Benemérita Universidad Autónoma de Puebla

Diseño de Bases de Datos

Reporte de Proyecto

Mtro. Roberto Elvira Enríquez

Alumnos: Kevin Eduardo González Valero 202039083

Vicente Zavaleta Sanchez 202062102

Fecha: 4 de mayo de 2023

Introducción

A continuación, se muestra la siguiente problemática a resolver, el cual deberá dar solución de una base de datos con ciertos parámetros que permitirán el almacenamiento de datos de clientes, productos, vendedores y distribuidores

Planteamiento del problema y especificación de requisitos de una base de datos

Dulcería “La Josefina” requiere una sistema gestor de datos que permita a los trabajadores llevar un control entre su inventario para ver la cantidad de dulces que llevan, así como su precio entre otras cosas, los proveedores que serán los que abastecen a la tienda, sus clientes que harán compras y dependiendo de su tarjeta de membresía tendrán un descuento especial en su compra, sus ventas donde se guardaran datos como el precio y artículos vendidos y los trabajadores para llevar a cabo el control de sus empleados.

Los requisitos de cada rubro son:

1. Producto: Numero de identificador del producto, nombre del producto, precio unitario e id del proveedor.
2. Proveedor: Numero de identificador del proveedor, nombre de la marca, teléfono y correo
3. Cliente: Numero de identificador del cliente, nombre, apellidos, teléfono, correo electrónico y tipo de membresía.
4. Venta: Numero de identificador de venta, su fecha en la que se realizó, id del cliente, id del producto, cantidad de productos, subtotal, descuento, total, identificador del trabajador que realizo la venta.
5. Vendedor: Numero de identificador de trabajador, nombre, apellidos, edad y teléfono

Así mismo, un sistema de membresías, que estarán divididas en 3 niveles:

1. ORO: Sera el nivel más alto donde tendrán descuentos del 10% en su compra total
2. PLATA: En este nivel el cliente tiene un 7% de descuento en su total de compra
3. BRONCE: Es el nivel básico el cual tiene como beneficios 5% de descuento

Diagrama entidad relación con cardinalidad

Diagrama

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Normalización

La normalización de una tabla tiene como objetivo eliminar la redundancia y garantizar la integridad de los datos, para lo cual se pueden aplicar una serie de reglas. En este caso, se puede aplicar la siguiente normalización:

A continuación, se mostrarán las tablas de nuestra base de datos:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLIENTE | PRODUCTO | PROVEEDOR | VENDEDOR | VENTA |
| -ID\_CLIENTE  -NOMBRE  -AP  -AM  -TELEFONO  -CORREO  -TIPO\_MEMB | -ID\_PRODUCTO  -NOMBRE  -PRECIO  -ID\_PROVEEDOR | -ID\_PROVEEDOR  -MARCA  -TELEFONO  -CORREO | -ID\_VENDEDOR  -NOMBRE  -AP  -AM  -EDAD  -TELEFONO | -ID\_VENTA  -FECHA  -ID\_CLIENTE  -ID\_PRODUCTO  -CANTIDAD  -SUBTOTAL  -DESCUENTO  -TOTAL  -ID\_TRABAJADOR |

La Primera forma normal (1FN) nos dice que:

* Una relación debe tener una clave primaria única y que cada atributo debe contener un solo valor.

En otras palabras, significa que no debe haber atributos múltiples o repetidos en una misma tabla. Por lo tanto, las tablas se encuentran en primera forma normal.

La Segunda forma normal (2FN) nos dice que:

* Debe cumplir con la 1FN.
* Los atributos no clave deben depender totalmente de la clave primaria.

Esto significa que si una tabla tiene varios atributos que dependen de la misma clave primaria, pero no todos ellos dependen de cada parte de la clave, entonces esos atributos deben dividirse en tablas separadas.

En este caso la tabla VENTA no estaría normalizada en Segunda Forma, por lo cual si aplicamos esta regla nuestras tablas quedarían de la siguiente manera:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CLIENTE | PRODUCTO | PROVEEDOR | VENDEDOR | VENTA | DETALLES |
| -ID\_CLIENTE  -NOMBRE  -AP  -AM  -TELEFONO  -CORREO  -TIPO\_MEMB | -ID\_PRODUCTO  -NOMBRE  -PRECIO  -ID\_PROVEEDOR | -ID\_PROVEEDOR  -MARCA  -TELEFONO  -CORREO | -ID\_VENDEDOR  -NOMBRE  -AP  -AM  -EDAD  -TELEFONO | -ID\_VENTA  -FECHA  -ID\_CLIENTE  -SUBTOTAL  -DESCUENTO  -TOTAL  -ID\_TRABAJADOR | -ID\_VENTA  -ID\_PRODUCTO  -CANTIDAD |

La Tercera forma normal (3FN) nos dice que:

* Debe cumplir con la 2FN.
* Los atributos no clave no deben depender de otros atributos no clave.

En otras palabras, esto significa que si una tabla tiene atributos no clave que dependen de otros atributos no clave, entonces esos atributos deben dividirse en tablas separadas.

Por lo tanto, ya estarían normalizadas en Tercera forma, así que nuestras tablas quedarían finalmente de esta manera:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CLIENTE | PRODUCTO | PROVEEDOR | VENDEDOR | VENTA | DETALLES |
| -ID\_CLIENTE  -NOMBRE  -AP  -AM  -TELEFONO  -CORREO  -TIPO\_MEMB | -ID\_PRODUCTO  -NOMBRE  -PRECIO  -ID\_PROVEEDOR | -ID\_PROVEEDOR  -MARCA  -TELEFONO  -CORREO | -ID\_VENDEDOR  -NOMBRE  -AP  -AM  -EDAD  -TELEFONO | -ID\_VENTA  -FECHA  -ID\_CLIENTE  -SUBTOTAL  -DESCUENTO  -TOTAL  -ID\_TRABAJADOR | -ID\_VENTA  -ID\_PRODUCTO  -CANTIDAD |

Y el código para la creación de estas sería el siguiente:

CREATE TABLE CLIENTE (

ID\_CLIENTE INT PRIMARY KEY,

NOMBRE VARCHAR(50),

AP VARCHAR(50),

AM VARCHAR(50),

TELEFONO VARCHAR(20),

CORREO VARCHAR(50),

TIPO\_MEMB VARCHAR(50)

);

CREATE TABLE VENDEDOR (

ID\_VENDEDOR INT PRIMARY KEY,

NOMBRE VARCHAR(50),

AP VARCHAR(50),

AM VARCHAR(50),

EDAD INT,

TELEFONO VARCHAR(20)

);

CREATE TABLE PROVEEDOR (

ID\_PROVEEDOR INT PRIMARY KEY,

MARCA VARCHAR(50),

TELEFONO VARCHAR(20),

CORREO VARCHAR(50)

);

CREATE TABLE VENTA (

ID\_VENTA INT PRIMARY KEY,

FECHA DATE,

ID\_CLIENTE INT,

SUBTOTAL DECIMAL(10, 2),

DESCUENTO DECIMAL(10, 2),

TOTAL DECIMAL(10, 2),

ID\_VENDEDOR INT,

FOREIGN KEY (ID\_CLIENTE) REFERENCES CLIENTE(ID\_CLIENTE),

FOREIGN KEY (ID\_VENDEDOR) REFERENCES VENDEDOR(ID\_VENDEDOR)

);

CREATE TABLE PRODUCTO (

ID\_PRODUCTO INT PRIMARY KEY,

NOMBRE VARCHAR(50),

PRECIO DECIMAL(10, 2),

ID\_PROVEEDOR INT,

FOREIGN KEY (ID\_PROVEEDOR) REFERENCES PROVEEDOR(ID\_PROVEEDOR)

);

CREATE TABLE DETALLES (

ID\_VENTA INT,

ID\_PRODUCTO INT,

CANTIDAD INT,

PRIMARY KEY (ID\_VENTA, ID\_PRODUCTO),

FOREIGN KEY (ID\_VENTA) REFERENCES VENTA(ID\_VENTA),

FOREIGN KEY (ID\_PRODUCTO) REFERENCES PRODUCTO(ID\_PRODUCTO)

);

Finalmente, nuestras tablas ya estarían normalizadas en Primera, Segunda y Tercera forma ya que:

* Las tablas en SQL Server presentan una estructura que cumple con la tercera forma normal (3FN).
* Las tablas CLIENTE, PRODUCTO, PROVEEDOR y VENDEDOR tienen una clave primaria y todas sus columnas correspondientes dependen directamente de ellas, además de que cada fila representa un registro único y no hay redundancia de datos.
* La tabla VENTA tiene una clave primaria, todas sus columnas dependen directamente de ella, además ID\_CLIENTE y ID\_VENDEDOR son claves foráneas que hacen referencia a sus respectivas tablas, por último, cada fila de estas representa un registro único y no hay redundancia de datos.
* La tabla DETALLES tiene dos claves foráneas (ID\_VENTA y ID\_PRODUCTO) que hacen referencia a las tablas VENTA y PRODUCTO, además de que todas las columnas dependen directamente de la combinación de estas dos claves, cada fila representa una relación entre un producto vendido y una venta especifica y no hay redundancia de datos.

Ejecución del programa

1.- Creación de base de datos y de las tablas

La construcción de tablas se realizó en base en la tercera forma normal de normalización para hacerla más eficiente y menos compleja

Create DataBase Dulceria\_Josefina;

CREATE TABLE CLIENTE (

ID\_CLIENTE INT PRIMARY KEY,

NOMBRE VARCHAR(50),

AP VARCHAR(50),

AM VARCHAR(50),

TELEFONO VARCHAR(20),

CORREO VARCHAR(50),

TIPO\_MEMB VARCHAR(50)

);

CREATE TABLE VENDEDOR (

ID\_VENDEDOR INT PRIMARY KEY,

NOMBRE VARCHAR(50),

AP VARCHAR(50),

AM VARCHAR(50),

EDAD INT,

TELEFONO VARCHAR(20)

);

CREATE TABLE PROVEEDOR (

ID\_PROVEEDOR INT PRIMARY KEY,

MARCA VARCHAR(50),

TELEFONO VARCHAR(20),

CORREO VARCHAR(50)

);

CREATE TABLE VENTA (

ID\_VENTA INT PRIMARY KEY,

FECHA DATE,

ID\_CLIENTE INT,

SUBTOTAL DECIMAL(10, 2),

DESCUENTO DECIMAL(10, 2),

TOTAL DECIMAL(10, 2),

ID\_VENDEDOR INT,

FOREIGN KEY (ID\_CLIENTE) REFERENCES CLIENTE(ID\_CLIENTE),

FOREIGN KEY (ID\_VENDEDOR) REFERENCES VENDEDOR(ID\_VENDEDOR)

);

CREATE TABLE PRODUCTO (

ID\_PRODUCTO INT PRIMARY KEY,

NOMBRE VARCHAR(50),

PRECIO DECIMAL(10, 2),

ID\_PROVEEDOR INT,

FOREIGN KEY (ID\_PROVEEDOR) REFERENCES PROVEEDOR(ID\_PROVEEDOR)

);

CREATE TABLE DETALLES (

ID\_VENTA INT,

ID\_PRODUCTO INT,

CANTIDAD INT,

PRIMARY KEY (ID\_VENTA, ID\_PRODUCTO),

FOREIGN KEY (ID\_VENTA) REFERENCES VENTA(ID\_VENTA),

FOREIGN KEY (ID\_PRODUCTO) REFERENCES PRODUCTO(ID\_PRODUCTO)

);

2.- Inserción de datos a las tablas

Se registraron algunos campos que la dulcería ya tenía registrados antes de la implementación de la base de datos

INSERT INTO CLIENTE (ID\_CLIENTE, NOMBRE, AP, AM, TELEFONO, CORREO, TIPO\_MEMB) VALUES

(1, 'Ana', 'García', 'Pérez', '555-1234', 'ana.garcia@example.com', 'ORO'),

(2, 'Juan', 'López', 'García', '555-5678', 'juan.lopez@example.com', 'PLATA'),

(3, 'María', 'Hernández', 'Sánchez', '555-9012', 'maria.hernandez@example.com', 'BRONCE'),

(4, 'Pedro', 'Martínez', 'Ruiz', '555-3456', 'pedro.martinez@example.com', 'ORO'),

(5, 'Laura', 'González', 'Gutiérrez', '555-7890', 'laura.gonzalez@example.com', 'PLATA');

INSERT INTO PROVEEDOR (ID\_PROVEEDOR, MARCA, TELEFONO, CORREO) VALUES

(1, 'Dulces S.A.', '555-1111', 'dulces@example.com'),

(2, 'Postres deliciosos', '555-2222', 'postres@example.com'),

(3, 'Sabores únicos', '555-3333', 'sabores@example.com'),

(4, 'Golosinas felices', '555-4444', 'golosinas@example.com'),

(5, 'Dulces sueños', '555-5555', 'suenos@example.com');

INSERT INTO VENDEDOR (ID\_VENDEDOR, NOMBRE, AP, AM, EDAD, TELEFONO) VALUES

(1, 'Carlos', 'Sánchez', 'Pérez', 28, '555-1234'),

(2, 'Mónica', 'García', 'López', 35, '555-5678'),

(3, 'Luis', 'Hernández', 'Gutiérrez', 42, '555-9012'),

(4, 'Paula', 'Martínez', 'Sánchez', 23, '555-3456'),

(5, 'José', 'González', 'Ruiz', 29, '555-7890');

INSERT INTO PRODUCTO (ID\_PRODUCTO, NOMBRE, PRECIO, ID\_PROVEEDOR) VALUES

(1, 'Paletas de fresa', 10.50, 1),

(2, 'Gomitas de ositos', 5.25, 1),

(3, 'Chicles de menta', 2.99, 2),

(4, 'Bombones de chocolate', 15.75, 3),

(5, 'Caramelos de caramelo', 3.50, 4),

(6, 'Chocolate con leche', 12.99, 3),

(7, 'Caramelos de goma ácidos', 4.50, 2),

(8, 'Barra de chocolate negro', 8.75, 5),

(9, 'Caramelos de cereza', 3.99, 4),

(10, 'Galletas de mantequilla', 7.25, 5);

3.- Funciones

-Función DESCUENTO:

Esta función llamada “DESCUENTO” se encarga de otorgar el descuento según el tipo de membresía del cliente:

CREATE FUNCTION DBO.DESCUENTO (@TIPO\_MEMB VARCHAR(50))

RETURNS DECIMAL(10,2)

AS

BEGIN

DECLARE @descuento DECIMAL(10,2)

SET @descuento =

CASE @TIPO\_MEMB

WHEN 'ORO' THEN 0.10

WHEN 'PLATA' THEN 0.07

WHEN 'BRONCE' THEN 0.05

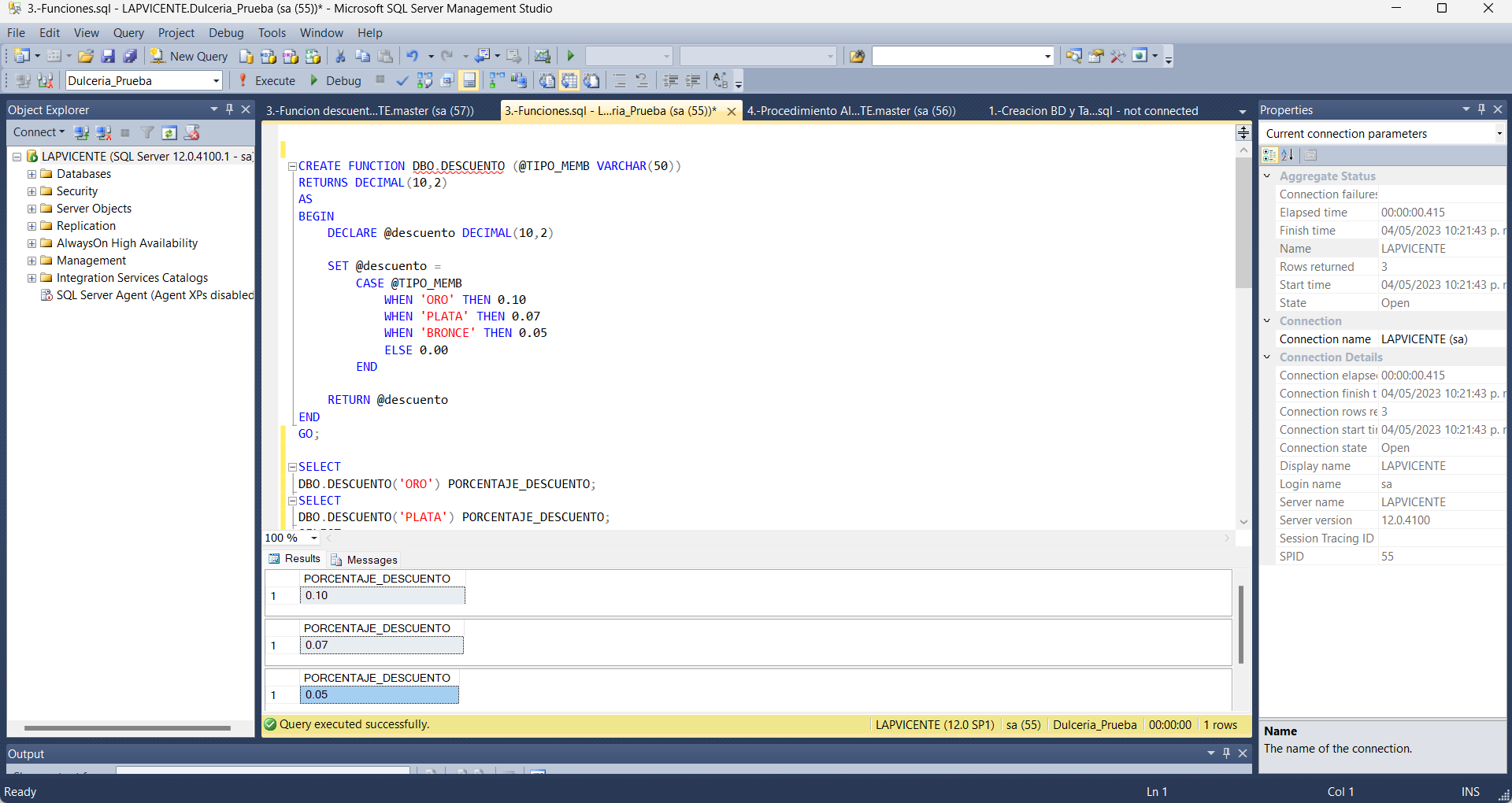
ELSE 0.00

END

RETURN @descuento

END

GO



-Función CAL\_SUBTOTAL:

Esta otra función llamada “CAL\_SUBTOTAL” calcula el subtotal de una venta:

CREATE FUNCTION DBO.CAL\_SUBTOTAL(@ID\_VENTA INT)

RETURNS DECIMAL(10, 2)

AS

BEGIN

DECLARE @COSTO DECIMAL(10, 2)

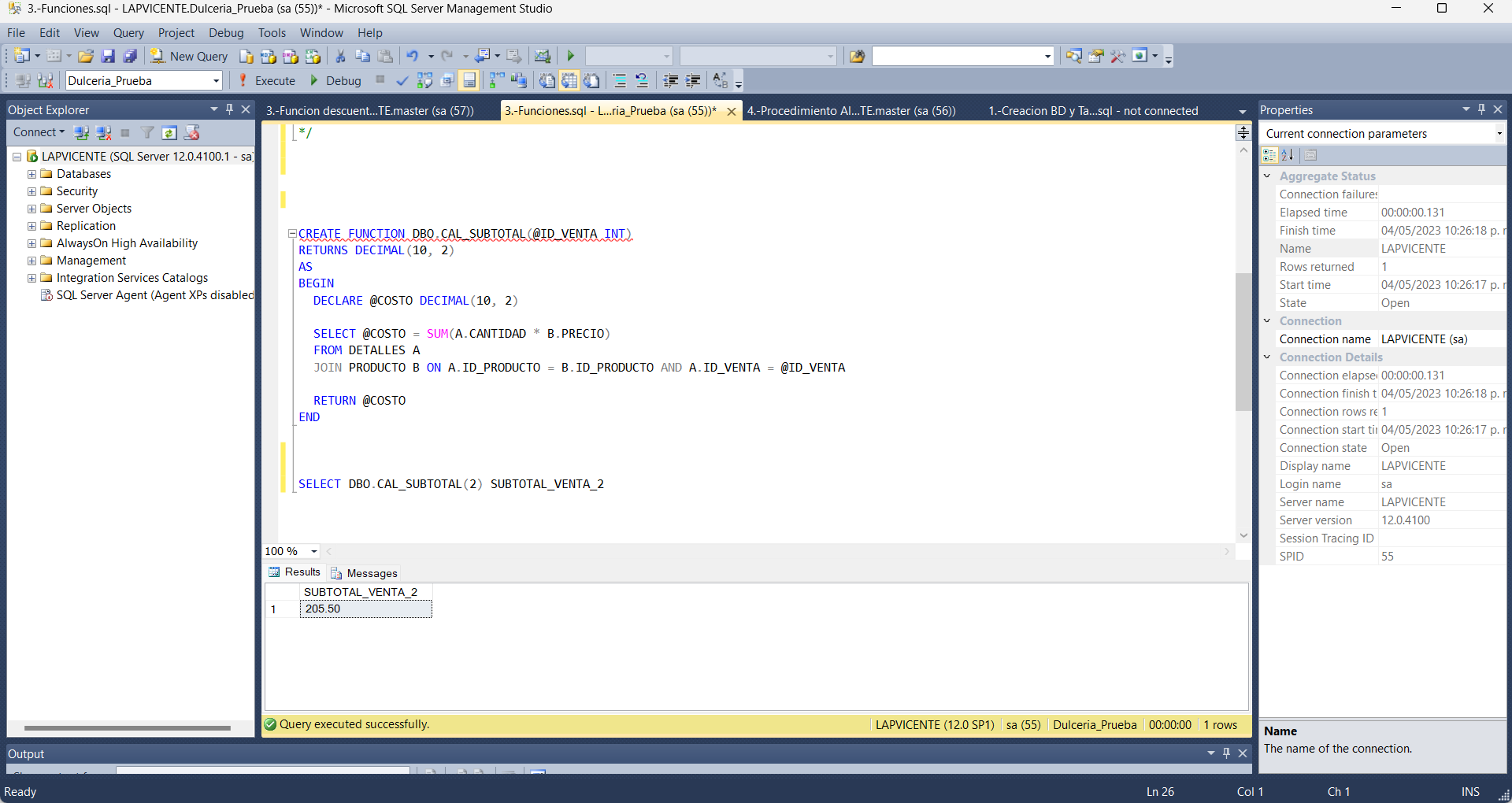
SELECT @COSTO = SUM(A.CANTIDAD \* B.PRECIO)

FROM DETALLES A

JOIN PRODUCTO B ON A.ID\_PRODUCTO = B.ID\_PRODUCTO AND A.ID\_VENTA = @ID\_VENTA

RETURN @COSTO

END;



4.- Procedimientos almacenados

-Procedimiento Almacenado ACT\_VENTA:

Este procedimiento almacenado llamado “ACT\_VENTA” se desarrolló para que actualice los detalles de los registros de la tabla de “VENTA” en base a los registros de la tabla “DETALLES” que estén relacionados al ID de la venta haciendo uso de las funciones anteriores:

CREATE PROCEDURE ACT\_VENTA @ID\_VENTA INT

AS

BEGIN

UPDATE VENTA

SET SUBTOTAL = DBO.CAL\_SUBTOTAL(DBO.VENTA.ID\_VENTA)

WHERE ID\_VENTA = @ID\_VENTA;

DECLARE @DESCUENTO DECIMAL(10, 2)

SELECT @DESCUENTO = DBO.DESCUENTO(B.TIPO\_MEMB)\*DBO.CAL\_SUBTOTAL(A.ID\_VENTA)

FROM VENTA A

JOIN CLIENTE B ON A.ID\_CLIENTE = B.ID\_CLIENTE WHERE ID\_VENTA=@ID\_VENTA;

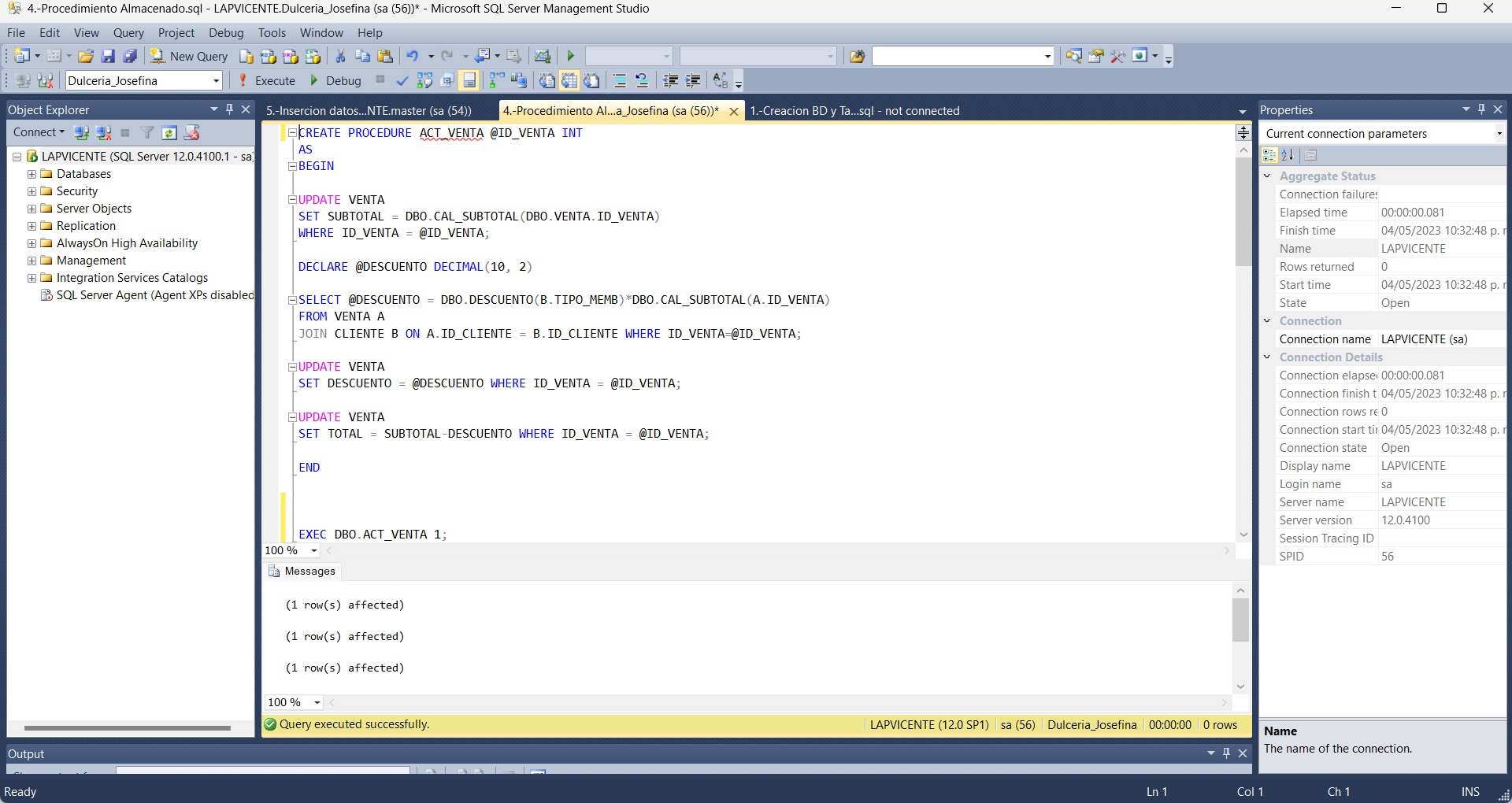
UPDATE VENTA

SET DESCUENTO = @DESCUENTO WHERE ID\_VENTA = @ID\_VENTA;

UPDATE VENTA

SET TOTAL = SUBTOTAL-DESCUENTO WHERE ID\_VENTA = @ID\_VENTA;

END;



-Procedimiento Almacenado CREAR\_VTA:

Este otro procedimiento almacenado llamado “CREAR\_VTA” permite agregar un registro para la tabla “VENTA” de una manera más sencilla y eficiente:

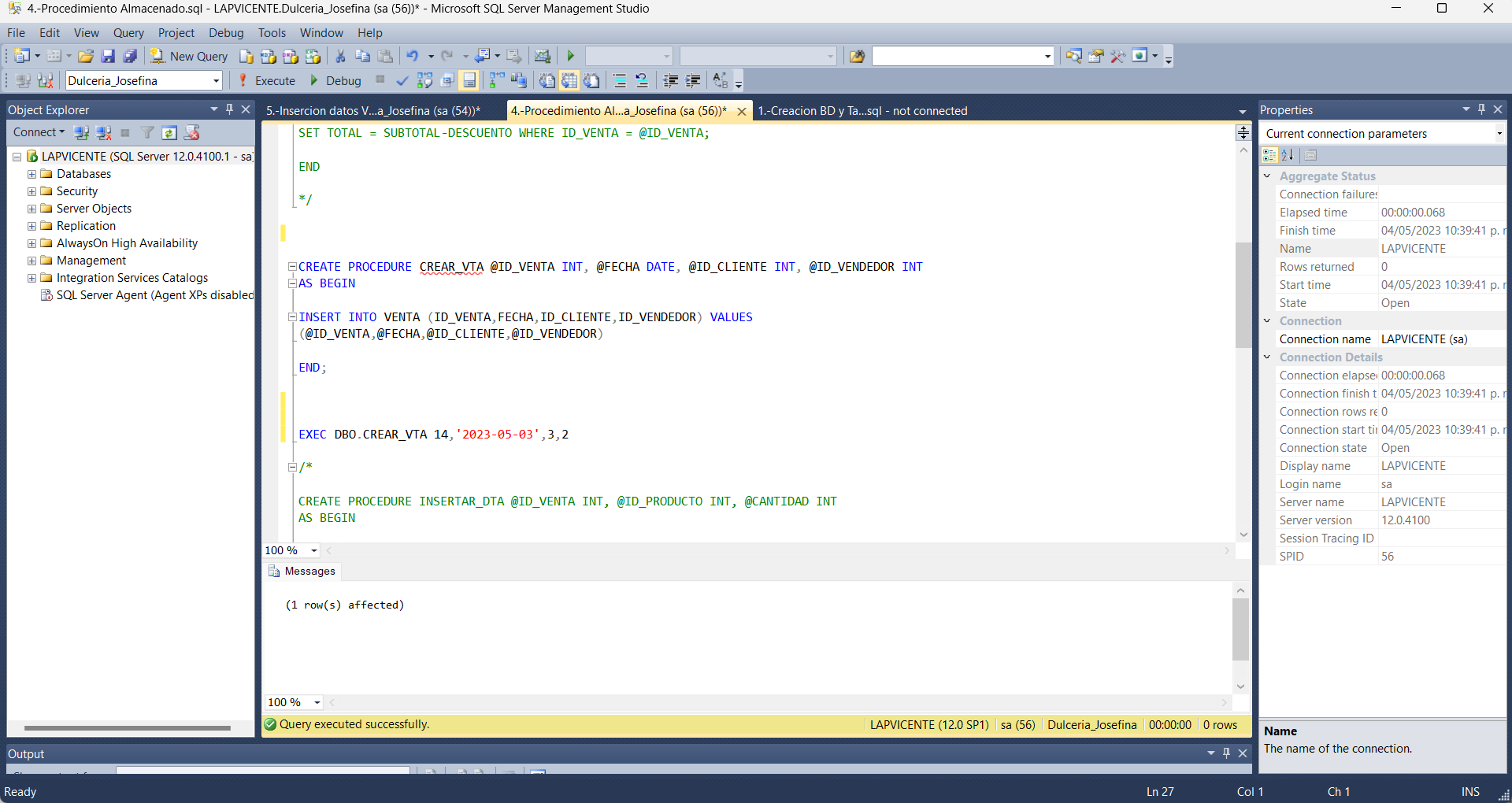
CREATE PROCEDURE CREAR\_VTA @ID\_VENTA INT, @FECHA DATE, @ID\_CLIENTE INT, @ID\_VENDEDOR INT

AS BEGIN

INSERT INTO VENTA (ID\_VENTA,FECHA,ID\_CLIENTE,ID\_VENDEDOR) VALUES

(@ID\_VENTA,@FECHA,@ID\_CLIENTE,@ID\_VENDEDOR)

END;



-Procedimiento Almacenado INSERTAR\_DTA:

Este otro procedimiento almacenado llamado “INSERTAR\_DTA” permite agregar un registro en la tabla “DETALLES” que este relaciona a alguna venta, además de que actualiza los datos de la tabla “VENTA” haciendo uso del procedimiento almacenado llamado “ACT\_VENTA”, por lo tanto, no solo facilita el agregar un registro, sino que actualiza y hace que los registros tengan coherencia.

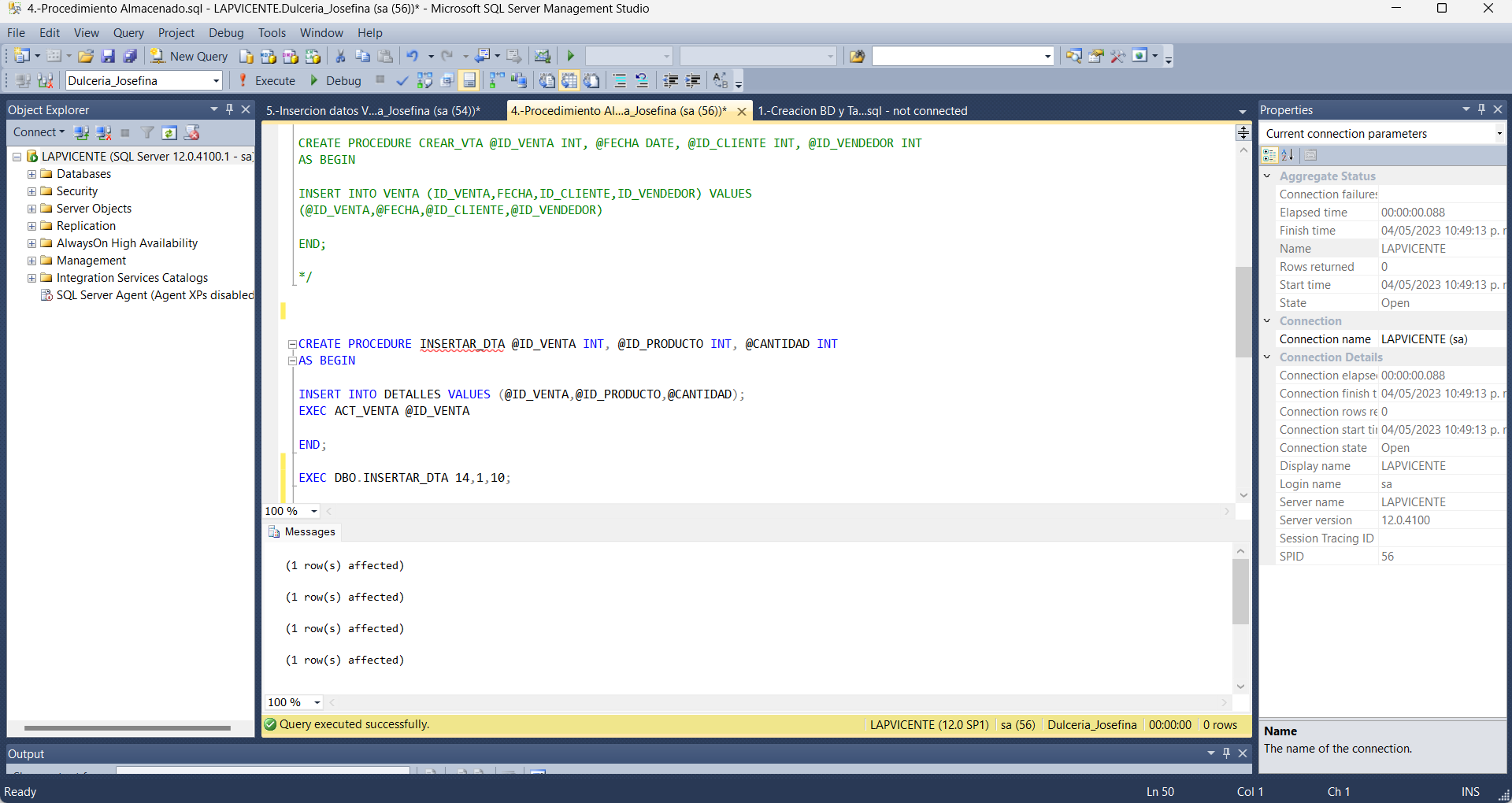
CREATE PROCEDURE INSERTAR\_DTA @ID\_VENTA INT, @ID\_PRODUCTO INT, @CANTIDAD INT

AS BEGIN

INSERT INTO DETALLES VALUES (@ID\_VENTA,@ID\_PRODUCTO,@CANTIDAD);

EXEC ACT\_VENTA @ID\_VENTA

END;



5.- Vistas

Esta vista se hizo para mostrar los productos que un cliente compro en cierto día en especifico

CREATE VIEW DETALLE\_GEN AS

SELECT v.id\_venta, c.id\_cliente, c.nombre, m.TIPO\_MEMB, v.fecha, v.subtotal, v.total, t.id\_vendedor

FROM venta v

JOIN CLIENTE c ON v.id\_cliente = c.id\_cliente

JOIN CLIENTE m ON c.TIPO\_MEMB = m.TIPO\_MEMB

JOIN VENDEDOR t ON v.id\_vendedor = t.id\_vendedor

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

6.- índices

CREATE INDEX Index\_Clientes ON CLIENTE (TIPO\_MEMB);

SELECT ID\_CLIENTE, NOMBRE, AP, TELEFONO, TIPO\_MEMB, DBO.DESCUENTO(TIPO\_MEMB) AS DESCUENTO

FROM CLIENTE;

Este índice se desarrolló para los casos en que el empleado o el mismo cliente no sepa el descuento que tiene por membresía, se le mostrara sus datos y su descuento

Texto

Descripción generada automáticamente

Conclusión

Para finalizar, se puede decir que se ha presentado una solución a la problemática de la dulcería "La Josefina", en la que se ha diseñado una base de datos con tablas normalizadas en tercera forma normal.

Se han especificado los requisitos para cada rubro, que incluyen inventario, proveedores, clientes, ventas y trabajadores. También se ha definido un sistema de membresías con tres niveles de descuento.

El programa creado para la base de datos incluye la creación de las tablas y las restricciones de integridad referencial necesarias para asegurar la consistencia de los datos.

En general, se ha buscado una solución eficiente y menos compleja para el manejo de datos de la dulcería.

En resumen, la creación de esta base de datos permitirá a la dulcería llevar un mejor control de sus inventarios, proveedores, ventas y clientes, lo que a su vez puede contribuir a una gestión más eficiente de su negocio y, en última instancia, mejorar su rentabilidad.