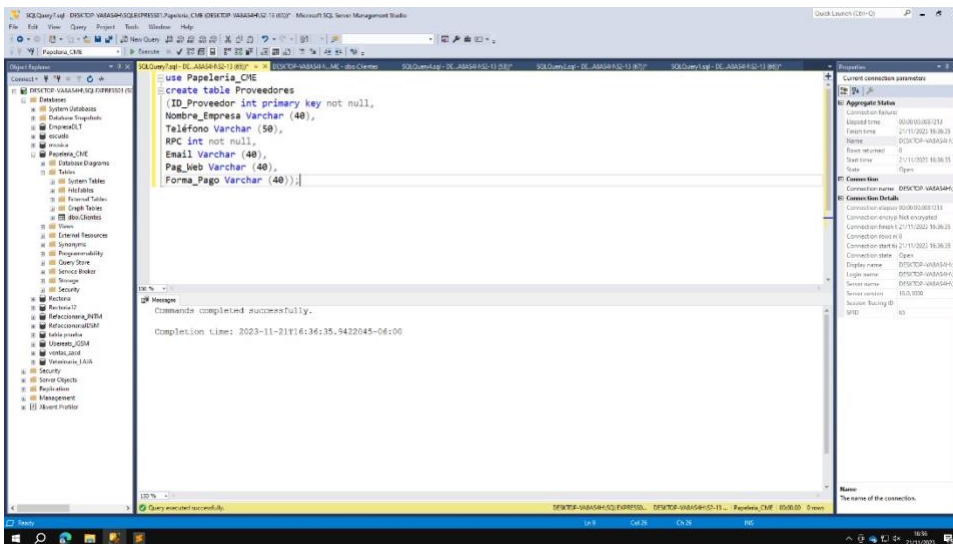
Una vez terminado ese paso vamos a ejecutar el Query, y verificamos que se haya ejecutado correctamente y sin errores, luego refrescamos en el apartado de la izquierda hasta que nos aparezcan los datos de la tabla. Una vez hecho eso, hacemos clic izquierdo en la tabla dentro de Object Explorer, y seleccionamos la opción para poder ver los primeros 200 registros, y se nos abrirá esta vista, y tenemos hecha la tabla de la base de datos.

ID_Proveedor	Nombre_Empresa	Telefono	Email	Pagina_Web
1001	Jose	01 202 2842	Papeiro	Efectivo
1002	Ruben	01 982 7770	Papeiro	Efectivo
1003	Luis	01 424 3404	Cadente	Efectivo
1004	Alfonso	01 202 4201	Cadente	Efectivo
1005	Alfonso	01 424 3404	Cadente	Efectivo
1006	Ruben	01 578 5291	Alfonso	Efectivo
1007	Jose	01 202 4201	Cadente	Efectivo
1008	Ruben	01 982 7770	Ruben	Efectivo
1009	Alfonso	01 578 5291	Alfonso	Efectivo
1010	Ruben	01 578 5291	Alfonso	Efectivo

Tabla: Proveedores

Ahora crearemos la tabla proveedores, agregamos use “Papeleria_CME” para que se agregue en la base de datos

Después agregamos el create table para poder crear la tabla y le agregamos el nombre después ponemos los campos con primary key para el id_peovedor y el varchar para los demás campos que necesitan texto y numero también agregamos el número de caracteres para poder escribir, ya hecho esto se ejecutara y verificaras si se encuentra en tu base de datos



Registros

Aquí ya empezamos a agregar el contenido de los campos agregando al principio “values” después es el id y para poder agregar los otros campos ocupas poner entre cada texto un ‘coma’.

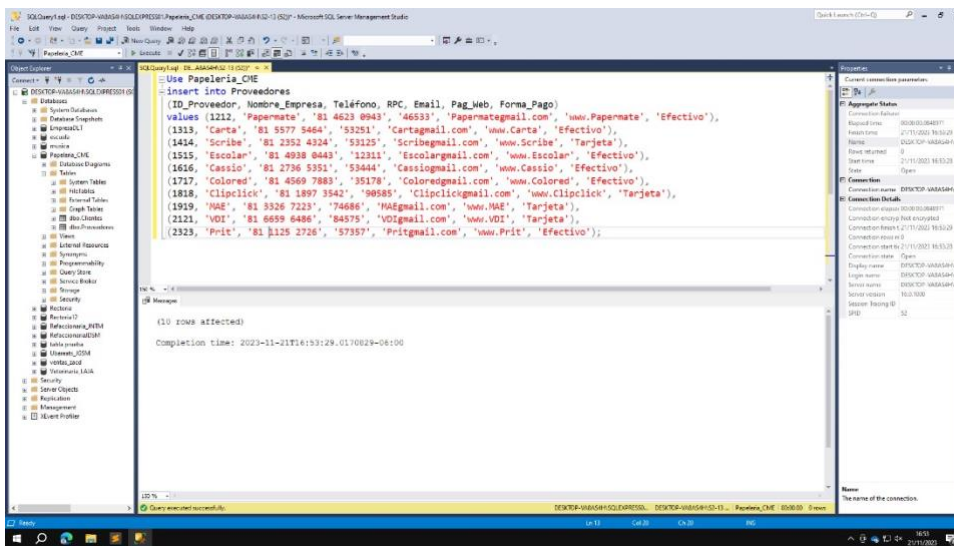


Tabla terminada

Después se ejecuta todo y se refresca para que aparezca en nuestra base, te aseguras de que funcione y de que no tenga errores.

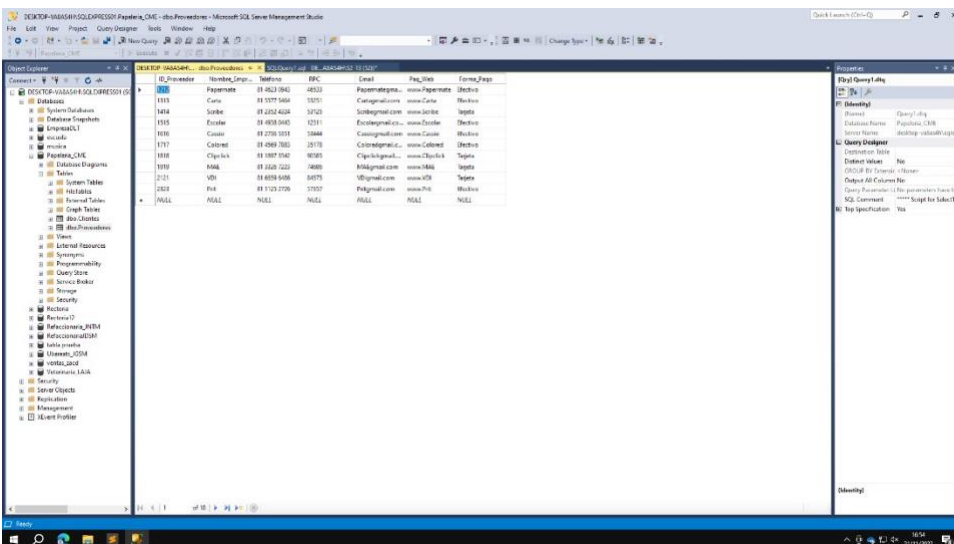
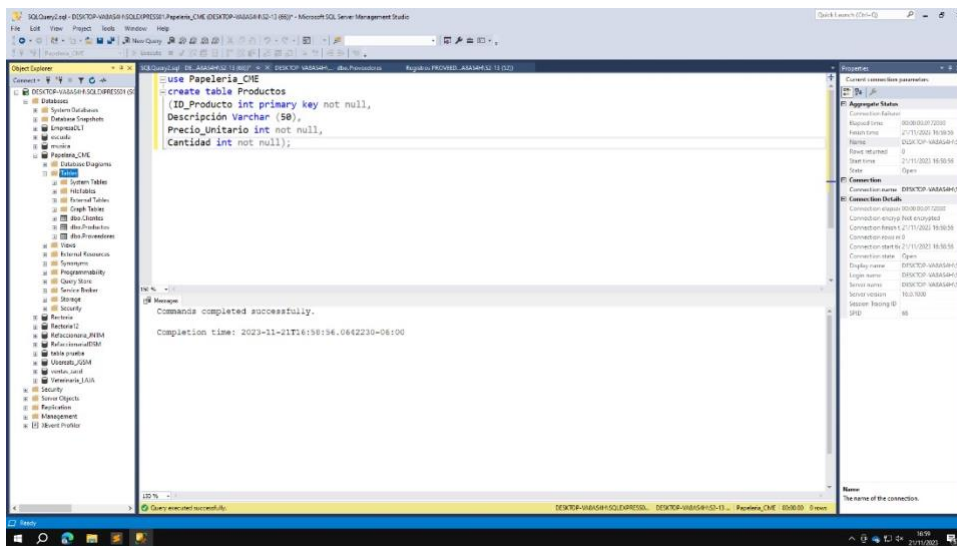


Tabla: Productos

Ahora haremos la tabla productos, para eso ocupamos agregar un nuevo query y agregaremos el “usePapeleria_CME” para poder encontrarla en la base después agregamos “create table” para crear la tabla agregando un nombre, en este caso es “productos” agregamos el id_productos y lo declaramos como int primary key despues ya todos los de más campos van declarados con varchar y un numero de caracteres para escribir, esto se ejecuta y refrescas para poder visualizar tu tabla.



Registros

Vuelves a agregar un query y agregas insert into para poder agregar a esta tabla los campos y los datos que necesites, ya que tengas toda la información en el código ejecutas el query y refrescas la página, te aseguras que aparece y verificas que no tenga errores.

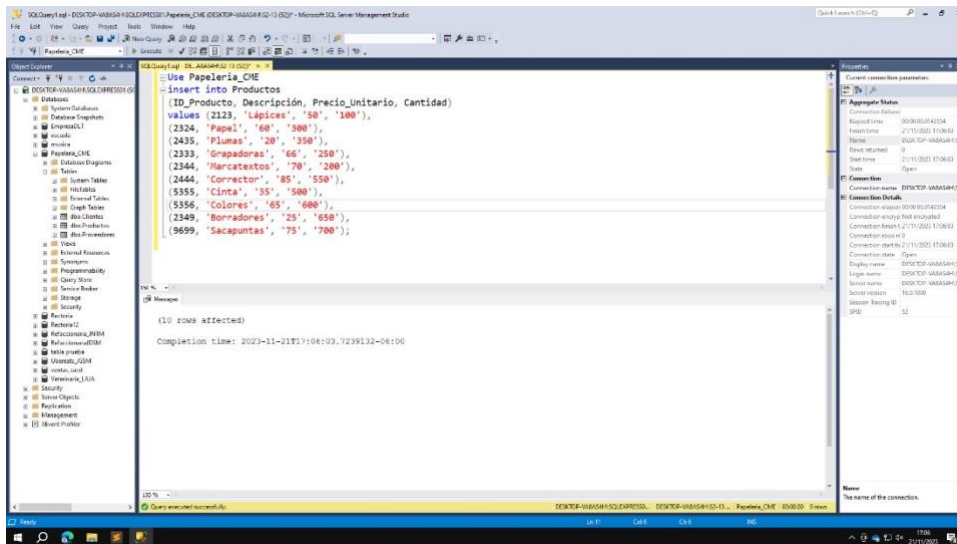


Tabla terminada

Ya que este echo y verifique todo, abres la tabla y ahora verificas que se encuentren todos los campos y todos los datos que ingresaste.

Registros

Para este es lo mismo que todos agregas los campos en un () y antes de poner los datos pones value, ejecutas el código y refrescas la página para que salga, ahora verificas que no tenga errores y todos los datos estén bien.

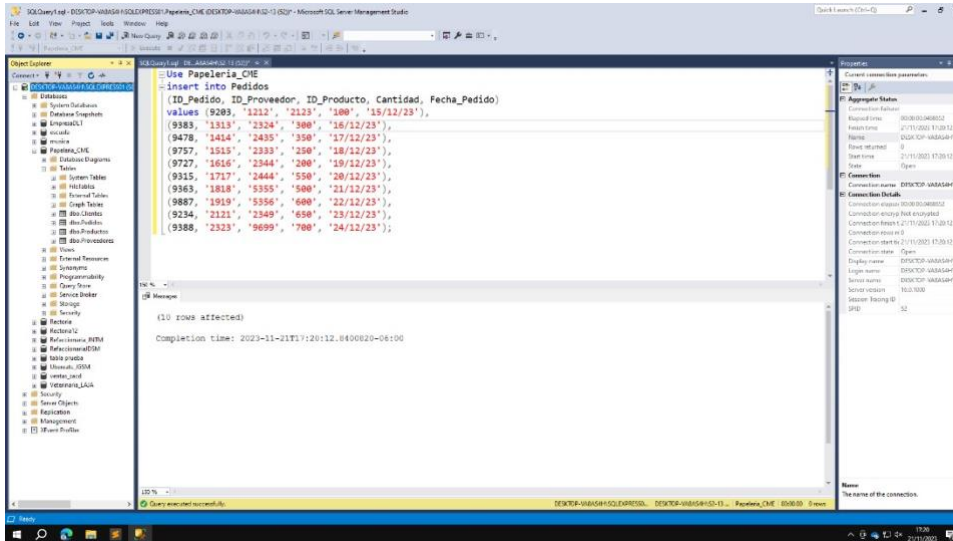


Tabla terminada

Ya que verifiques abres la tabla y te aseguras de que estén todos los datos y que no tenga errores.

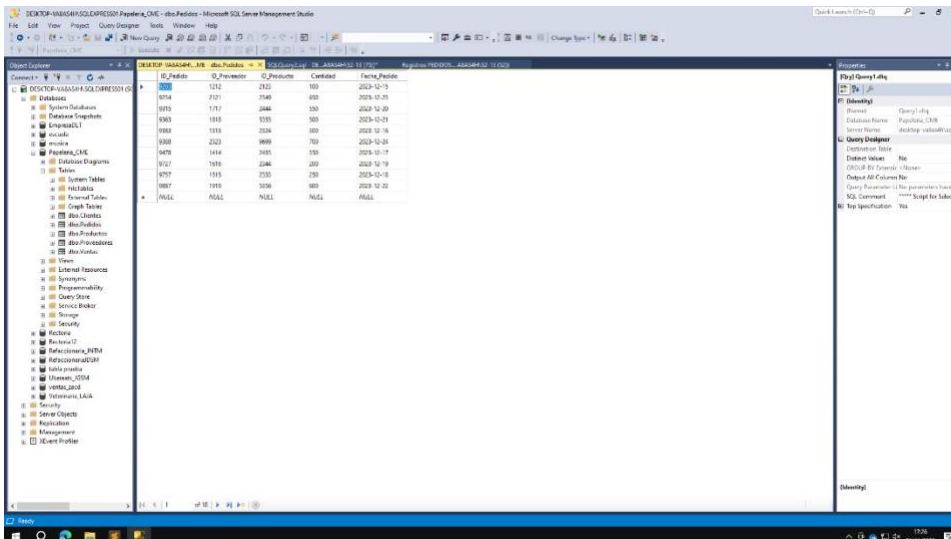
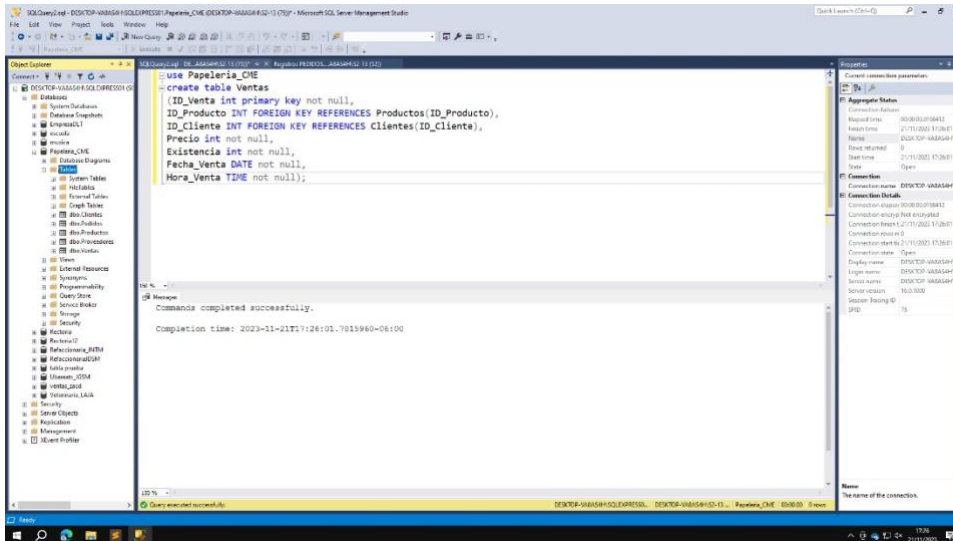


Tabla: Ventas

En este de ventas también necesitaras el “INT FOREIGN KEY REFERENCES” para los id_producto e id_Cliente, para el id_venta que es el principal usas el INT PRIMARY KEY, para el precio y

existencia se usa el “INT”, para la fech_venta usas el “DATE” y para la hora_venta se usa el “TIME” se ejecuta y refrescas la página para asegurarte de que si se guardó y se agregó a tu base de datos



Registros

Ya que verifiques que tengas tu tabla en la base agregaras el contenido, para esto ocupas el insert into y entre parentesis() y agregas los campos después agregas “values” y empiezas agregando el contenido que requieras para esta tabla y para que se agregue ocupas estas comillas ‘ ’ ya que se pinte rojo es porque si se va a agregar, para pasar a el otro campo utilizas

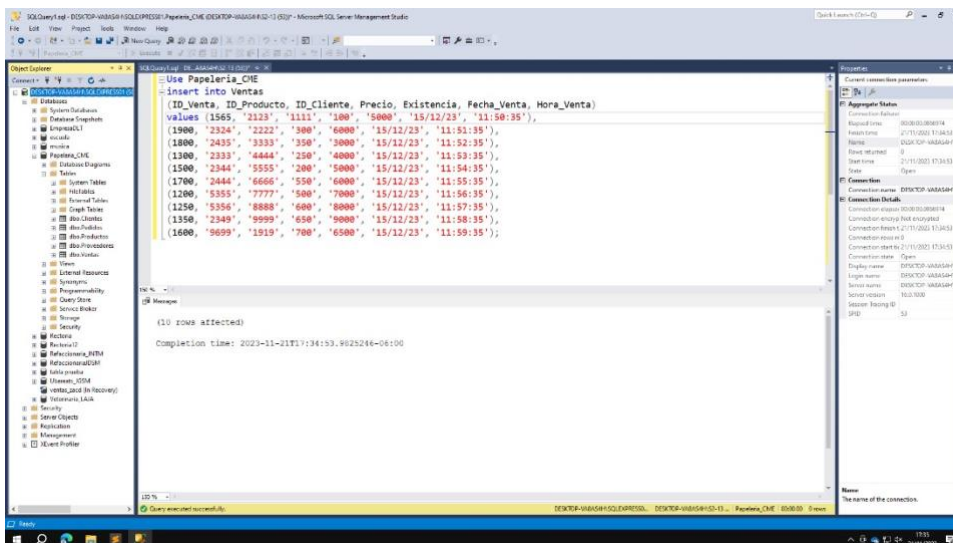


Tabla terminada

Ya que tengas todos los datos se ejecuta y refrescas, para verificar que si se agregó abres la tabla y veras todos los datos que agregaste

Tabla: Inventario

Ahora haremos la tabla “inventario” para eso agregas al código el “use” y el nombre de nuestra base, después agregas el código que va a crear la tabla que es “create table” y el nombre de esta tabla para este código el id_producto va a utilizar “INT FOREIGN KEY REFERENCES”, después agregas todos los otros campos, descripción utilizara el “varchar” y cantidad_entrada,cantidad_vendida y total restante van a utilizar el INT.

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The central pane shows a SQL query that has been executed successfully. The query creates a table named 'Inventario' with the following schema:

```

create table Inventario
(
    ID_Producto int FOREIGN KEY REFERENCES Productos(ID_Producto),
    Descripción Varchar (50),
    Cantidad_Entrada int not null,
    Cantidad_Vendida int not null,
    Total_Restante int not null);

```

The status bar at the bottom indicates that the commands were completed successfully and provides the completion time: 2023-11-21T17:36:09.9100902+06:00.

The right-hand pane shows the 'Properties' window for the 'Inventario' table, displaying various configuration options such as 'Compression', 'Collation', and 'Indexing'.

Registros

Después que agregues todo el código se ejecuta y refrescas para que te aparezca en la base de datos, ahora vas a abrir otro query para agregar la información utilizando “insert into” y agregaras el nombre de tu tabla, después agregarás los campos entre paréntesis (), al final para poder empezar usas “values” y empiezas a escribir

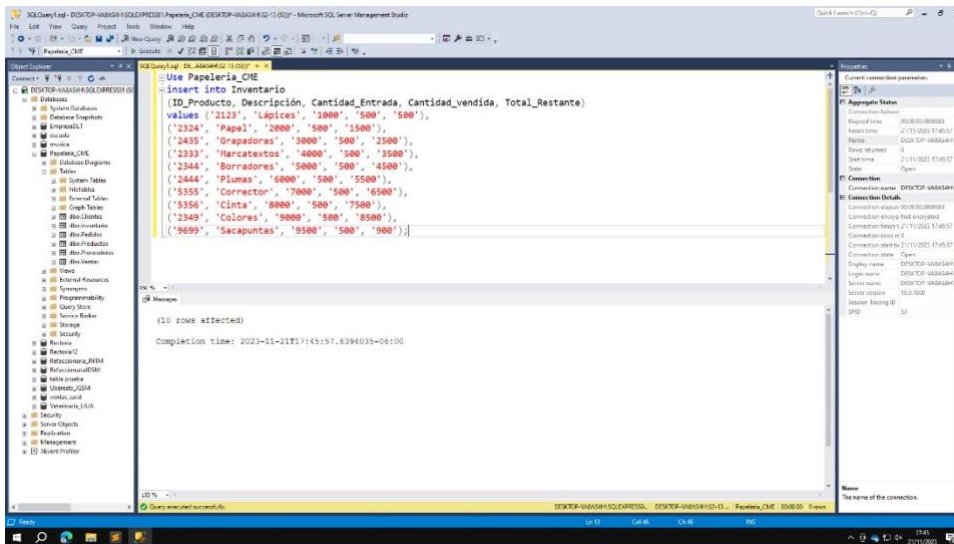


Tabla terminada

Ya que termines de agregar todo el código vas a ejecutar para que se te guarde en la base de datos y refrescaras para que te aparezca y poder editar, ahora abrirás la tabla y verificaras que se encuentren todos tus campos y en ellos los datos que ingresaste en el query anterior

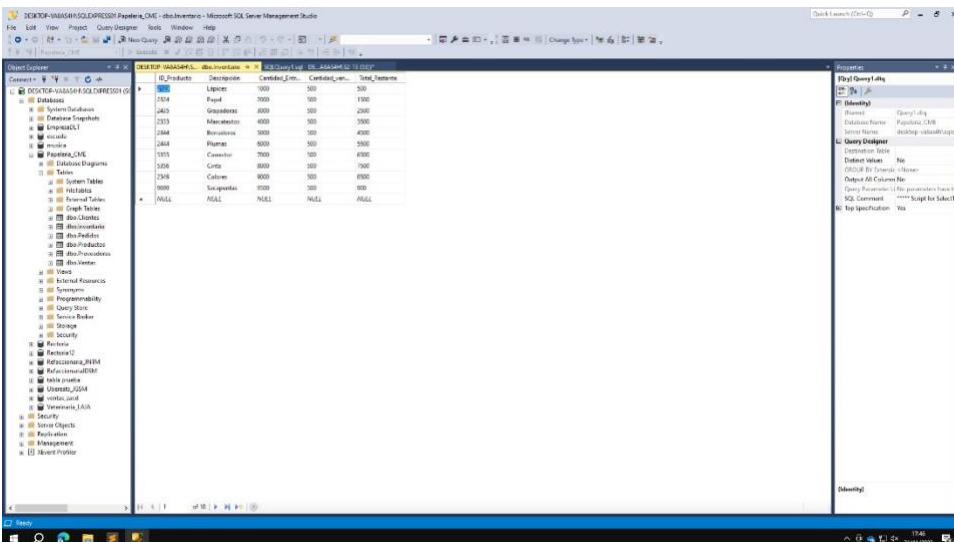
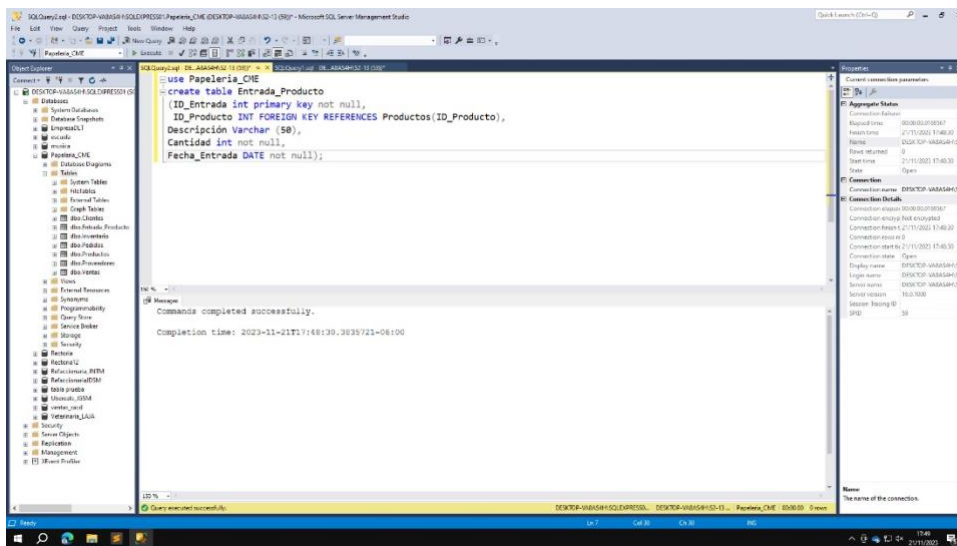


Tabla: ENTRADA_Producto

Ahora utilizaremos la tabla entrada_producto para crear esta tabla utilizaremos la tabla papeleria_CME, despues le daremos el valor a cada campo de la tabla como (descripción, cantidad y fecha_entrada)



Registros

En los siguientes campos (iD_entrada, ID_producto, descripcion, cantidad, fecha_entrada) pondremos los registros con insert into utilizando values, todo esto lo pondremos en la tabla entrada_producto que esta ejecutada en nuestra base de datos llamada papeleria_CME.

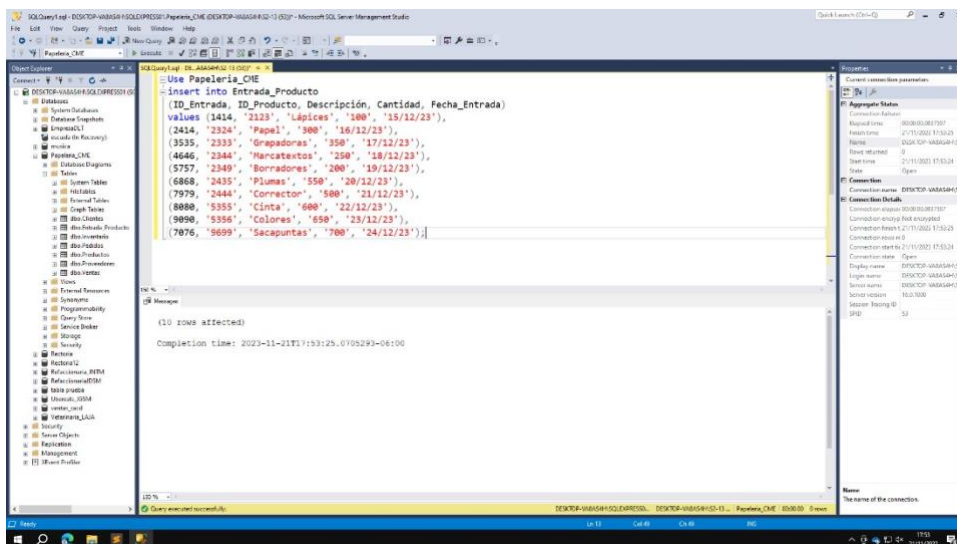


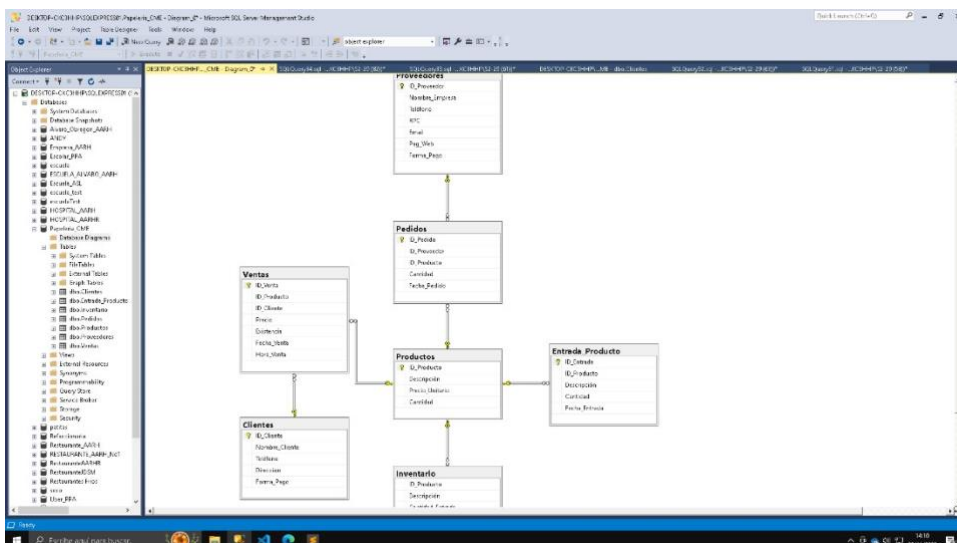
Tabla terminada

Despues de poner los registros en entrada_producto ejecutaremos la tabla para poder ver los registros ya reflejados en la tabla de nuestra papeleria_CME.

ID_Entrega	ID_Producto	Descripción	Cantidad	Fecha_Entrega
2102	2102	Lápices	100	2023-12-15
2104	2104	Papel	500	2023-12-16
2105	2105	Guadrapas	250	2023-12-17
2106	2106	Mecaneros	250	2023-12-18
2107	2107	Rotadores	200	2023-12-19
2108	2108	Pumas	350	2023-12-20
2109	2109	Sanguetas	700	2023-12-21
2110	2110	Cometas	300	2023-12-22
2111	2111	Cintas	600	2023-12-23
2112	2112	Cedulas	800	2023-12-24
2113	2113	Notas	1000	2023-12-25

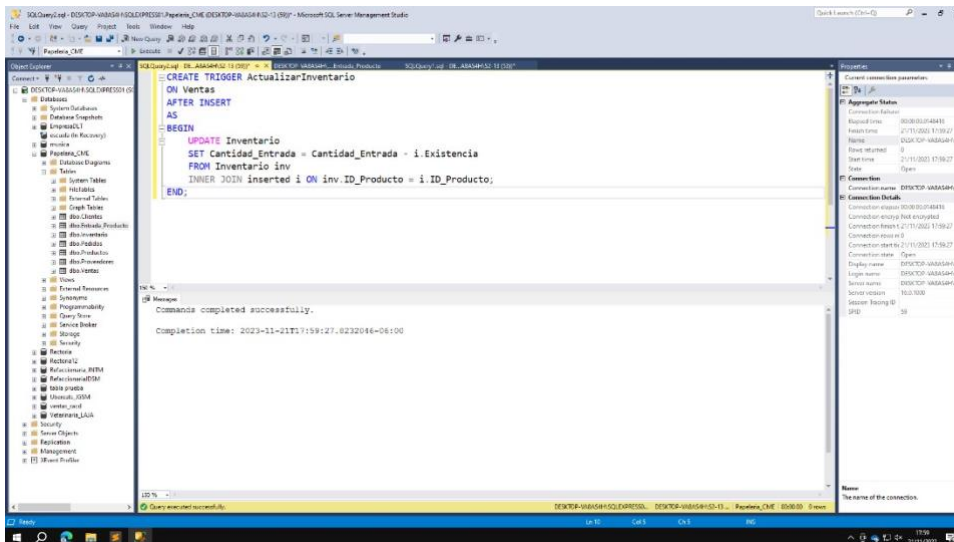
DIAGRAMA

En esta siguiente query conectaremos las tablas en forma de diagrama que creamos para la papeleria_CME en la cual se conecta proveedores,pedidos,productos,inventario,ventas, clientes y entrada_producto



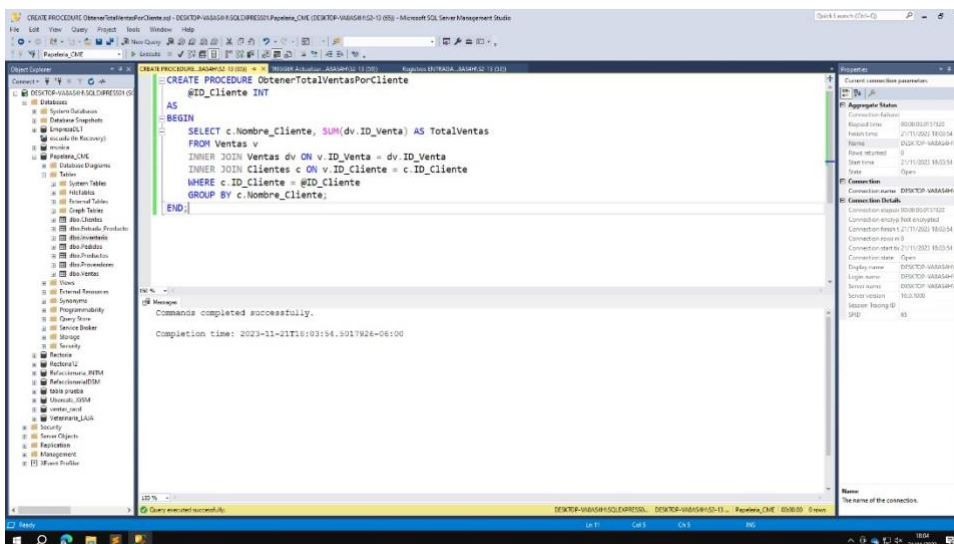
TRIGGER

Despues crearemos una tabla llamada actualizar inventario, utilizaremos tambien la tabla ventas para utilizar esos registros despues utilizaremos la tabla inventario para utilizar algunos campos que son conectados como cantidad_entrada = cantidad_entrada = i.existencia para despues utilizarlo en invenario inv y para finalizar tambien conectaremos inserted i con inv. Id_producto = i. id_producto.



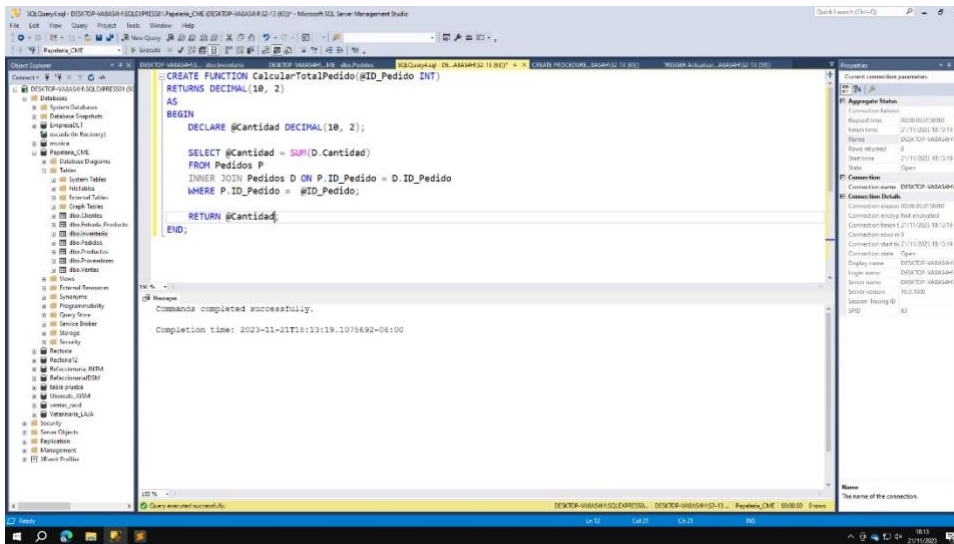
CREATE PROCEDURE

Despues de crear la tabla le daremos los valores a caa puerto, utilizaremos los id_venta, id_cliente_nombre: cliente para poder conectarlos para la creacion de la tabla



FUNCIONES

En esta tabla declararemos los decimales en cada cantidad, despues utilizaremos la tabla de pedidos para concetar los id, utilizaremos los id para conectarlos (id_pedido, id_pedido) y al final del codigo poder terminarlos con la tabla de cantidad.



CONSULTAS:

SELECT

Esta consulta nos sirve para ver todos los datos que se encuentran dentro de la tabla, sin exepciones, para utilizarla primero debemos llamar la base de datos de donde tomaremos la tabla, en este caso escribimos el nombre de la base de datos “Papeleria_CME”, debajo de esa linea pondremos la instrucción SELECT * FROM, esto se define como, “seleccionar todos los datos de la tabla ‘Pedidos ‘. Cuando ejecutamos el Query podemos ver que se nos han mostrado todos los registros como lo solicitamos.

The screenshot shows the SQL Server Enterprise Manager interface. The 'Pedidos' table is selected in the 'Object Explorer' on the left. The 'Query Editor' in the center displays the query 'SELECT * FROM Pedidos'. The 'Results' pane shows the following data:

ID_Pedido	ID_Proveedor	ID_Producto	Cantidad	Fecha_Pedido
1	5803	1212	2123	2023-12-19
2	5804	2121	2345	2023-12-23
3	5805	1787	2456	2023-12-20
4	5803	1818	5309	2023-12-21
5	5803	1213	2729	2023-12-18
6	5808	2323	9809	2023-12-24
7	5816	1414	2423	2023-12-17
8	5127	1616	2348	2023-12-19
9	5157	1915	2333	2023-12-19
10	5807	1919	5306	2023-12-22

SELECT.SUM

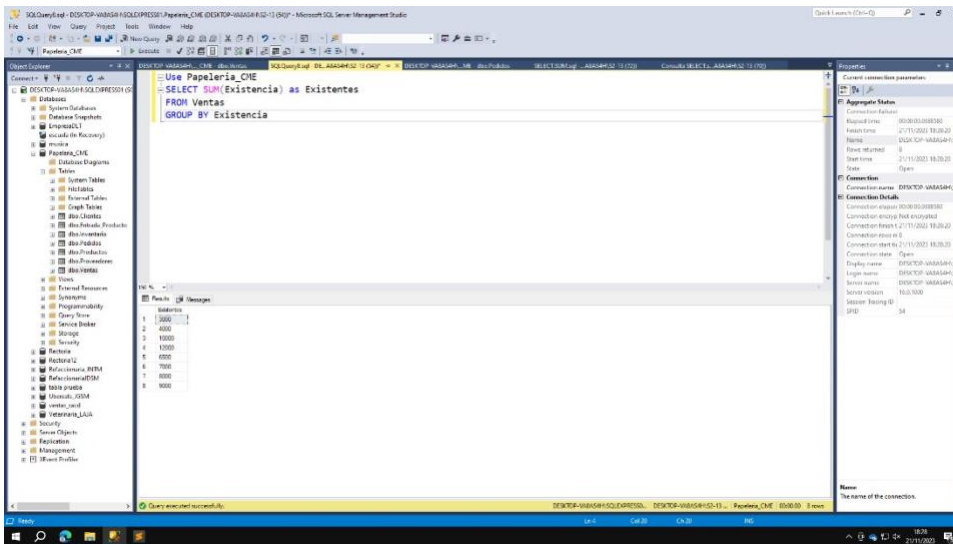
Esta consulta nos sirve para sumar datos numéricos de los registros que deseemos, en este caso sumamos la cantidad de pedidos de la tabla de Pedidos, y como Podemos ver, se muestra el total de la suma abajo.

The screenshot shows the SQL Server Enterprise Manager interface. The 'Query Editor' in the center displays the query 'SELECT SUM(Cantidad) as CantidadTotal FROM Pedidos'. The 'Results' pane shows the following data:

CantidadTotal
4055

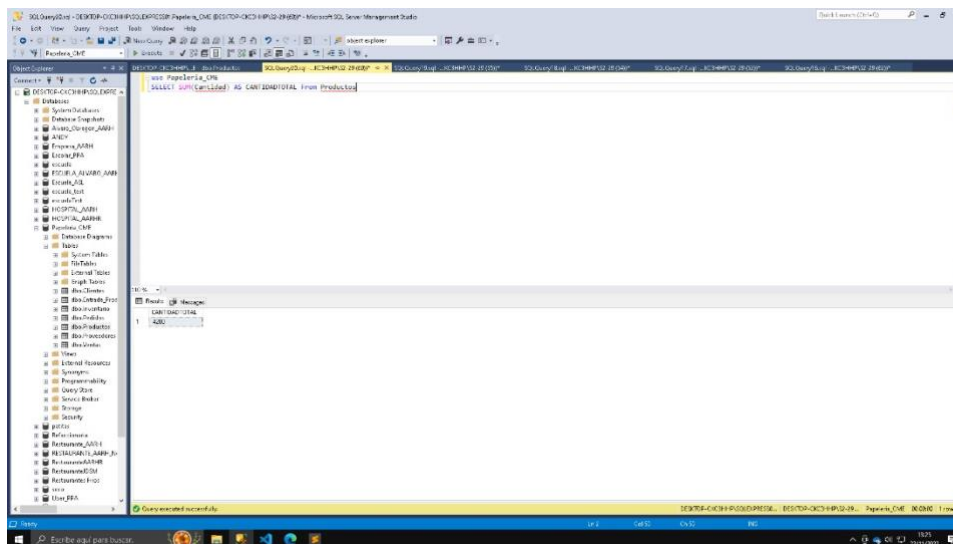
Sum Existencia as Existentes

La instrucción AS nos sirve para cambiar el nombre de alguna columna, para ello primero seleccionamos el campo o realizamos antes la instrucción necesaria, después agregamos AS y seguido, el nuevo nombre que se quiera asignar.



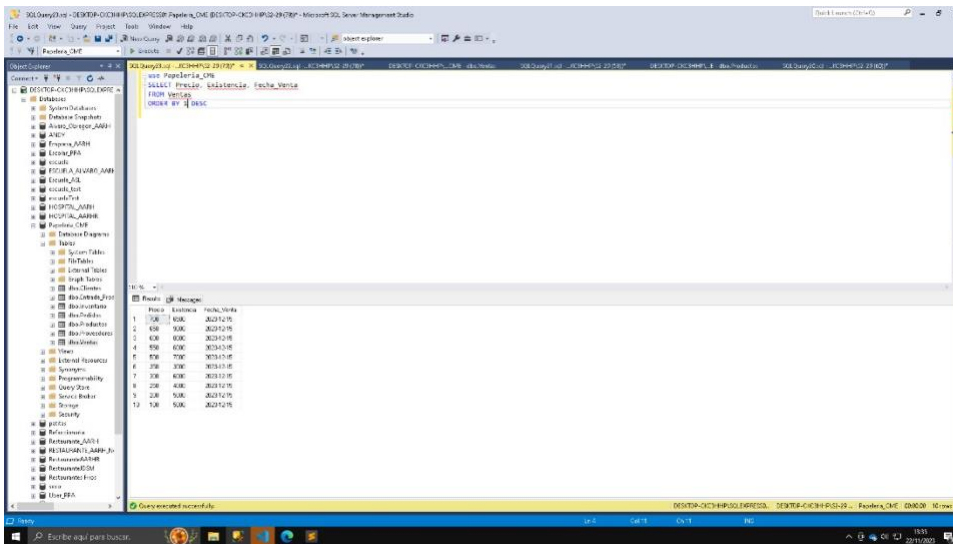
SELECT SUM(Cantidad) AS CANTIDADTOTAL

Usamos la instrucción SUM para sumar toda la cantidad total de productos, para después cambiarle el nombre a CANTIDADTOTAL, después nombramos la tabla de donde se obtendrán los datos, y se realiza correctamente lo que se solicita.



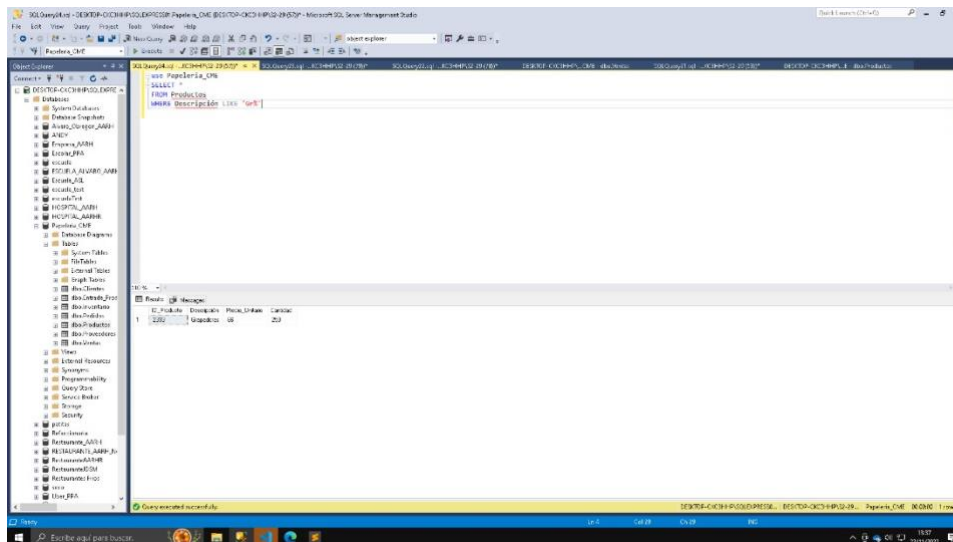
ORDER BY

Esta instrucción sirve para agrupar columnas de la tabla, aquí agrupamos la columna de Precio de la tabla de “Ventas” donde el precio sea “500” y ejecutamos el Query, en la parte de abajo podemos observar que se realizó correctamente lo que se solicitó.



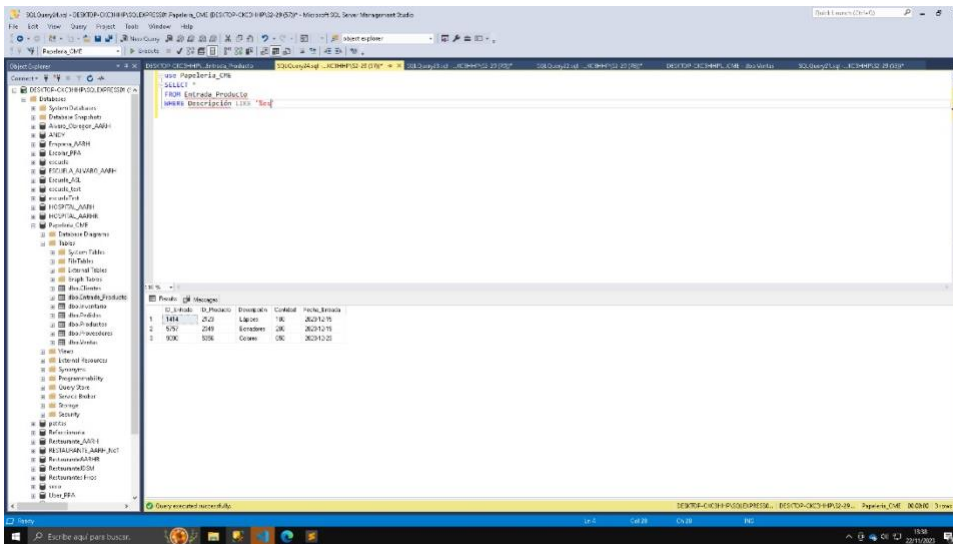
LIKE 'Gr%'

Esta instrucción LIKE, nos sirve para mostrar los datos del campo que especificaremos con WHERE, donde solo los cuales lleven en ellos “Gr” al inicio. En este caso. Nombramos la base de datos, y con SELECT, seleccionamos todos los datos de la tabla de Productos, donde se muestren los que tengan “Gr” al inicio de su escritura, dentro del campo de “Descripción”. Ejecutamos el código y como podemos observar, se realizó lo solicitado.



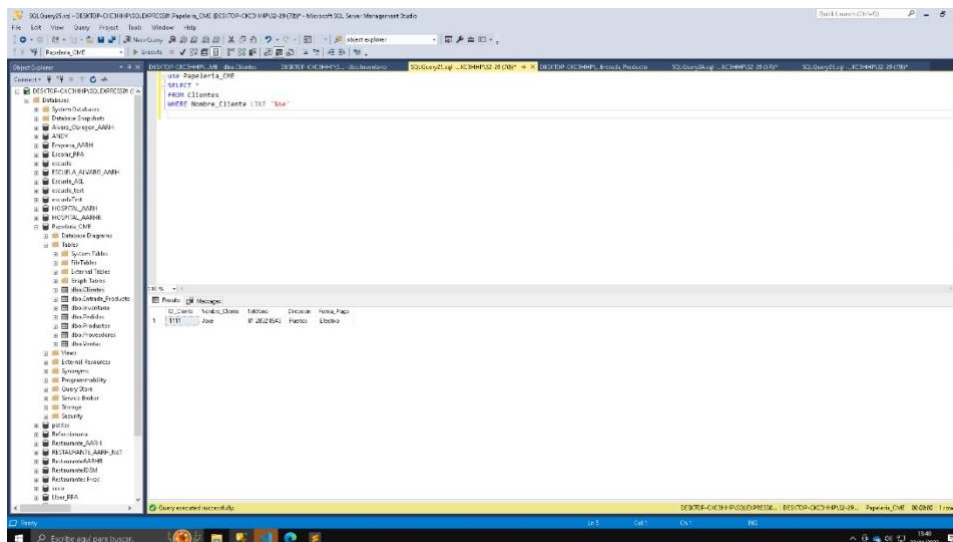
LIKE '%es'

Utilizamos la instrucción LIKE para encontrar los registros que lleven al final de su escrito “es”, estos serán de la tabla de Entrada_Producto y tomaremos los datos de la tabla de Descripción.



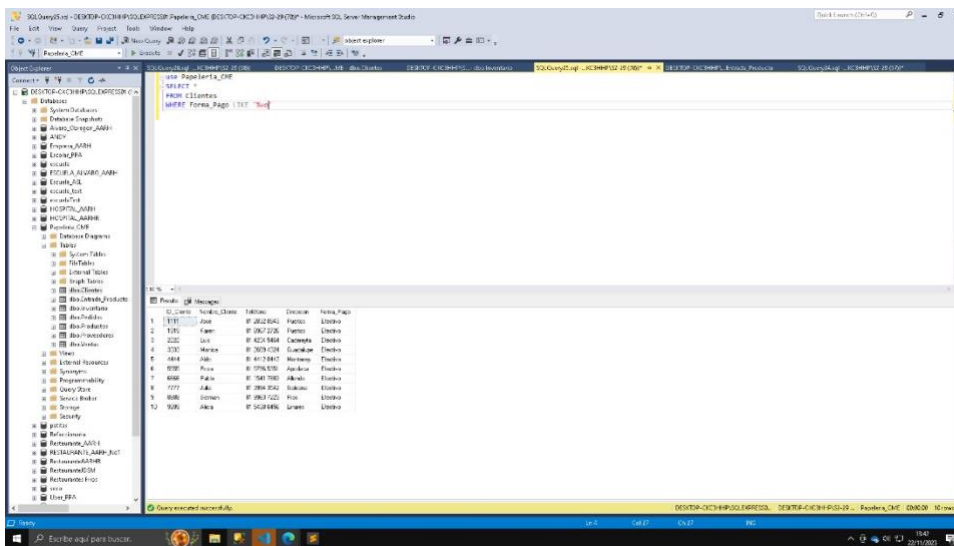
LIKE '%se '

Utilizamos la misma instrucción para seleccionar solamente los datos que contengan al final de su escrito las letras “se”, estos los elegiremos de la tabla de Clientes en el campo de “Nombre_Cliente”, al ejecutarlo se mostrarán solo los que cumplan.



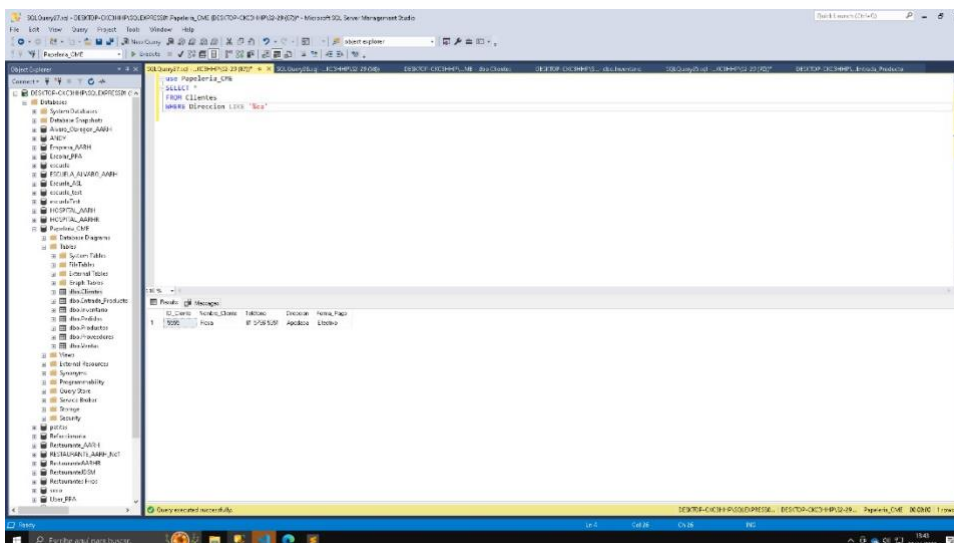
LIKE '%vo'

Utilizamos la instrucción LIKE para encontrar los registros que lleven al inicio de su escrito “vo”, estos serán de la tabla de Clientes y tomaremos los datos de la tabla de Forma_Pago.



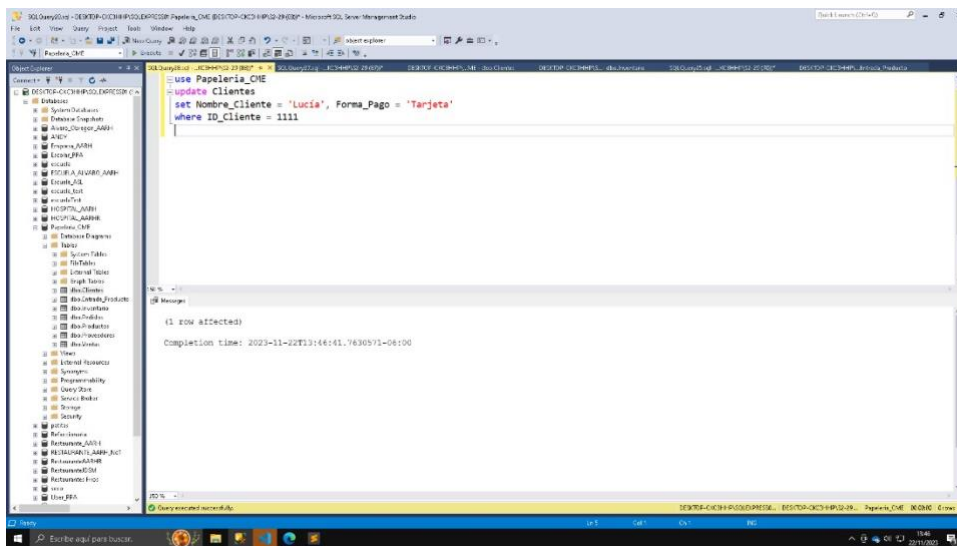
LIKE '%ca'

Utilizamos la instrucción LIKE para encontrar los registros que lleven al final de su escrito “ca”, estos serán de la tabla de Clientes y tomaremos los datos de la tabla de Dirección. Si lo ejecutamos, se mostrarán solo los registros que cumplan con esa condición.



UPDATE

Esta instrucción nos permite actualizar una parte de la tabla o algún registro de alguna de estas, aquí actualizamos de la tabla de Clientes el Nombre_Cliente a “Lucía”, y la Forma_Pago a “Tarjeta” del cliente que lleve por id “1111”. Al ejecutar correctamente se supone que los datos de la tabla deberían estar actualizados, entonces refrescamos la tabla y la ejecutamos para poder asegurarnos de que se haya realizado el cambio.



Antes:

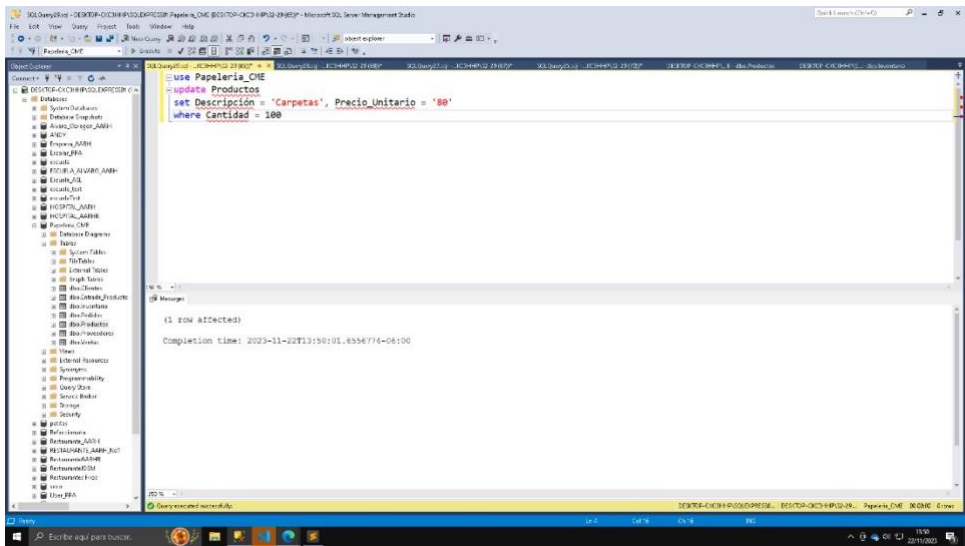
ID_Cliente	Nombre_Cliente	Forma_Pago	Forma_Pago
1111	Lucia	Tarjeta	Tarjeta

Después:

ID_Cliente	Nombre_Cliente	Forma_Pago	Forma_Pago
1111	Lucia	Tarjeta	Tarjeta

UPDATE descripción y precio unitario

Esta instrucción nos permite actualizar una parte de la tabla o algún registro de alguna de estas, aquí actualizamos de la tabla de Productos la Descripción a “Carpetas”, y el Precio_Unitario a “Precio_Unitario” donde la cantidad de estos sea igual a 100. Al ejecutar correctamente se supone que los datos de la tabla deberían estar actualizados



Antes:

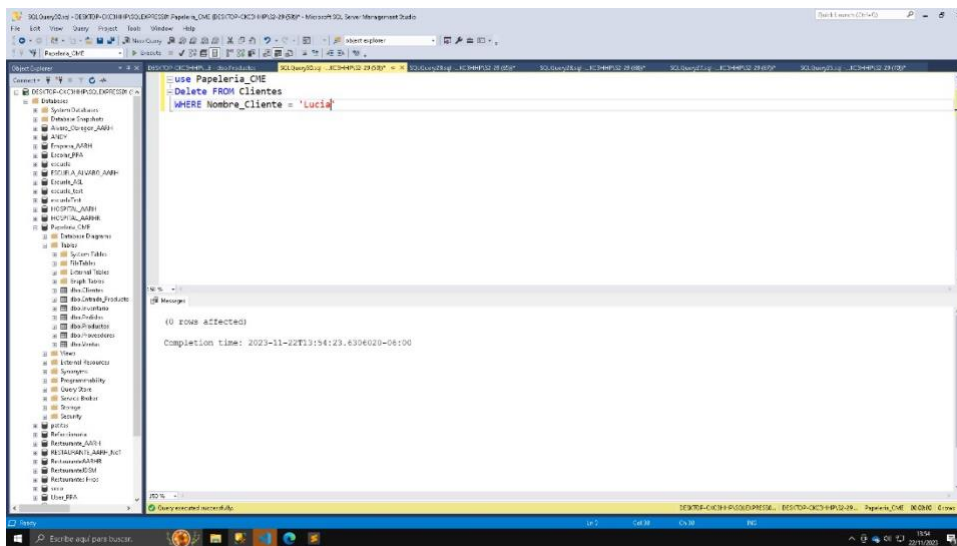
Después:

Id_Cliente	Nombre	Apellido	Fecha	Cantidad
100	Lucía	100		
101	Lucía	101		
102	Lucía	102		
103	Lucía	103		
104	Lucía	104		
105	Lucía	105		
106	Lucía	106		
107	Lucía	107		
108	Lucía	108		
109	Lucía	109		
110	Lucía	110		

Id_Cliente	Nombre	Apellido	Fecha	Cantidad
100	Lucía	100		
101	Lucía	101		
102	Lucía	102		
103	Lucía	103		
104	Lucía	104		
105	Lucía	105		
106	Lucía	106		
107	Lucía	107		
108	Lucía	108		
109	Lucía	109		
110	Lucía	110		

Delete

Con esta instrucción podemos eliminar los datos que deseemos de cualquier tabla, en este caso, de la tabla de Clientes se eliminará la fila donde el Nombre_Cliente sea igual a “Lucía”. Ejecutamos y refrescamos la tabla para asegurarnos que se haya realizado el cambio.



Antes:

The screenshot shows the 'Clientes' table in the 'Papeleria_OIE' database. The table contains the following data:

ID_Cliente	Nombre_Cliente	Apellido	Correo	Telefono
1001	Lucia	01234567	Papeteria	01234567
1002	Lucia	01234567	Papeteria	01234567
1003	Lucia	01234567	Papeteria	01234567
1004	Lucia	01234567	Papeteria	01234567
1005	Lucia	01234567	Papeteria	01234567
1006	Lucia	01234567	Papeteria	01234567
1007	Lucia	01234567	Papeteria	01234567
1008	Lucia	01234567	Papeteria	01234567
1009	Lucia	01234567	Papeteria	01234567
1010	Lucia	01234567	Papeteria	01234567

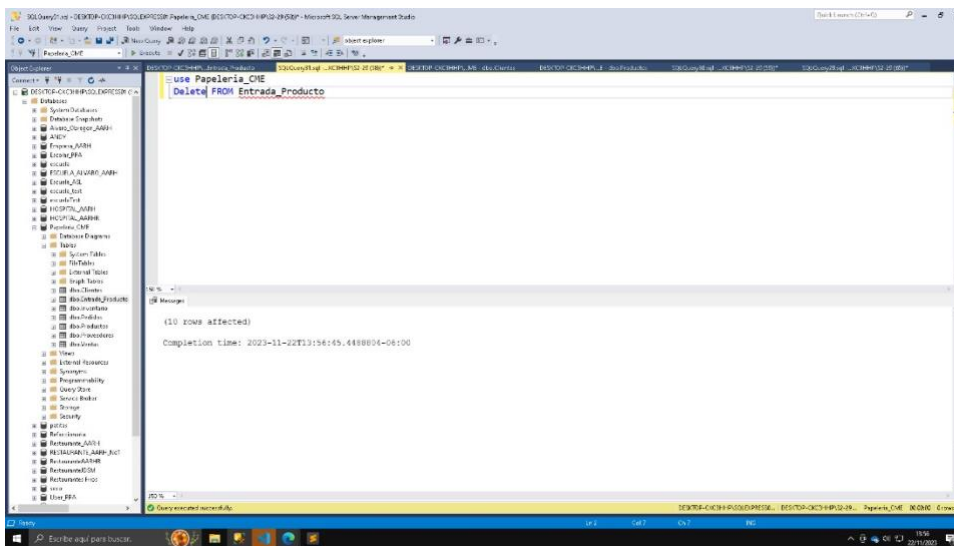
Después:

The screenshot shows the 'Clientes' table in the 'Papeleria_OIE' database after the deletion of the record where 'Nombre_Cliente' is 'Lucia'. The table is now empty.

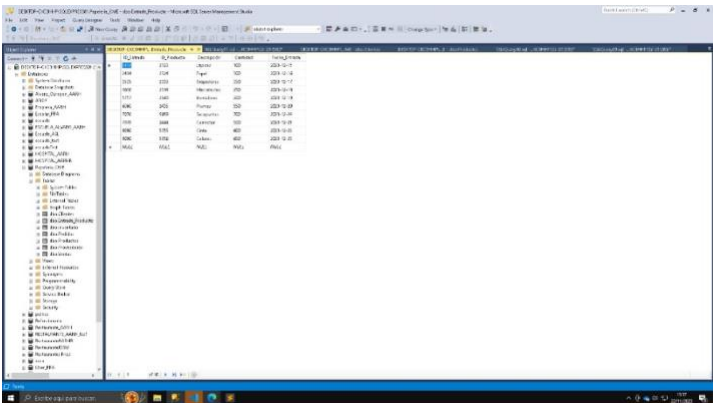
ID_Cliente	Nombre_Cliente	Apellido	Correo	Telefono
------------	----------------	----------	--------	----------

Delete FROM Entrada Producto

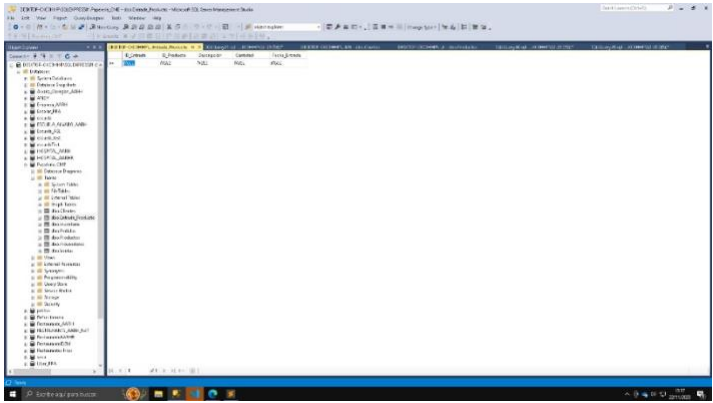
Esta instrucción solo especificando la tabla de donde se deberá borrar algo lo que hará será borrar todos los datos almacenados en la tabla sin exepción, dejando la tabla completamente vacia.



Antes:



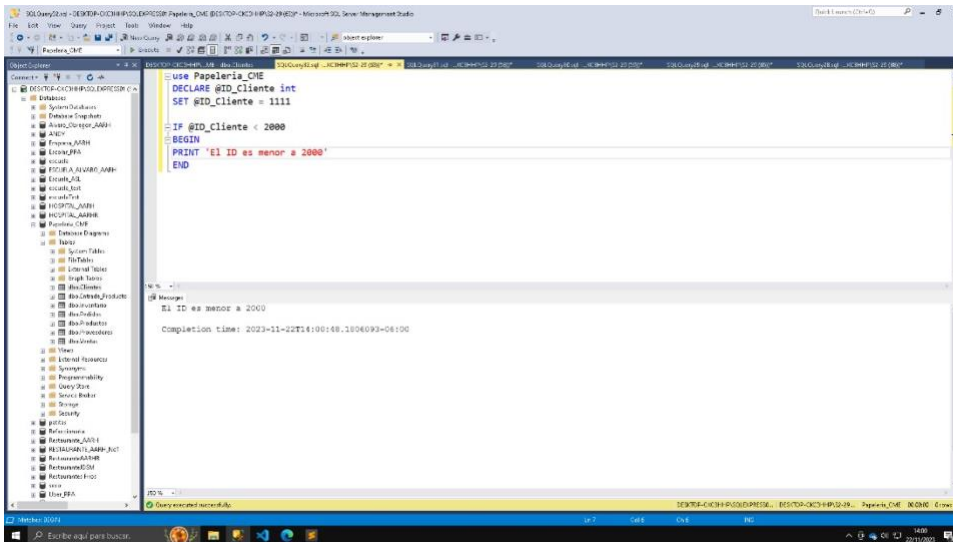
Después:



IF

Para esta consulta IF, primero definimos la base de datos donde se tomarán los datos, que es Papeleria_CME, y declaramos el valor @ID_Cliente int asignándole el valor de "1111", esta es la ID de un cliente. Después ponemos que, si @ID_Cliente es menor a 2000, entonces se deberá imprimir

que “El ID es menor a 2000”, finalizamos el Código y lo ejecutamos para verificar que se haya ejecutado correctamente y se imprima lo que especificamos.



```
use Papeleria_CHE
DECLARE @ID_Cliente int
SET @ID_Cliente = 1111

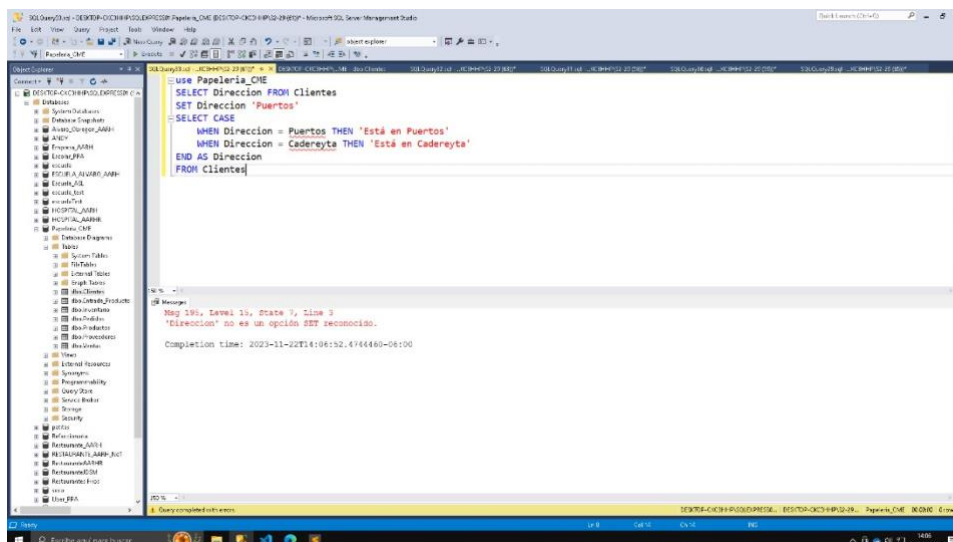
-- IF @ID_Cliente < 2000
-- BEGIN
-- PRINT 'El ID es menor a 2000'
-- END
```

Message: El ID es menor a 2000

Completion time: 2023-11-22T14:00:00.1046993+08:00

CASE

Para esta sentencia CASE, nombramos la base de datos donde se tomará la información y se selecciona el campo Dirección de la tabla de Clientes, Después declaramos que el valor de este campo será igual a “Puertos” y utilizamos la sentencia CASE para que, mientras Dirección sea ‘Puertos’, entonces se imprimirá que la persona de la que está en Puertos, y que si es Cadereyta entonces se imprimirá que está en Cadereyta.



```
use Papeleria_CHE
SELECT Direction FROM Clientes
SET Direction 'Puertos'
-- SELECT CASE
-- WHEN Direction = Puertos THEN 'Está en Puertos'
-- WHEN Direction = Cadereyta THEN 'Está en Cadereyta'
-- END AS Direction
FROM Clientes
```

Message: May 19, 2023, 14:06:52.4744440+08:00
'Direction' no es un opción SET reconocida.

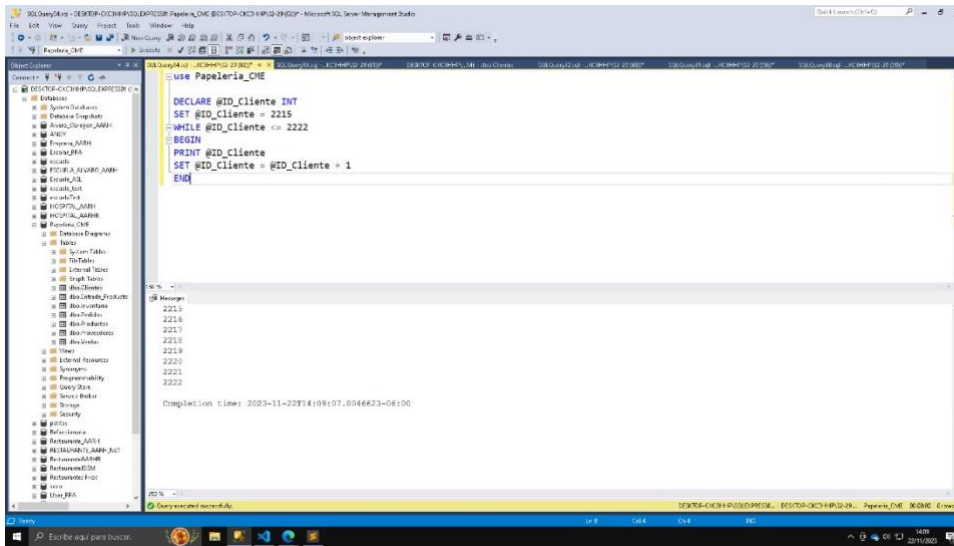
Completion time: 2023-11-22T14:06:52.4744440+08:00

WHILE

Para esta sentencia primero nombramos la base de datos de donde se deberán sacar los datos, y declaramos @ID_Cliente int y le daremos el valor de 2215, y después utilizando la sentencia WHILE, definimos que, mientras @ID_Cliente sea menor o igual a 2222, se imprimirá el

@ID_Cliente, pero mientras no lo sea se le sumará cada vez el 1 hasta llegar a la ID declarada, que es 2222, y cuando llegue hasta ese número se finaliza la instrucción.

Ejecutamos el Código y verificamos en la parte de abajo que se haya ejecutado correctamente.



```
USE Papeleria_CIE
DECLARE @ID_Cliente INT
SET @ID_Cliente = 2215
WHILE @ID_Cliente <= 2222
BEGIN
    PRINT @ID_Cliente
    SET @ID_Cliente = @ID_Cliente + 1
END
```

Msg
2215
2216
2217
2218
2219
2220
2221
2222

Completion Time: 2023-11-22T14:09:07.0346623+06:00

ZIP



Papeleria EMC_PIA_zip

[Papeleria EMC_PIA](#)