

Manual rápido de instalación y configuración de Microsoft SQL Server

Elaborado por el Ing. Marlon Guadalupe.

1. Instalación de Microsoft SQL Server

Ingresar a la página de descarga de SQL Server y descargar la versión Express.

<https://www.microsoft.com/es-MX/sql-server/sql-server-downloads#>

O descargue una edición especializada gratis



Developer

SQL Server 2019 Developer es una edición gratuita con todas las características, que se puede usar como base de datos de desarrollo y pruebas en los entornos no productivos.

[Descargar ahora ↓](#)

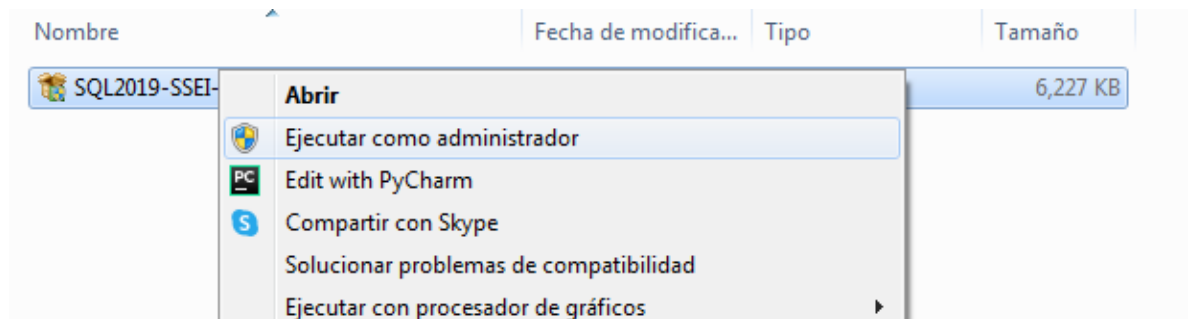


Express

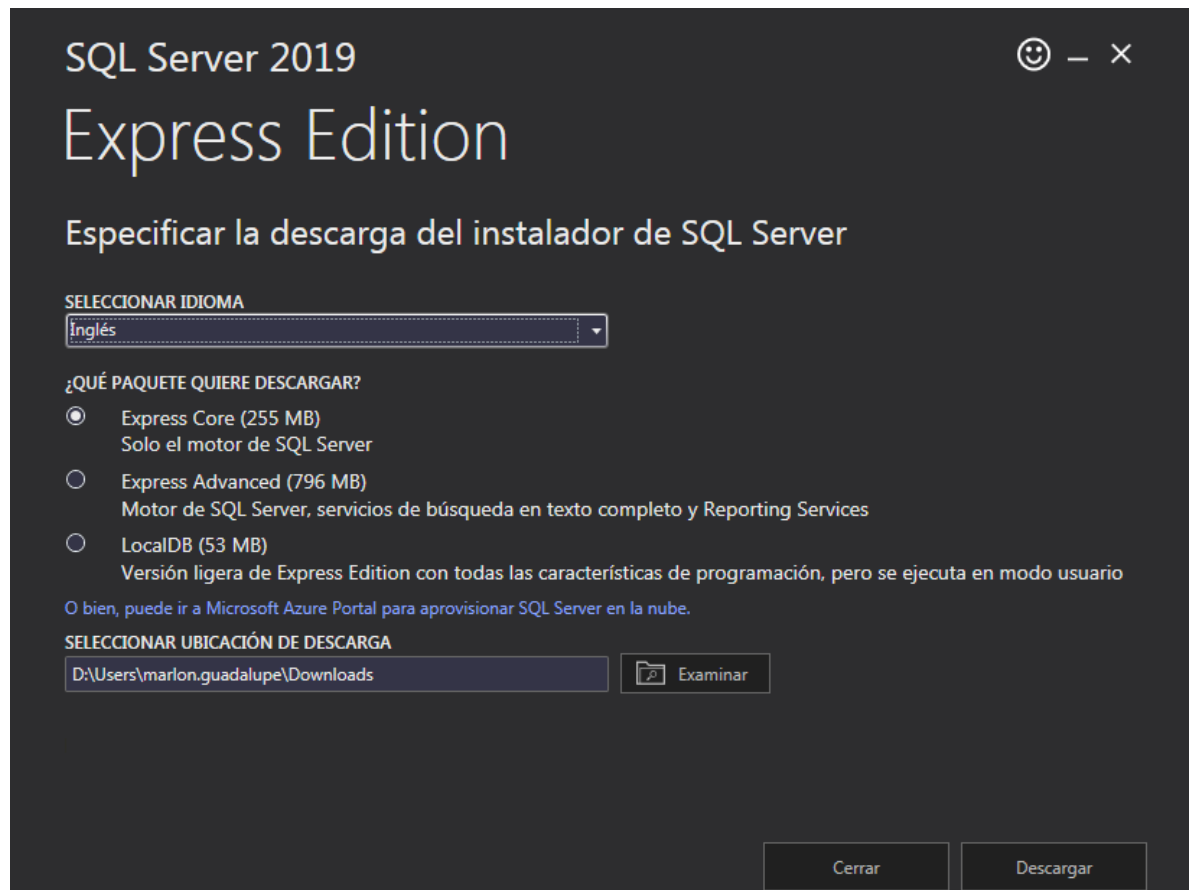
SQL Server 2019 Express es una edición gratuita de SQL Server, que es ideal para el desarrollo y la producción, para aplicaciones de escritorio, Internet y pequeños servidores.

[Descargar ahora ↓](#)

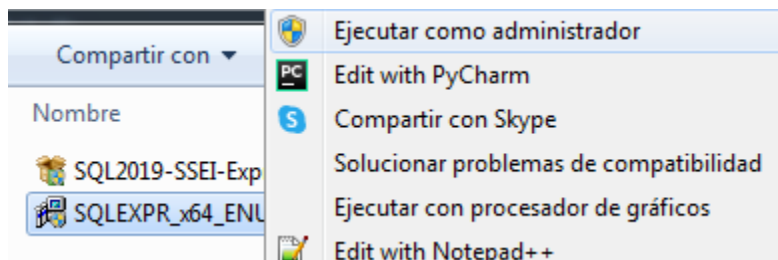
Ejecutar como administrador el archivo descargado.



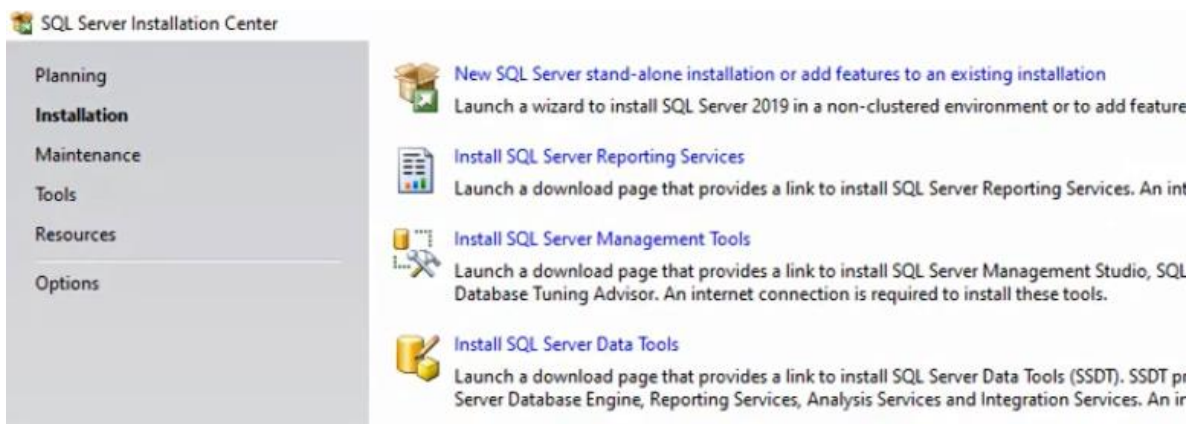
Seleccionar el idioma Inglés, el paquete Express Core y proceder a descargar.



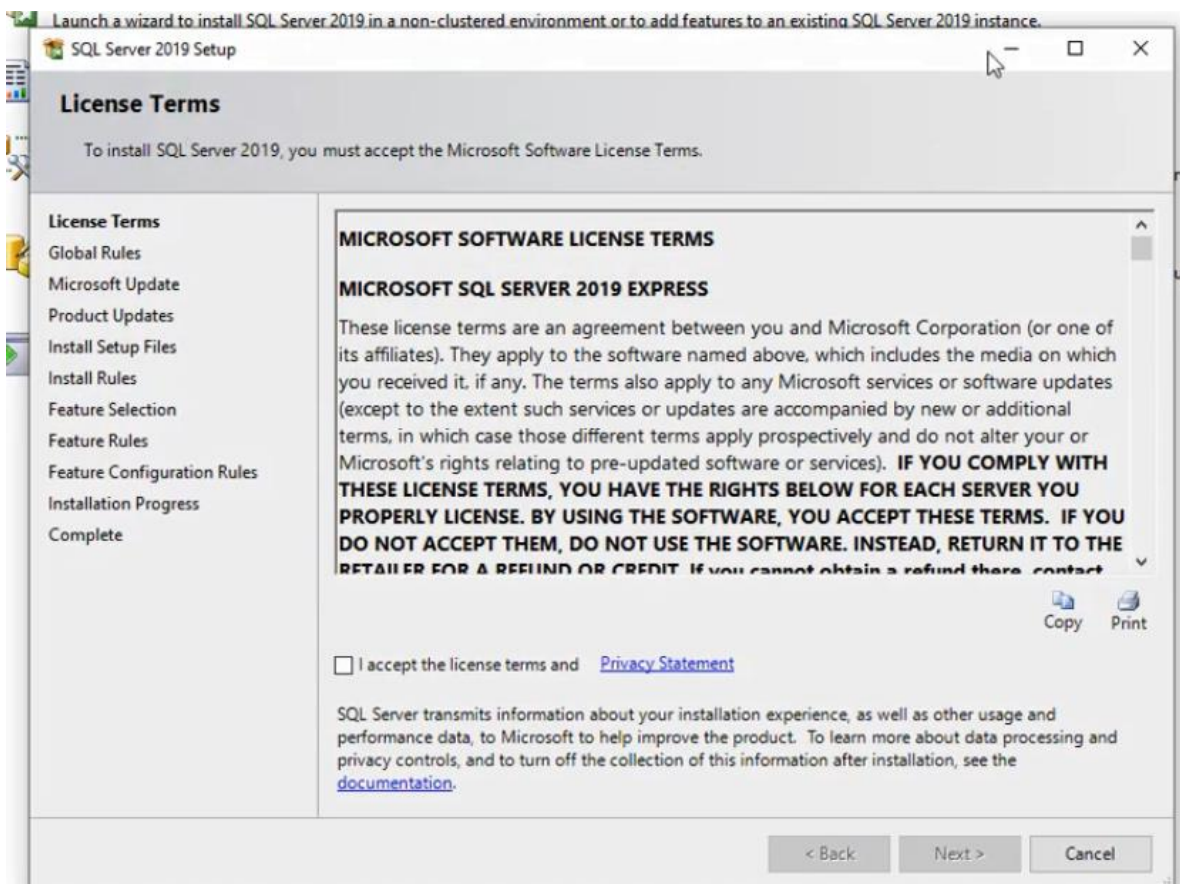
Ejecutar como administrador el archivo descargado.



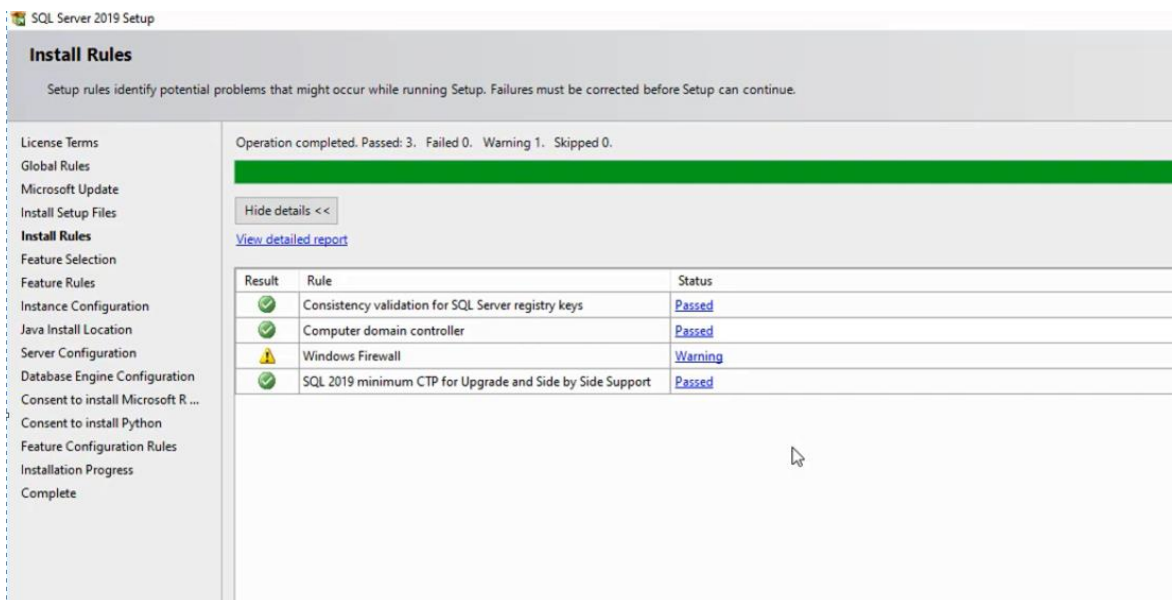
Nos dirigimos a Installation y New SQL Server satand-alone...



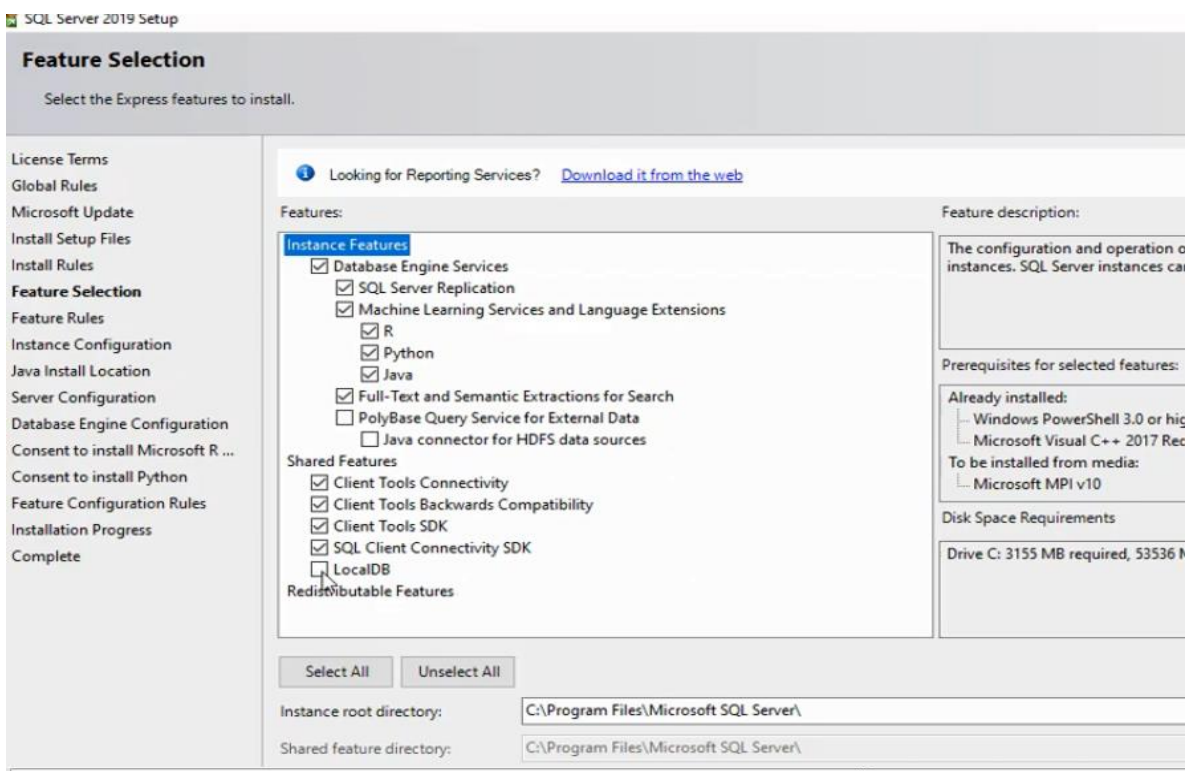
Acepta los términos y clic en Next.



Omitimos la advertencia y clic en Next.



Selecciona LocalDB y clic en Next.



Dejamos los valores por defecto y clic en Next.

The screenshot shows the 'Instance Configuration' window in the SQL Server 2019 Setup. The left sidebar lists various setup steps, with 'Instance Configuration' currently selected. The main area has a title bar and a subtitle: 'Specify the name and instance ID for the instance of SQL Server. Instance ID becomes part of the installation path.' Below this, there are two radio buttons: 'Default instance' (unselected) and 'Named instance:' (selected). The 'Named instance:' field contains the text 'SQLExpress'. Below this, the 'Instance ID:' field contains 'SQLEXPRESS'. The 'SQL Server directory:' field shows 'C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS'. The 'Installed instances:' section contains a table with three columns: 'Instance Name', 'Instance ID', and 'Features'. The table is currently empty.

SQL Server 2019 Setup

Instance Configuration

Specify the name and instance ID for the instance of SQL Server. Instance ID becomes part of the installation path.

License Terms
Global Rules
Microsoft Update
Install Setup Files
Install Rules
Feature Selection
Feature Rules
Instance Configuration
Server Configuration
Database Engine Configuration
Feature Configuration Rules
Installation Progress
Complete

☐ Default instance
☒ Named instance:

Instance ID:

SQL Server directory:

Installed instances:

Instance Name	Instance ID	Features
---------------	-------------	----------

Cambiamos el SQL Server Browser a Manual y clic en Next.

The screenshot shows the 'Server Configuration' window in the SQL Server 2019 Setup. The left sidebar lists various setup steps, with 'Server Configuration' currently selected. The main area has a title bar and a subtitle: 'Specify the service accounts and collation configuration.' Below this, there are two tabs: 'Service Accounts' (selected) and 'Collation'. The 'Service Accounts' tab contains a table with four columns: 'Service', 'Account Name', 'Password', and 'Startup Type'. The table lists three services: 'SQL Server Database Engine' (Automatic), 'SQL Full-text Filter Daemon Launcher' (Manual), and 'SQL Server Browser' (Manual). Below the table, there is a checkbox labeled 'Grant Perform Volume Maintenance Task privilege to SQL Server Database Engine Service'. A note below the checkbox states: 'This privilege enables instant file initialization by avoiding zeroing of data pages. This may lead to information disclosure by allowing deleted content to be accessed. Click here for details'.

SQL Server 2019 Setup

Server Configuration

Specify the service accounts and collation configuration.

License Terms
Global Rules
Microsoft Update
Install Setup Files
Install Rules
Feature Selection
Feature Rules
Instance Configuration
Server Configuration
Database Engine Configuration
Feature Configuration Rules
Installation Progress
Complete

Service Accounts Collation

Microsoft recommends that you use a separate account for each SQL Server service.

Service	Account Name	Password	Startup Type
SQL Server Database Engine	NT Service\MSSQL\$SQLEXPRESS		Automatic
SQL Full-text Filter Daemon Launcher	NT Service\MSSQLFDLauncher\$SQLEXPRESS		Manual
SQL Server Browser	NT AUTHORITY\LOCAL SERVICE		Manual

☐ Grant Perform Volume Maintenance Task privilege to SQL Server Database Engine Service

This privilege enables instant file initialization by avoiding zeroing of data pages. This may lead to information disclosure by allowing deleted content to be accessed.
[Click here for details](#)

Seleccionamos Mixed mode, colocamos una contraseña y clic en Next.

SQL Server 2019 Setup

Database Engine Configuration

Specify Database Engine authentication security mode, administrators, data directories, TempDB, Max degree of parallelism, Memory limits, and Filestream

License Terms
Global Rules
Microsoft Update
Install Setup Files
Install Rules
Feature Selection
Feature Rules
Instance Configuration
Server Configuration
Database Engine Configuration
Feature Configuration Rules
Installation Progress
Complete

Server Configuration | Data Directories | TempDB | Memory | User Instances | FILESTREAM

Specify the authentication mode and administrators for the Database Engine.

Authentication Mode

☐ Windows authentication mode

☒ Mixed Mode (SQL Server authentication and Windows authentication)

Specify the password for the SQL Server system administrator (sa) account.

Enter password: [password masked]

Confirm password: [password masked]

Specify SQL Server administrators

SQL Server administrator
DESKTOP-97OOLB3\vaishnavi (vaishnavi)

Esperamos a que termine la instalación

SQL Server 2019 Setup

Complete

Your SQL Server 2019 installation completed successfully.

License Terms
Global Rules
Microsoft Update
Install Setup Files
Install Rules
Feature Selection
Feature Rules
Instance Configuration
Server Configuration
Database Engine Configuration
Feature Configuration Rules
Installation Progress
Complete

Information about the Setup operation or possible next steps:

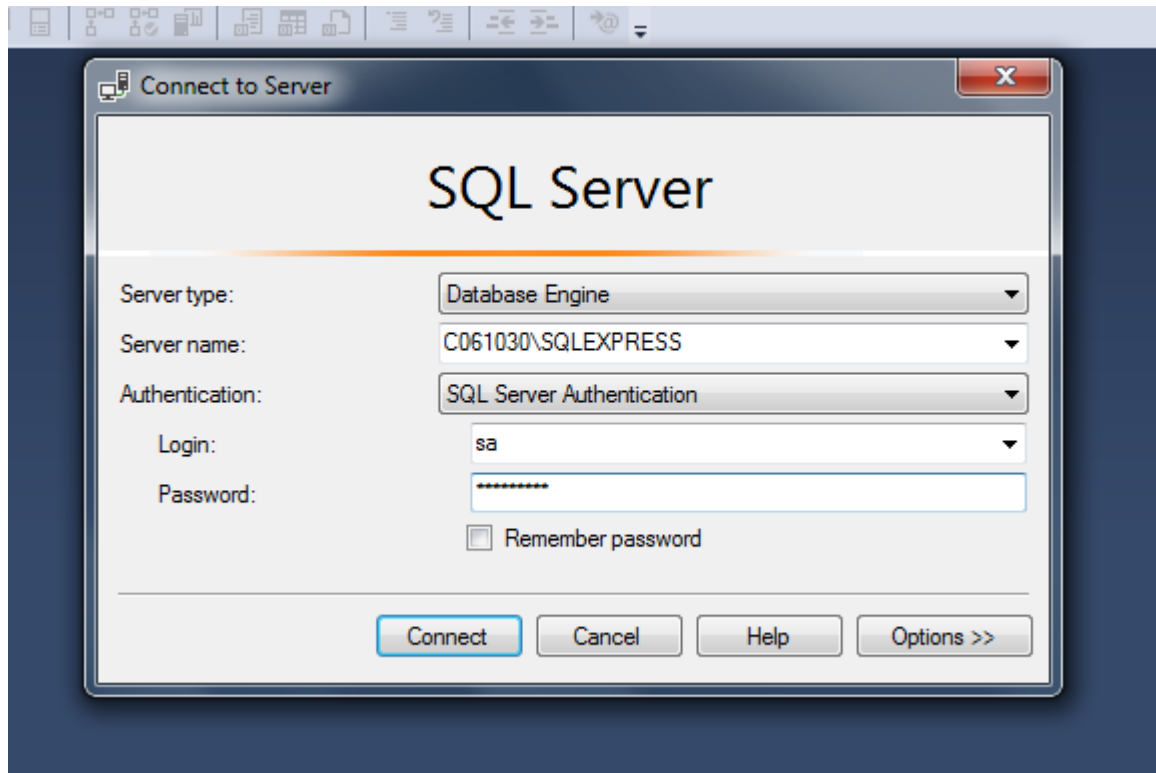
Feature	Status
Full-Text and Semantic Extractions for Search	Succeeded
Database Engine Services	Succeeded
SQL Server Replication	Succeeded
SQL Browser	Succeeded
SQL Writer	Succeeded
LocalDB	Succeeded
Client Tools Backwards Compatibility	Succeeded
Client Tools SDK	Succeeded
Client Tools Connectivity	Succeeded
SQL Client Connectivity SDK	Succeeded

Details:

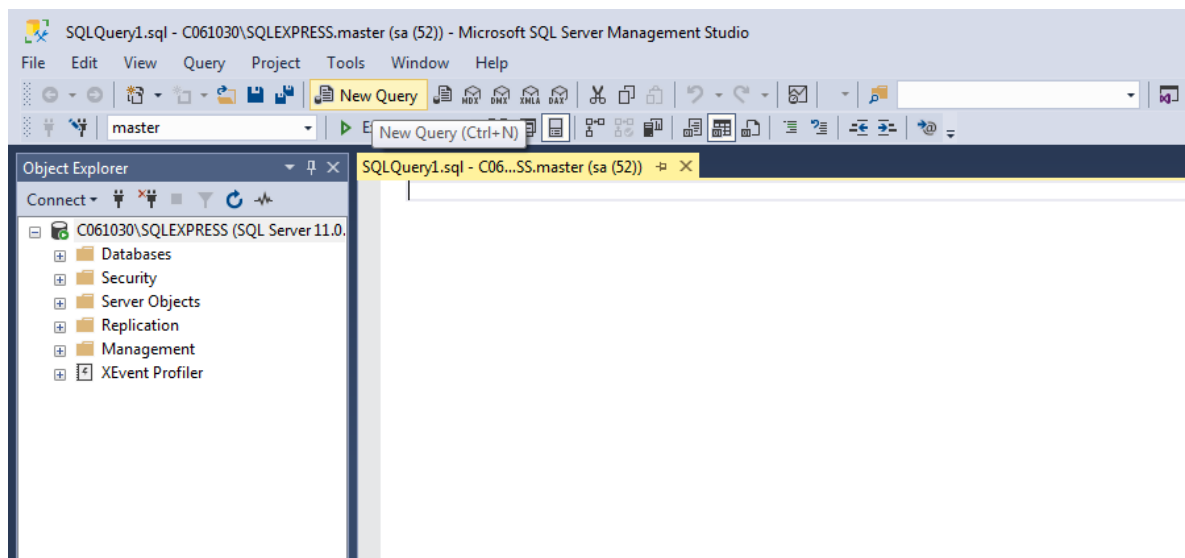
Install successful.

2. Gestión de Bases de Datos

Ingresamos a la base de datos local mediante el SSMM (los detalles se encuentran en el manual correspondiente.)



Seleccionar New Query.

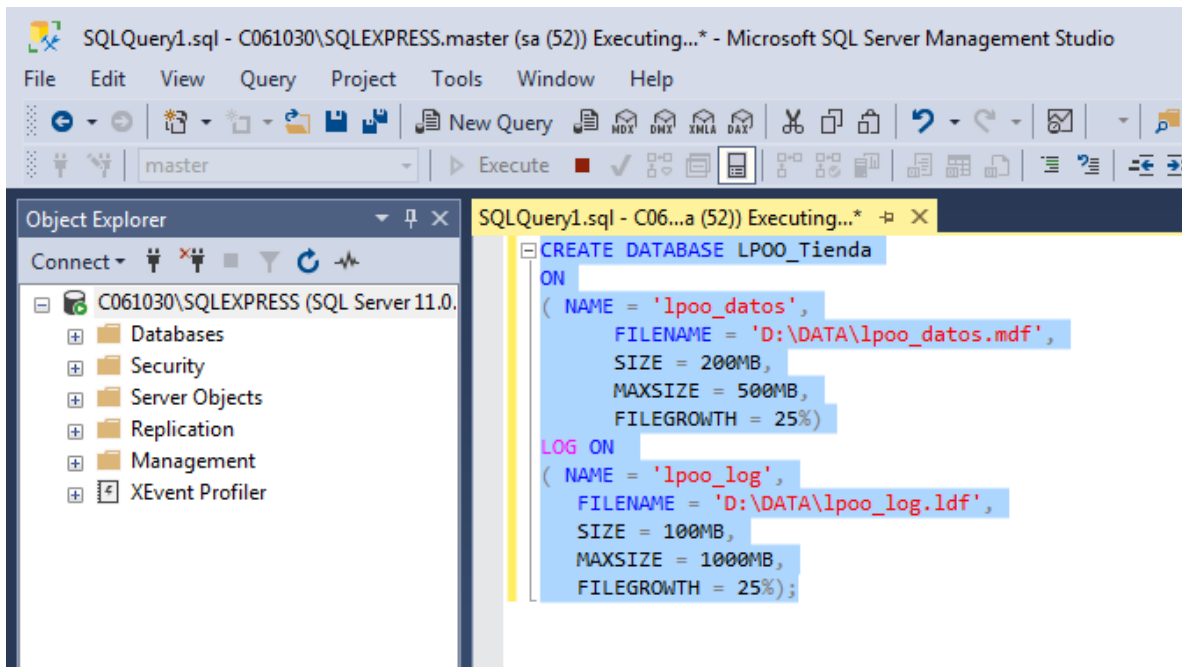


2.1. Creación de Base de datos:

Ejecutamos el comando:

```
CREATE DATABASE LP00_Tienda
ON
( NAME = 'lpoo_datos',
  FILENAME = 'D:\DATA\lpoo_datos.mdf',
  SIZE = 200MB,
  MAXSIZE = 500MB,
  FILEGROWTH = 25%)
LOG ON
( NAME = 'lpoo_log',
  FILENAME = 'D:\DATA\lpoo_log.ldf',
  SIZE = 100MB,
  MAXSIZE = 1000MB,
  FILEGROWTH = 25%);
```

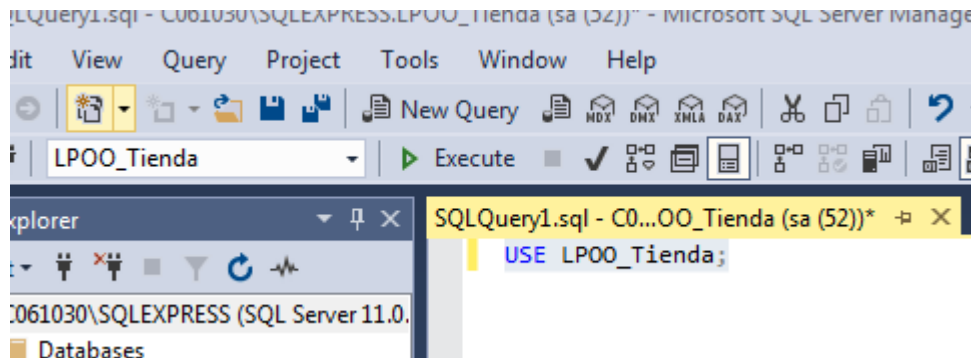
Nota: La dirección D:\DATA\ puede ser modificada por una válida para su computador respectivo.



2.2.Creación de Tablas.

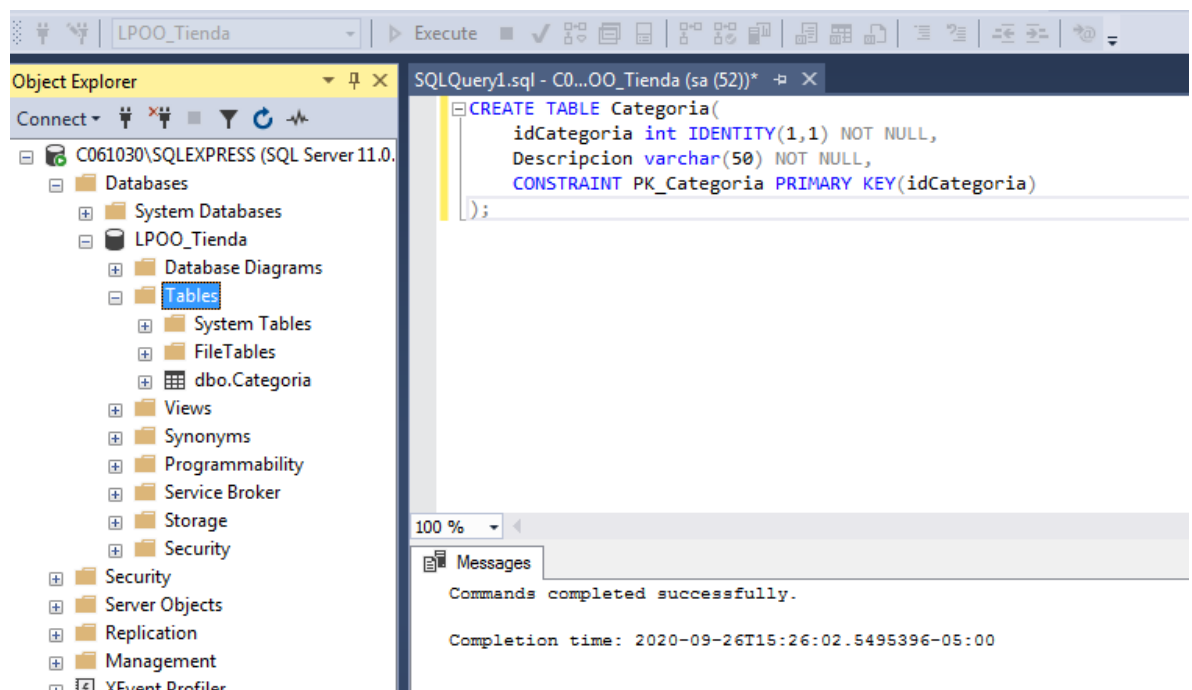
Seleccionamos la base de datos a usar con el comando USE:

```
USE LP00_Tienda;
```



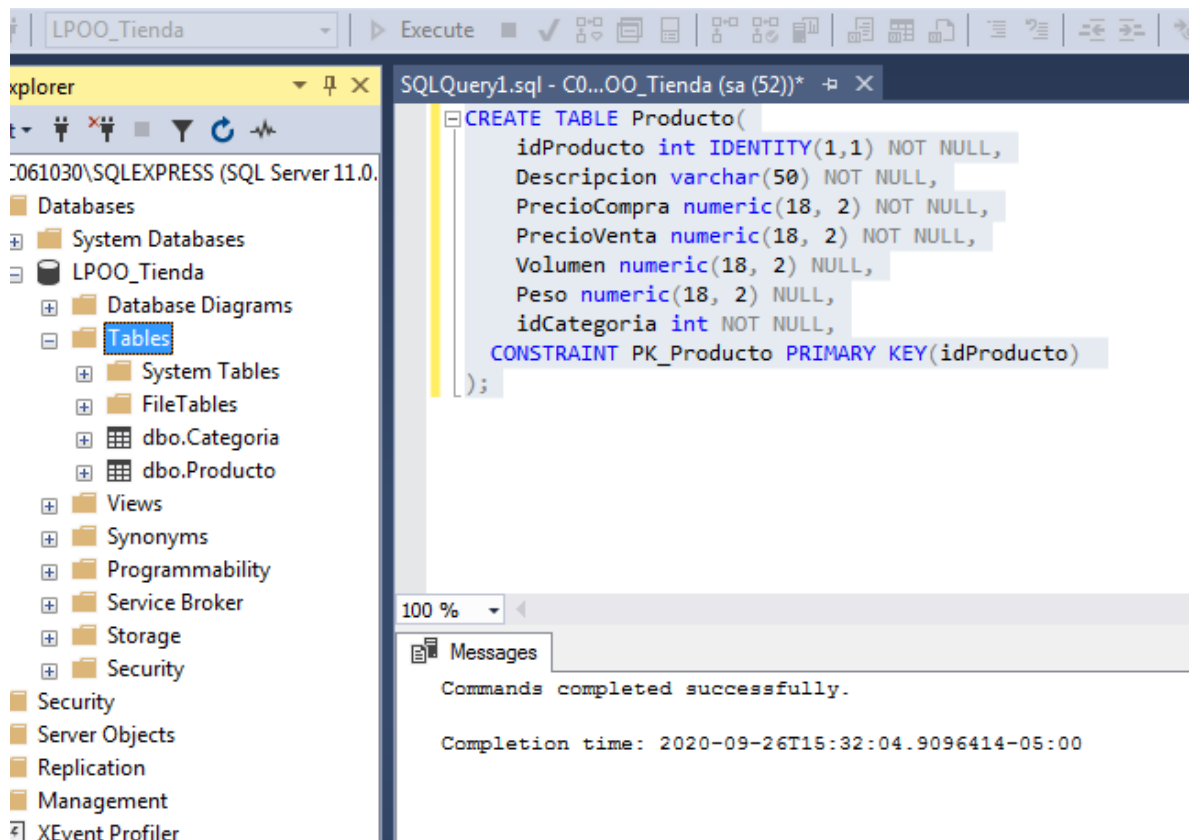
Ejecutamos el siguiente comando para crear la tabla Producto.

```
CREATE TABLE Categoria(  
    idCategoria int IDENTITY(1,1) NOT NULL,  
    Descripcion varchar(50) NOT NULL,  
    CONSTRAINT PK_Categoria PRIMARY KEY(idCategoria)  
);
```



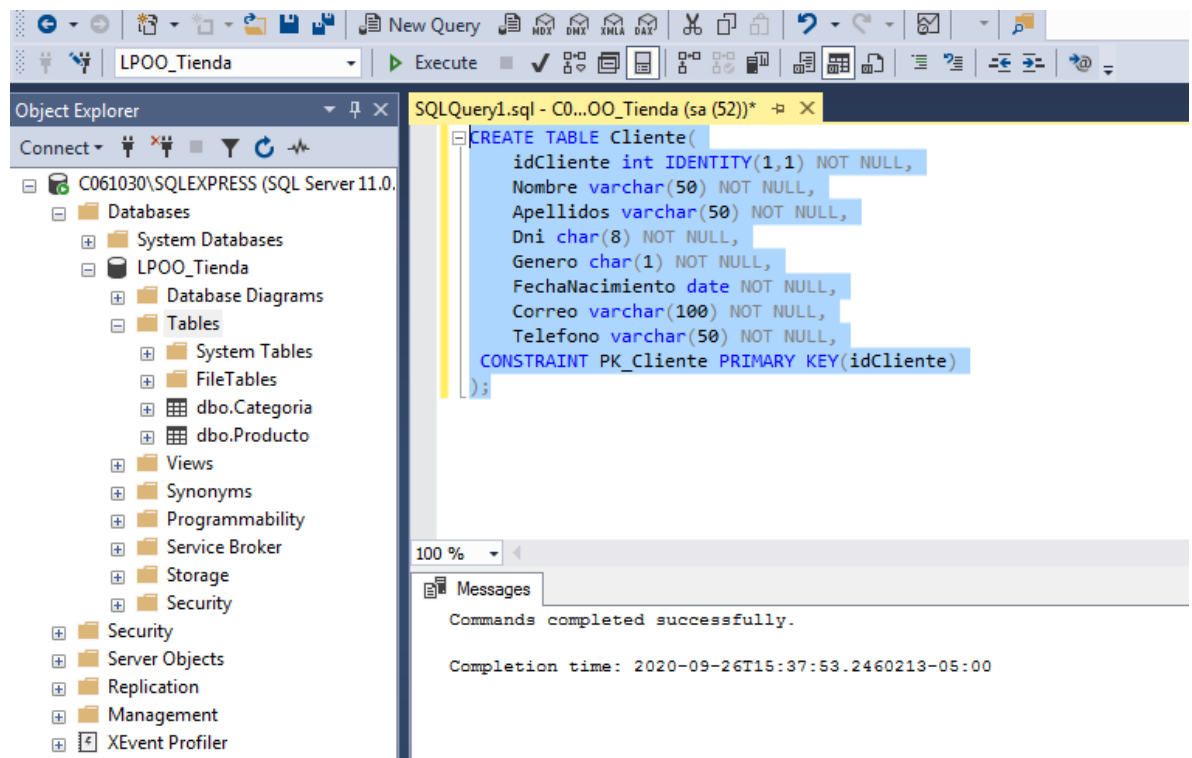
Ejecutamos el siguiente comando para crear la tabla producto:

```
CREATE TABLE Producto(  
    idProducto int IDENTITY(1,1) NOT NULL,  
    Descripcion varchar(50) NOT NULL,  
    PrecioCompra numeric(18, 2) NOT NULL,  
    PrecioVenta numeric(18, 2) NOT NULL,  
    Volumen numeric(18, 2) NULL,  
    Peso numeric(18, 2) NULL,  
    idCategoria int NOT NULL,  
    CONSTRAINT PK_Producto PRIMARY KEY(idProducto)  
);
```



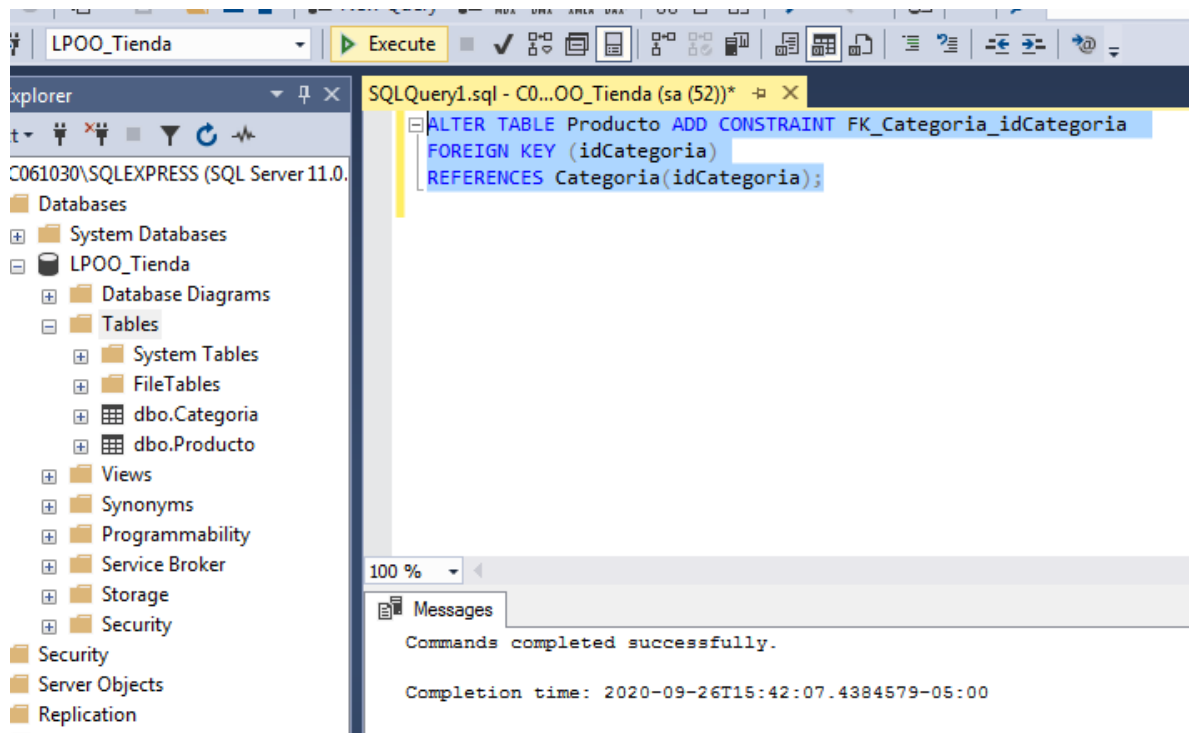
Ejecutamos el siguiente comando para crear la tabla Cliente:

```
CREATE TABLE Cliente(  
    idCliente int IDENTITY(1,1) NOT NULL,  
    Nombre varchar(50) NOT NULL,  
    Apellidos varchar(50) NOT NULL,  
    Dni char(8) NOT NULL,  
    Genero char(1) NOT NULL,  
    FechaNacimiento date NOT NULL,  
    Correo varchar(100) NOT NULL,  
    Telefono varchar(50) NOT NULL,  
    CONSTRAINT PK_Cliente PRIMARY KEY(idCliente)  
);
```



Ejecutamos el siguiente query para establecer la relación entre categoría y producto (llave foránea).

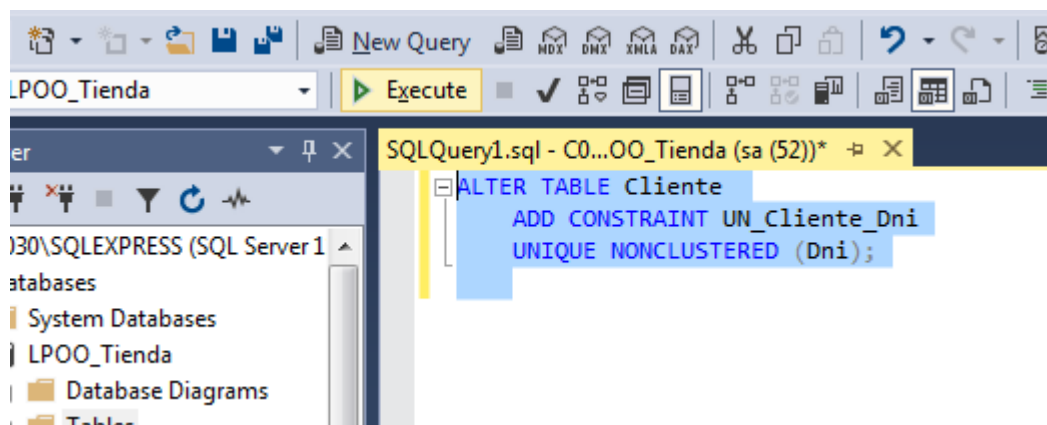
```
ALTER TABLE Producto ADD CONSTRAINT FK_Categoria_idCategoria  
FOREIGN KEY (idCategoria)  
REFERENCES Categoria(idCategoria);
```



2.3. Creación de Constraints (Restricciones)

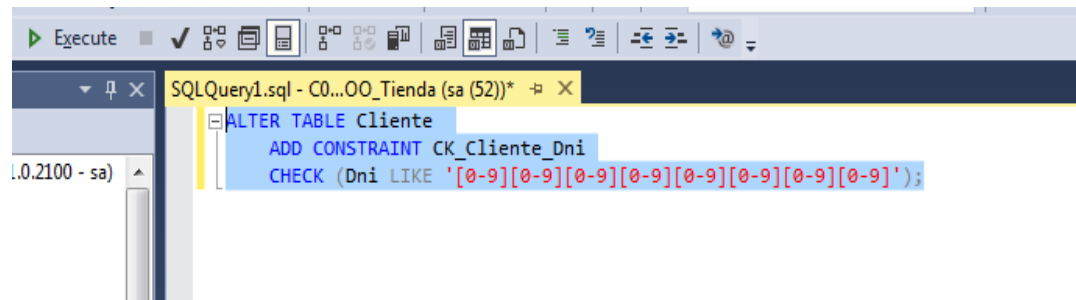
UNIQUE: Estable que el campo es único y no puede repetirse dentro de la tabla.

```
ALTER TABLE Cliente
ADD CONSTRAINT UN_Cliente_Dni
UNIQUE NONCLUSTERED (Dni);
```

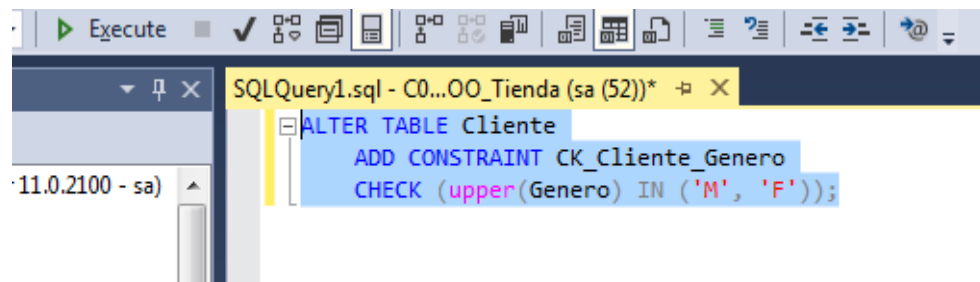


CHECK: Es una restricción que exige la integridad del dominio al limitar los valores posibles que se pueden escribir en una o varias columnas.

```
ALTER TABLE Cliente
  ADD CONSTRAINT CK_Cliente_Dni
  CHECK (Dni LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]');
```

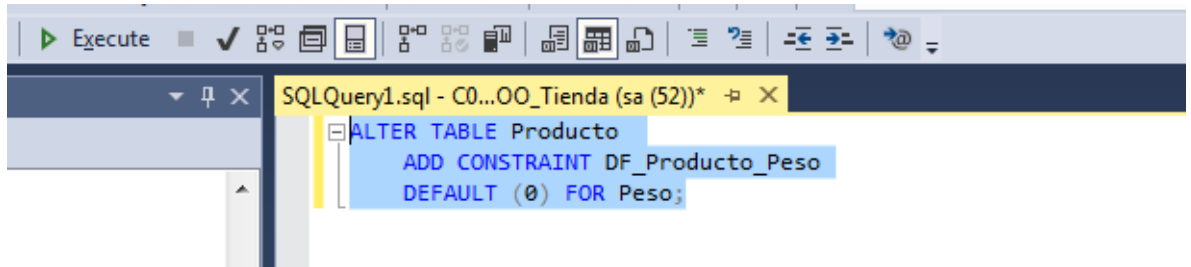


```
ALTER TABLE Cliente
  ADD CONSTRAINT CK_Cliente_Genero
  CHECK (upper(Genero) IN ('M', 'F'));
```



DEFAULT: Valor por defecto cuando no se ingresa el dato.

```
ALTER TABLE Producto
ADD CONSTRAINT DF_Producto_Peso
DEFAULT (0) FOR Peso;
```



2.4. Inserción de Datos:

Tabla Categorías:

```
INSERT INTO Categoria(Descripcion)
VALUES('Gaseosa'),
      ('Galleta'),
      ('Snacks'),
      ('Cerveza'),
      ('Vino'),
      ('Detergente'),
      ('Jabón'),
      ('Shampoo'),
      ('Fideo'),
      ('Atún');
```

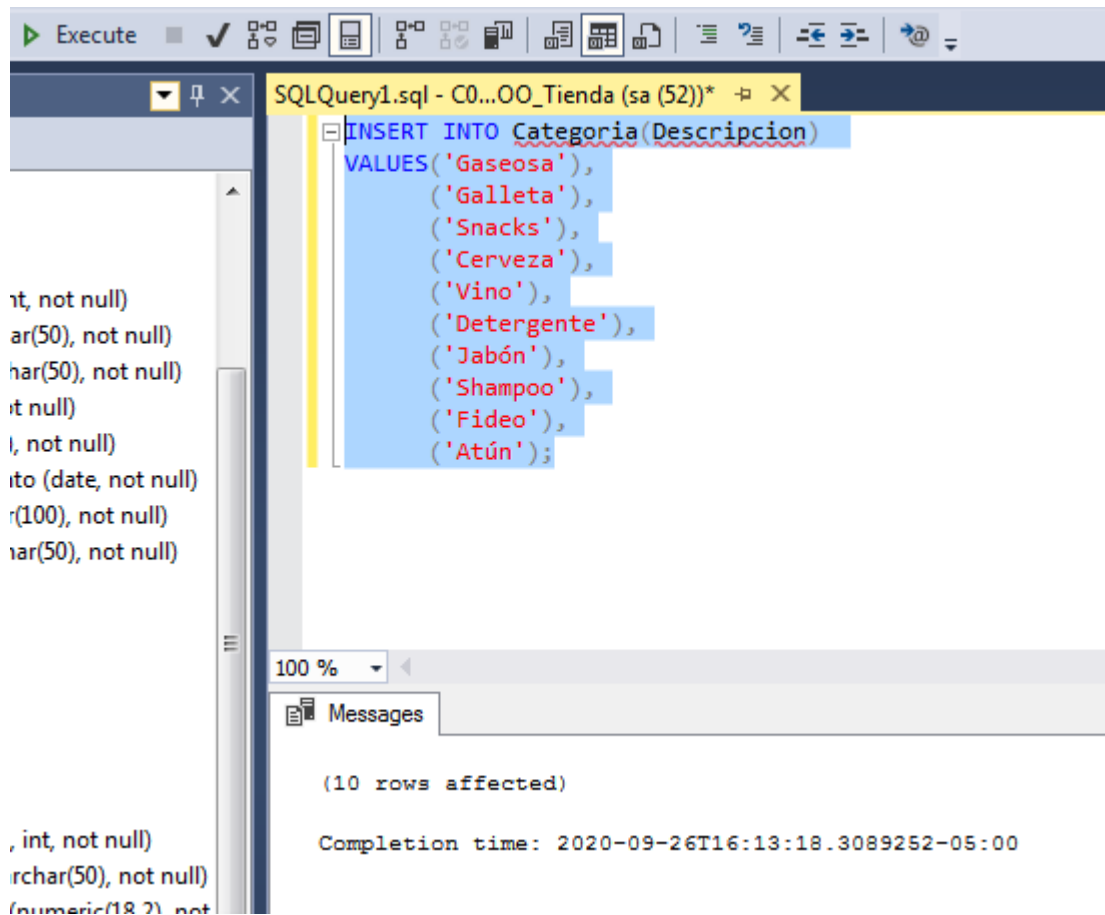


Tabla Producto:

```

--Ingresando Productos de la Categoría Gaseosa
INSERT INTO
Producto(Descripcion,PrecioCompra,PrecioVenta,Volumen,idCategoria)
VALUES('Coca Cola','2.00','2.50','500',1),
      ('Inca Kola','2.10','2.50','500',1),
      ('Fanta','1.50','2.00','500',1),
      ('Sprite','2.00','2.50','500',1);

--Ingresando Productos de la Categoría Galleta
INSERT INTO Producto(Descripcion,PrecioCompra,PrecioVenta,idCategoria)
VALUES('Casino','0.40','0.60',2),
      ('Oreo','0.50','0.70',2),
      ('Tentación','0.30','0.50',2),
      ('Integrackers','0.60','0.90',2);

--Ingresando Productos de la Categoría Snacks
INSERT INTO Producto(Descripcion,PrecioCompra,PrecioVenta,idCategoria)
VALUES('Doritos','0.70','1.00',3),
      ('Cheetos','0.60','1.00',3),
      ('Cuates','0.30','0.50',3),

```



```
('Papi Ricas', '0.30', '0.50', 3);
```

--Ingresando Productos de la Categoría Cerveza

INSERT INTO

Producto(Descripcion,PrecioCompra,PrecioVenta,Volumen,Peso,idCategoria)

```
VALUES('Cusqueña', '5.50', '6.50', '620', '0.90', 4),
      ('Cristal', '5.00', '5.50', '600', '0.92', 4),
      ('Pilsen', '5.00', '6.00', '620', '0.90', 4),
      ('Brahma', '4.20', '5.00', '650', '1', 4);
```

--Ingresando Productos de la Categoría Vino

INSERT INTO

Producto(Descripcion,PrecioCompra,PrecioVenta,Volumen,Peso,idCategoria)

```
VALUES('Tabernero', '12.50', '14.90', '920', '1.5', 5),
      ('Tacama', '18.50', '24.90', '920', '1.4', 5),
      ('Concha y Toro', '29.00', '3.00', '950', '1.7', 5),
      ('La moras', '52.50', '61.50', '900', '1.4', 5);
```

--Ingresando Productos de la Categoría Detergente

INSERT INTO Producto(Descripcion,PrecioCompra,PrecioVenta,Peso,idCategoria)

```
VALUES('Ace', '2.00', '2.50', '0.6', 6),
      ('Opal', '2.10', '2.50', '0.6', 6),
      ('Bolívar', '1.90', '2.30', '0.5', 6),
      ('Ariel', '2.30', '2.80', '0.7', 6);
```

--Ingresando Productos de la Categoría Jabón

INSERT INTO Producto(Descripcion,PrecioCompra,PrecioVenta,idCategoria)

```
VALUES('Neko', '1.40', '2.00', 7),
      ('Dove', '1.90', '2.50', 7),
      ('Heno de Pravia', '2.30', '3.20', 7),
      ('Moncler', '2.90', '3.50', 7);
```

--Ingresando Productos de la Categoría Shampoo

INSERT INTO

Producto(Descripcion,PrecioCompra,PrecioVenta,Volumen,idCategoria)

```
VALUES('Pantene', '12.00', '14.50', '400', 8),
      ('Ego', '14.00', '15.90', '500', 8),
      ('Sedal', '8.50', '10.50', '340', 8),
      ('H&S', '12.50', '14.9', '375', 8);
```

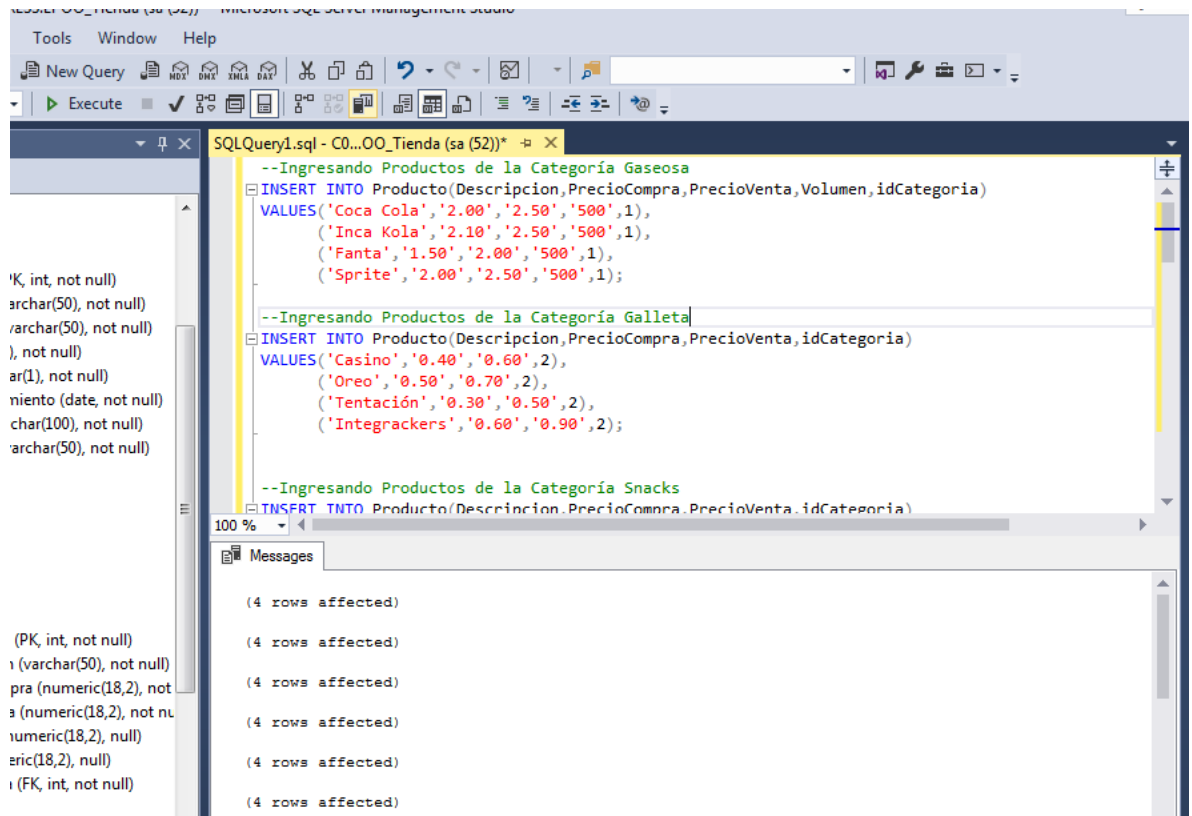
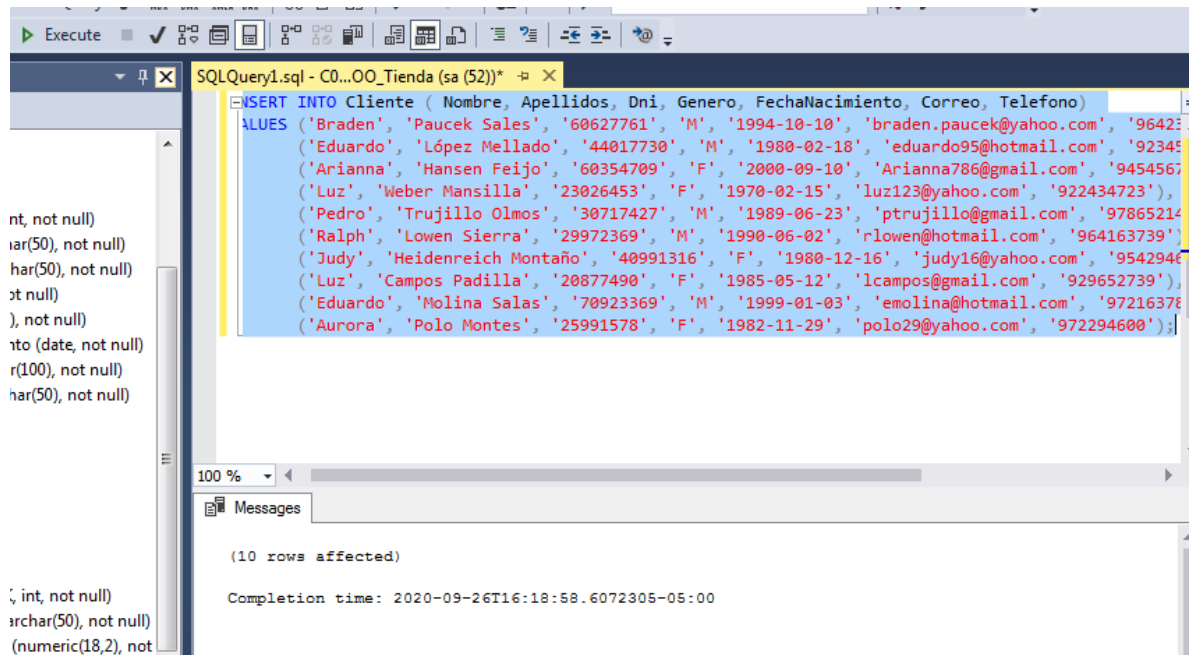


Tabla Cliente:

```
INSERT INTO Cliente ( Nombre, Apellidos, Dni, Genero, FechaNacimiento,
Correo, Telefono)
VALUES ('Braden', 'Paucek Sales', '60627761', 'M', '1994-10-10',
'braden.paucek@yahoo.com', '964234567'),
('Eduardo', 'López Mellado', '44017730', 'M', '1980-02-18',
'eduardo95@hotmail.com', '923456789'),
('Arianna', 'Hansen Feijo', '60354709', 'F', '2000-09-10',
'Arianna786@gmail.com', '945456723'),
('Luz', 'Weber Mansilla', '23026453', 'F', '1970-02-15',
'luz123@yahoo.com', '922434723'),
('Pedro', 'Trujillo Olmos', '30717427', 'M', '1989-06-23',
'ptrujillo@gmail.com', '978652145'),
('Ralph', 'Lowen Sierra', '29972369', 'M', '1990-06-02',
'rlowen@hotmail.com', '964163739'),
('Judy', 'Heidenreich Montaña', '40991316', 'F', '1980-12-16',
'judy16@yahoo.com', '954294619'),
('Luz', 'Campos Padilla', '20877490', 'F', '1985-05-12',
'lcampos@gmail.com', '929652739'),
('Eduardo', 'Molina Salas', '70923369', 'M', '1999-01-03',
'emolina@hotmail.com', '972163784'),
('Aurora', 'Polo Montes', '25991578', 'F', '1982-11-29',
'polo29@yahoo.com', '972294600');
```

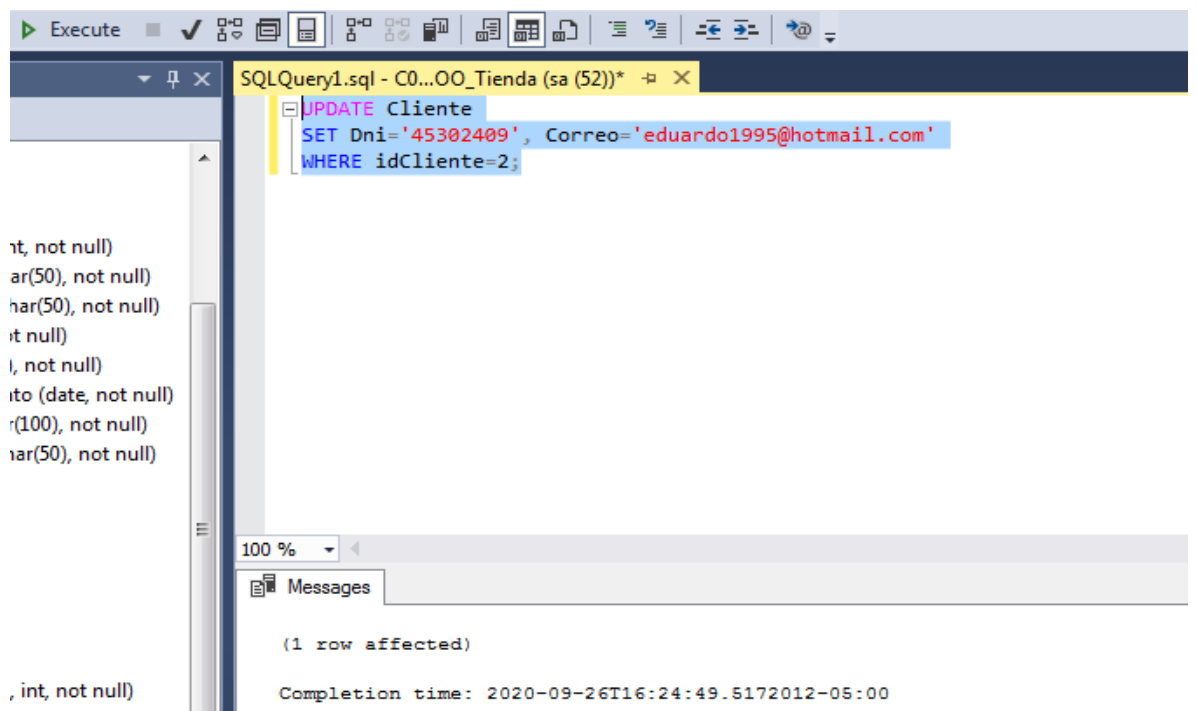


2.5. Actualización de Datos:

Ejecutamos el siguiente query para actualizar algunos datos de un cliente.

```

UPDATE Cliente
SET Dni='45302409', Correo='eduardo1995@hotmail.com'
WHERE idCliente=2;
  
```



int, not null)
char(50), not null)
char(50), not null)
not null)
1), not null)
into (date, not null)
ar(100), not null)
:char(50), not null)

K, int, not null)
rchar(50), not null)
s (numeric(18,2), not null)
numeric(18,2), not null)
neric(18,2), null)
:(18,2), null)
K, int, not null)

SQLQuery1.sql - C0...OO_Tienda (sa (52))

```
SELECT * FROM Producto;
```

100 %

Results Messages

	idProducto	Descripcion	PrecioCompra	PrecioVenta	Volumen	Peso	idCategoria
1	1	Coca Cola	2.00	2.50	500.00	0.00	1
2	2	Inca Kola	2.10	2.50	500.00	0.00	1
3	3	Fanta	1.50	2.00	500.00	0.00	1
4	4	Sprite	2.00	2.50	500.00	0.00	1
5	5	Casino	0.40	0.60	NULL	0.00	2
6	6	Oreo	0.50	0.70	NULL	0.00	2
7	7	Tentación	0.30	0.50	NULL	0.00	2
8	8	Integrackers	0.60	0.90	NULL	0.00	2
9	9	Doritos	0.70	1.00	NULL	0.00	3
10	10	Cheetos	0.60	1.00	NULL	0.00	3
11	11	Cuates	0.30	0.50	NULL	0.00	3

Query executed successfully. | C061030\SQLEXPRESS (11.0 RTM) | sa (52) | LPOO_Tienda | 00:00:00 | 32 rows

Seleccionar la descripción del producto y sus precios para aquellos que tienen un precio de venta mayor a 2.00

```
SELECT Descripcion, PrecioCompra, PrecioVenta
FROM Producto
WHERE PrecioVenta > 2.00;
```

SQLQuery1.sql - C0...OO_Tienda (sa (52))

```

SELECT Descripcion, PrecioCompra, PrecioVenta
FROM Producto
WHERE PrecioVenta > 2.00;

```

100 %

Results Messages

	Descripcion	PrecioCompra	PrecioVenta
1	Coca Cola	2.00	2.50
2	Inca Kola	2.10	2.50
3	Sprite	2.00	2.50
4	Cusqueña	5.50	6.50
5	Cristal	5.00	5.50
6	Pilsen	5.00	6.00
7	Brahma	4.20	5.00
8	Tabemero	12.50	14.90
9	Tacama	18.50	24.90
10	Concha y Toro	29.00	3.00
11	La moras	52.50	61.50

Query executed successfully. | C061030\SQLEXPRESS (11.0 RTM) | sa (52) | LPOO_Tienda | 00:00:00 | 22 rows

Seleccionar la descripción del producto y sus precios para aquellos que tienen un precio de venta mayor a 2.00 ordenados por su descripción de forma descendente.

```

SELECT Descripcion, PrecioCompra, PrecioVenta
FROM Producto
WHERE PrecioVenta > 2.00
ORDER BY Descripcion DESC;

```

SQLQuery1.sql - C0...OO_Tienda (sa (52))*

```

SELECT Descripcion, PrecioCompra, PrecioVenta
FROM Producto
WHERE PrecioVenta > 2.00
ORDER BY Descripcion DESC;

```

100 %

Results Messages

	Descripcion	PrecioCompra	PrecioVenta
1	Tacama	18.50	24.90
2	Tabemero	12.50	14.90
3	Sprite	2.00	2.50
4	Sedal	8.50	10.50
5	Pilsen	5.00	6.00
6	Pantene	12.00	14.50
7	Opal	2.10	2.50
8	Moncler	2.90	3.50
9	La moras	52.50	61.50
10	Inca Kola	2.10	2.50
11	Heno de Pravia	2.30	3.20

Query executed successfully. | C061030\SQLEXPRESS (11.0 RTM) | sa (52) | LPOO_Tienda | 00:00:00 | 22 rows

Seleccionar la descripción y precio de venta de los 3 productos más caros.

```

SELECT TOP 3 Descripcion, PrecioVenta
FROM Producto
ORDER BY PrecioVenta DESC;

```


SQLQuery1.sql - C0...OO_Tienda (sa (52))*

```

SELECT TOP 3 Descripcion, PrecioVenta
FROM Producto
ORDER BY PrecioVenta DESC;

```

100 %

Results Messages

	Descripcion	PrecioVenta
1	La moras	61.50
2	Tacama	24.90
3	Ego	15.90

Consultas Anidadas

Seleccionar todos los productos de la categoría Vino:

```

SELECT * FROM Producto
WHERE idCategoria = (SELECT idCategoria FROM Categoria
                     WHERE Descripcion='Vino');

```

SQLQuery1.sql - C0...OO_Tienda (sa (52))*

```

SELECT * FROM Producto
WHERE idCategoria = (SELECT idCategoria FROM Categoria
                     WHERE Descripcion='Vino');

```

100 %

Results Messages

	idProducto	Descripcion	PrecioCompra	PrecioVenta	Volumen	Peso	idCategoria
1	17	Tabernero	12.50	14.90	920.00	1.50	5
2	18	Tacama	18.50	24.90	920.00	1.40	5
3	19	Concha y Toro	29.00	3.00	950.00	1.70	5
4	20	La moras	52.50	61.50	900.00	1.40	5

Consultas a más de una tabla (JOINS)

Mostrar la descripción, precio venta y categoría de todos los productos.

```

SELECT P.Descripcion AS Producto, P.PrecioVenta AS Precio, C.Descripcion AS
Categoría
FROM Producto AS P
INNER JOIN Categoria AS C
ON P.idCategoria=C.idCategoria;

```

SQLQuery1.sql - C0...OO_Tienda (sa (52))

```

SELECT P.Descripcion AS Producto, P.PrecioVenta AS Precio, C.Descripcion AS Categoría
FROM Producto AS P
INNER JOIN Categoria AS C
ON P.idCategoria=C.idCategoria;

```

100 %

Results Messages

	Producto	Precio	Categoría
1	Coca Cola	2.50	Gaseosa
2	Inca Kola	2.50	Gaseosa
3	Fanta	2.00	Gaseosa
4	Sprite	2.50	Gaseosa
5	Casino	0.60	Galleta
6	Oreo	0.70	Galleta
7	Tentación	0.50	Galleta
8	Integrackers	0.90	Galleta
9	Doritos	1.00	Snacks
10	Cheetos	1.00	Snacks
11	Cuates	0.50	Snacks

Query executed successfully. | C061030\SQLEXPRESS (11.0 RTM) | sa (52) | LPOO_Tienda | 00:00:00 | 32 rows

La sentencia **GROUP BY**: Se utiliza para agrupar registros que contienen los mismos valores. Se puede usar las siguientes funciones de agrupamiento: COUNT(), SUM(), MIN(), MAX(), AVG().

La sentencia **HAVING** especifica una condición de búsqueda para un grupo o agregado.

Listar la cantidad de clientes por género

```

SELECT Genero, COUNT(idCliente) as Cantidad
FROM Cliente
GROUP BY Genero

```

SQLQuery1.sql - C0...OO_Tienda (sa (52))*

```

SELECT Genero, COUNT(idCliente) as Cantidad
FROM Cliente
GROUP BY Genero

```

100 %

Results Messages

	Genero	Cantidad
1	F	4
2	M	5

Listar el precio promedio por categorías de los productos. Omitir precios menores a 1 sol

```

SELECT c.Descripcion as 'Producto', AVG(p.PrecioVenta) as 'Precio Promedio'
FROM Producto as p
INNER JOIN Categoria as c ON p.idCategoria=c.idCategoria
GROUP BY c.Descripcion
HAVING AVG(p.PrecioVenta)>1
ORDER BY 2 DESC;

```

SQLQuery1.sql - C0...OO_Tienda (sa (52))*

```

SELECT c.Descripcion as 'Producto', AVG(p.PrecioVenta) as 'Precio Promedio'
FROM Producto as p
INNER JOIN Categoria as c ON p.idCategoria=c.idCategoria
GROUP BY c.Descripcion
HAVING AVG(p.PrecioVenta)>1
ORDER BY 2 DESC;

```

100 %

Results Messages

	Producto	Precio Promedio
1	Vino	26.075000
2	Shampoo	13.950000
3	Cerveza	5.750000
4	Jabón	2.800000
5	Detergente	2.525000
6	Gaseosa	2.375000

2.8. Procedimientos

Crear un procedimiento que permita buscar un cliente por DNI.

```

CREATE PROCEDURE usp_clienteXdni
@dni char(8)
AS
SELECT * FROM Cliente
WHERE Dni = @dni;

```

```
SQLQuery1.sql - C0...OO_Tienda (sa (52))* -p X
CREATE PROCEDURE usp_clienteXdni
@dni char(8)
AS
SELECT * FROM Cliente
WHERE Dni = @dni;
```

100 %

Messages

Commands completed successfully.

Completion time: 2020-10-02T11:36:59.6718464-05:00

Ejecución:

```
EXECUTE usp_clienteXdni '40991316';
```

```
SQLQuery1.sql - C0...OO_Tienda (sa (52))* -p X
EXECUTE usp_clienteXdni '40991316';
```

100 %

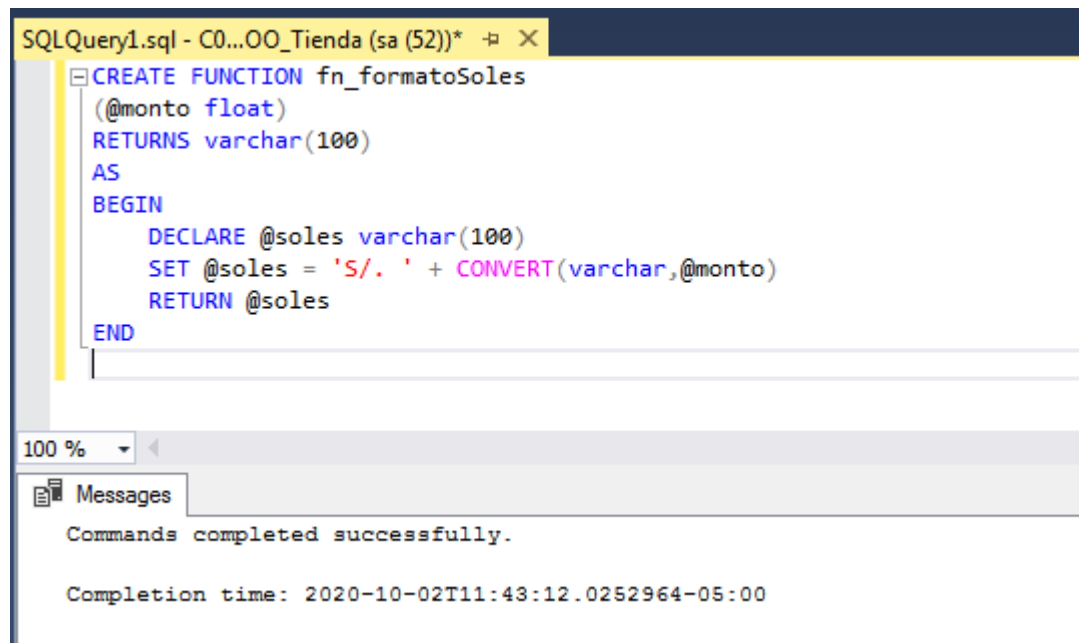
Results Messages

	idCliente	Nombre	Apellidos	Dni	Genero	FechaNacimiento	Correo	Telefono
1	7	Judy	Heidenreich Montaña	40991316	F	1980-12-16	judy16@yahoo.com	954294619

2.9. Funciones:

Crear una función que permita agregar el formato de soles.

```
CREATE FUNCTION fn_formatoSoles
(@monto float)
RETURNS varchar(100)
AS
BEGIN
    DECLARE @soles varchar(100)
    SET @soles = 'S/. ' + CONVERT(varchar,@monto)
    RETURN @soles
END
```



Uso de la función:

```
SELECT Descripcion, dbo.fn_formatoSoles(PrecioVenta) as Precio
FROM Producto;
```


SQLQuery1.sql - C0...OO_Tienda (sa (52))*

```
SELECT Descripcion, dbo.fn_formatoSoles(PrecioVenta) as Precio  
FROM Producto;
```

100 %

Results

Messages

	Descripcion	Precio
1	Coca Cola	S/. 2.5
2	Inca Kola	S/. 2.5
3	Fanta	S/. 2
4	Sprite	S/. 2.5
5	Casino	S/. 0.6
6	Oreo	S/. 0.7
7	Tentación	S/. 0.5
8	Integrackers	S/. 0.9
9	Doritos	S/. 1
10	Cheetos	S/. 1
11	Cuates	S/. 0.5
12	Papi Ricas	S/. 0.5
13	Cusqueña	S/. 6.5
14	Cristal	S/. 5.5
15	Pilsen	S/. 6
16	Brahma	S/. 5
17	Tabemero	S/. 1...
18	Tacama	S/. 2...
19	Concha y ...	S/. 3

Query executed successfully.

C061030\SQLEXPRESS (11.0 RTM)

sa (52)

LPOO_Tienda

00:00:00