Gestión de Datos

T.P.: Venta de Electromésticos

1° Cuatrimestre – Año: 2011

Grupo: DOTNETPY
Axel Navarro
Nicolás Far

Contenido

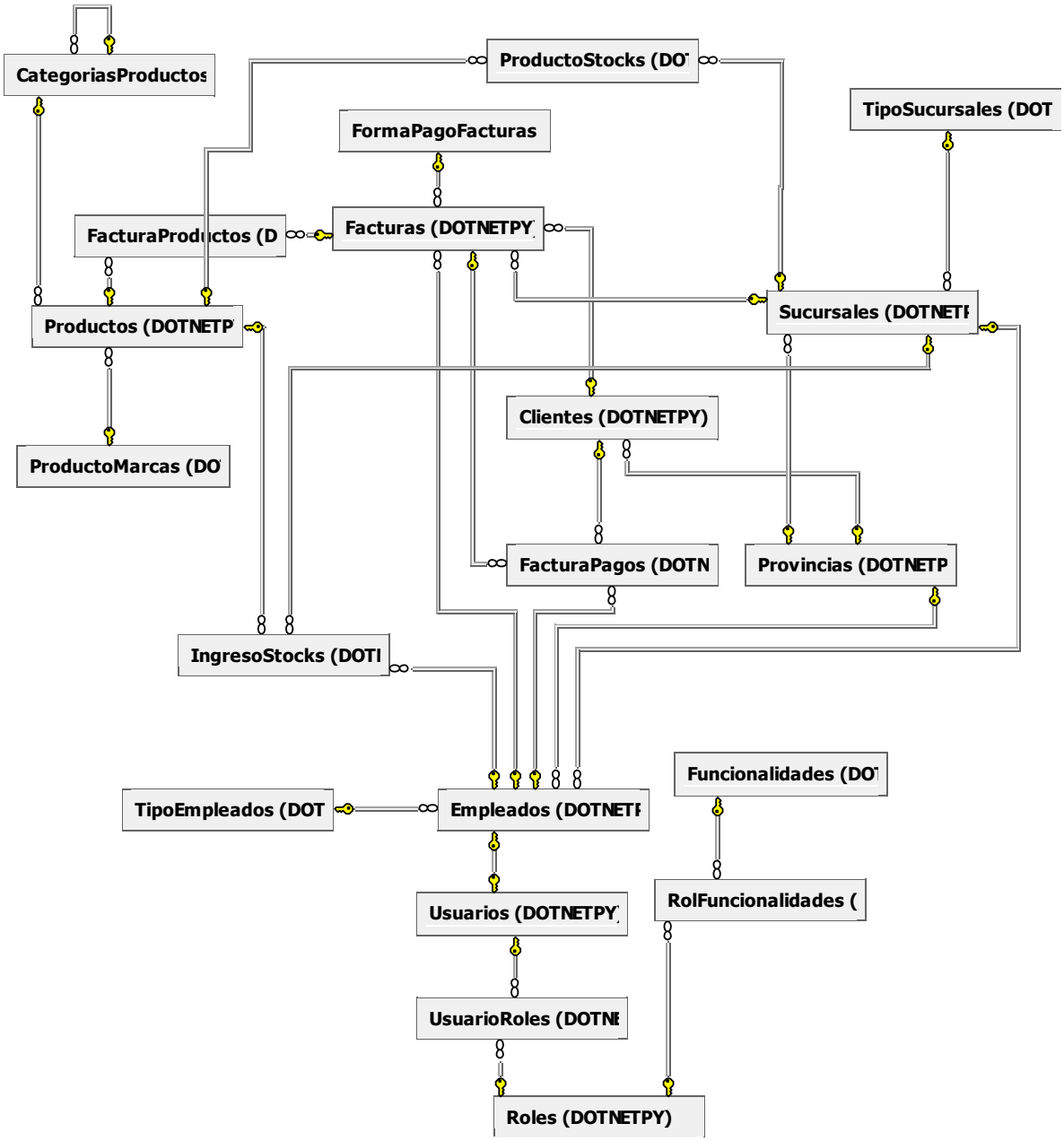
Introducción

La solución adjunta a este documento es el trabajo práctico realizado por el grupo DOTNETPY a lo solicitado en el primer cuatrimestre de 2011 por la cátedra de Gestión de Datos de la Universidad Tecnológica Nacional Facultad Regional Buenos Aires.

Se desarrolló el trabajo práctico solicitado en base a las especificaciones solicitadas en el archivo Enunciado.PDF versión 1.1. Se tuvieron en cuenta las normas de seguridad requeridas para la aplicación y se desarrolló cada una de las funcionalidades pedidas respetando la arquitectura en tres capas (ver diagrama de arquitectura).

Modelo de Datos

DER



Entidades

- TipoSucursales
  - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
  - Descripcion nvarchar(15) NOT NULL
- Sucursales
  - Id int [PRIMARY KEY IDENTITY (1, 1)] NOT NULL
  - TipoSucursalId int NOT NULL

    - Direccion nvarchar(100) NOT NULL
    - ProvinciaId int NOT NULL
    - Telefono nvarchar(20) NOT NULL
- TipoEmpleados
    - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
    - Descripcion nvarchar(15) NOT NULL
- Clientes
    - Dni int[PRIMARY KEY] NOT NULL
    - Nombre nvarchar(40) NOT NULL
    - Apellido nvarchar(40) NOT NULL
    - Mail nvarchar(50) NOT NULL
    - Telefono nvarchar(40) NOT NULL
    - Direccion nvarchar(100) NOT NULL
    - ProvinciaId int NOT NULL [FOREIGN KEY DOTNETPY.Provincias (id)]
    - Activo bit NOT NULL DEFAULT(1)
- Empleados
    - Dni int[PRIMARY KEY] NOT NULL
    - Nombre nvarchar(40) NOT NULL
    - Apellido nvarchar(40) NOT NULL
    - Mail nvarchar(50) NOT NULL
    - Direccion nvarchar(100) NOT NULL
    - Telefono nvarchar(40) NOT NULL
    - TipoEmpleadoId int [FOREIGN KEY DOTNETPY.TipoEmpleados (Id)]
    - ProvinciaId int NOT NULL [FOREIGN KEY DOTNETPY.Provincias (Id)]
    - SucursalId int NOT NULL [FOREING KEY DOTNETPY.SucursalId (Id)]
    - Activo bit NOT NULL DEFAULT(1)
- Usuarios
    - DniEmpleado int [PRIMARY KEY] NOT NULL [FOREIGN KEY DOTNETPY.Empleados(Dni)]
    - Username nvarchar(50) NOT NULL [INDEX UNIQUE]
    - Password nvarchar(100) NOT NULL
    - Intentos int NOT NULL DEFAULT(0)
    - Activo bit NOT NULL DEFAULT(1)
- Funcionalidades
    - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
    - Descripcion nvarchar(100) NOT NULL
- Roles
    - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
    - Nombre nvarchar(100) NOT NULL
    - Activo bit NOT NULL DEFAULT(1)
- RolFuncionalides

    - RolId int NOT NULL [FOREING KEY DOTNETPY.Roles(Id)]
    - FuncionalidadId int NOT NULL[FOREING KEY DOTNETPY.Funcionalidades(Id)]
- UsuarioRoles
    - DniEmpleado int [PRIMARY KEY] NOT NULL [FOREING KEY DOTNETPY.Usuarios(DniEmpleado)]
    - RolId int NOT NULL [FOREIGN KEY DOTNETPY.Roles(id)]
- ProductoMarcas
    - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
    - Descripcion nvarchar(100) NOT NULL
- Productos
    - Id int [PRIMARY KEY IDENTITY(1248681717, 1)] NOT NULL
    - Nombre nvarchar(100) NOT NULL
    - Descripcion nvarchar(100) NOT NULL
    - Precio float NOT NULL DEFAULT(0),
    - MarcaId int NOT NULL,
    - CategoriaId int NOT NULL [FOREIGN KEY DOTNETPY.CategoriasProductos(id)]
    - Activo bit NOT NULL DEFAULT(1)
- ProductoStocks
    - ProductoId int [PRIMARY KEY] NOT NULL
    - SucursalId int [PRIMARY KEY] NOT NULL
    - Cantidad int NOT NULL
- CategoriasProductos
    - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
    - Nombre nvarchar(100) NOT NULL
    - PadreId int NULL [FOREIGN KEY DOTNETPY.CategoriasProductos(id)]
- Facturas
    - Nro int NOT NULL
    - Fecha datetime NOT NULL
    - DniCliente int NOT NULL [FOREIGN KEY DOTNETPY.Clientes(Dni)]
    - DniEmpleado int NOT NULL [FOREIGN KEY DOTNETPY.Empleados(Dni)]
    - SucursalId int NOT NULL [FOREIGN KEY DOTNETPY.SucursalId(Id)]
        - INDEX NONCLUSTERED SucursalId INCLUDE (Nro, Fecha, DniCliente, Total)
    - FormaPagoId int NOT NULL [FOREIGN KEY DOTNETPY.FormaPagoFacturas(Id)]
    - Cuotas int NOT NULL
    - Subtotal float NOT NULL
    - Descuento float NOT NULL
    - Total float NOT NULL
- FacturaProductos
    - FacturaNro int [PRIMARY KEY] NOT NULL [FOREIGN KEY DOTNETPY.Facturas(Nro)]
    - ProductoId int [PRIMARY KEY] NOT NULL [FOREIGN KEY DOTNETPY.Productos(Id)]
    - Cantidad int NOT NULL

    - PrecioUnitario float NULL
- FacturaPagos
    - Id int IDENTITY(1,1) NOT NULL
    - FacturaNro int NOT NULL [FOREIGN KEY DOTNETPY.Facturas(Nro)]
    - DniCliente int NOT NULL [FOREIGN KEY DOTNETPY.Clientes(Dni)]
    - DniEmpleado int NOT NULL [FOREIGN KEY DOTNETPY.Empleados(Dni)]
    - Fecha datetime NOT NULL
    - Cuotas int NOT NULL
    - Monto float NOT NULL
- FormaPagoFacturas
    - Id [PRIMARY KEY] int NOT NULL
    - Descripcion nvarchar(15) NOT NULL
- IngresoStocks
    - Id int [PRIMARY KEY IDENTITY(1, 1)] NOT NULL
    - FechaIngreso datetime NOT NULL
    - SucursalId int NOT NULL [FOREIGN KEY DOTNETPY.Sucursales(id)]
    - ProductoId int NOT NULL [FOREIGN KEY DOTNETPY.Productos(id)]
    - EmpleadoId int NOT NULL [FOREIGN KEY DOTNETPY.Empleados(id)]
    - Cantidad int NOT NULL

Vistas

VentaProductos:

Muestra la información de la venta de un producto:

- Su Id y Categoría
- A qué precio y cuantos se vendieron
- Cuando
- En que sucursal
- Por cual empleado

Constraints

- TipoSucursales
  - Id
    - Clave autoincremental, identifica únicamente a los tipos de sucursal
- Sucursales
  - Id
    - Clave autoincremental, identifica únicamente a las sucursales
- TipoEmpleados
  - Id
    - Clave autoincremental, identifica únicamente a los tipos de empleado
- Clientes
  - Dni
    - Clave que identifica únicamente a los clientes.
  - ProvinciaId
    - Clave foránea que relaciona al cliente con su provincia de origen.
- Empleados
  - Dni
    - Clave que identifica únicamente a los empleados.
  - TipoEmpleadoId
    - Clave foránea que relaciona al empleado con su rol en la empresa (vendedor o analista)
  - ProvinciaId
    - Clave foránea que relaciona el empleado con su provincia de origen.
  - SucursalId
    - Clave foránea que relaciona al empleado con la sucursal donde trabaja
- Usuarios
  - DniEmpleado
    - Clave primaria que identifica únicamente a los usuarios del sistema y los relaciona con el empleado al cual está asignado, de modo que cada empleado tenga un solo usuario
  - Username [INDEX UNIQUE]
    - Índice único, para evitar que se dupliquen los nombres de usuarios
- Funcionalidades
  - Id
    - Clave que identifica únicamente a las funcionalidades del sistema
- Roles
  - Id
    - Clave que identifica únicamente a los roles de usuarios del sistema.
- RolFuncionalides
  - RolId y FuncionalidadId

        - Son clave primaria y clave foráneas a las tablas de Roles y Funcionalidades permiten hacer la relación de muchos a muchos entre los roles y las Funcionalidades.

- UsuarioRoles
    - DniEmpleado y RolId
        - Son clave primaria y clave foráneas a las tablas de Usuarios y Roles permiten hacer la relación de muchos a muchos entre los roles y usuarios del sistema.
- ProductoMarcas
    - Id
        - Clave autoincremental que identifica únicamente a las marcas de productos.
- Productos
    - Id
        - Clave autoincremental que identifica únicamente a los productos
    - CategoriaId
        - Clave foránea que relaciona el producto con la categoría a la que pertenece
- ProductoStocks
    - ProductoId y SucursalId
        - Son clave primaria ya que el stock es propio de cada producto en cada sucursal
- CategoriasProductos
    - Id
        - Clave autoincremental que identifica únicamente la categoría de productos.
    - PadreId
        - Clave foránea a la misma tabla, me permite realizar una relación padre/hijo entre las categorías, pudiendo asi realizar un árbol de categorías.
- Facturas
    - Nro
        - Clave autoincrementa que me identifica únicamente a la factura.
    - DniCliente
        - Clave foránea que relaciona a la factura con el cliente a cual corresponde.
    - DniEmpleado
        - Clave foránea que relaciona la factura con el vendedor que hizo la venta.
    - SucursalId
        - Clave foránea que relaciona la factura con la sucursal en la cual se realizo la venta.
    - FormaPagoId

        - Clave foránea que relaciona la factura con cómo se va a pagar (efectivo o cuotas)
- FacturaProductos
    - FacturaNro y ProductoId
        - Son la clave primaria de la tabla y son respectivamente clave foránea a la tabla de Facturas y a la de Productos, me permite hacer la relación muchos a mucho entre estas dos tablas.
- FacturaPagos
    - Id
        - Clave autoincrementa que identifica únicamente a la factura.
    - FacturaNro
        - Clave foránea que me indica a que factura pertenece el pago.
    - DniCliente [FOREIGN KEY DOTNETPY.Clientes(Dni)]
    - DniEmpleado [FOREIGN KEY DOTNETPY.Empleados(Dni)]
- FormaPagoFacturas
    - Id
        - Clave autoincrementa que identifica únicamente a la forma de pago.
- IngresoStocks
    - Id
        - Clave autoincrementa que me identifica únicamente a los ingresos de stock.
    - SucursalId
        - Clave foránea que relaciona el ingreso de stock con la sucursal en la cual sucedio
    - ProductoId
        - Clave foránea que indica de que producto fue el ingreso de stock
    - EmpleadoId
        - Clave foránea que indica quien fue el analista que valido el ingreso de producto

Funciones

- ParseoCategorias

  Recibe la cadena de la categoría como esta en la tabla maestra, la parsea e inserta su contenido en la tabla de Categorías.
- CargaCategoriaProductos

  Recorre las distintas categorías existentes en la maestra y por cada una llama a ParseoCategorias.
- StringSplit

  Recibe una cadena y un separador y devuelve una tabla de una columna donde cada registro es una de las partes del contenido separado por el separador.
- GenerateUser

Recibe el nombre y apellido de un empleado y devuelve el nombre de usuario (en nuestro caso el nombre de usuario es generado con la primera letra del nombre y el apellido)

- GetCategoriaDescripcion
  Recibe el Id de una categoria, devuelve la categoria como se encontraba en la tabla maestra. Con todos sus padres y separado por ' ¦ '
- GetCategoryId
  Es la funcion inversa a la anterior, recibe la cadena de la categoria como esta en la maestra, y devuelve cual es el ID de la categoria en la tabla Categorias
- rfindchr
  Recibe una cadena y un caracter, y devuelve el índice de la última ocurrencia de ese caracter en la cadena.
- GetMayorCategoria
  Esta función recibe un id de categoría y retorna el id de su mayor categoría padre (raíz del árbol).

Stored Procedures

- GetClientesPremium
  La estrategia utilizada para este reporte fue en primer lugar localizar a los 30 clientes que tengan el mayor monto por compras acumulado en el año y sucursal elegidos, este primer select devuelve DNI, Nombre, Apellido, Monto total acumulado y número de la última factura.
  Luego, con este conjunto de resultados se realiza un join con la tabla de facturas con el número de la última factura del cliente, esto es para conseguir el dato de la fecha y el DNI del vendedor de la última compra del cliente.
  Luego por cada uno de los 30 clientes se ejecuta un sub-select sobre la tabla de facturas con un join con la tabla de productos de facturas buscando la sumatoria de la cantidad total de productos comprados por el cliente en el año y sucursal elegidos.
  A partir del desarrollo de este stored procedure el análisis del Query Execution Plan recomendaba la creación de un índice en la tabla de Facturas por el campo de id de sucursal incluyendo los campos Número, Fecha, DNI de Empleado y el monto total de la factura. Esto mejoró la performance de la ejecución de este reporte y resulto útil también en el desarrollo de los dos reportes restantes ya que el motor aprovecho la presencia de este índice por sucursal.
- GetTableroControl
  Debido a la complejidad del cálculo del producto con más días de faltante de stock en el año seleccionado se dividió en dos partes el desarrollo de este reporte.
  Se realizó un select sobre la tabla de facturas filtrando por la sucursal y año seleccionado, utilizando las siguientes funciones de agregación:
  - `COUNT(*)` para contabilizar el total de ventas del año y sucursal.
  - `SUM(Total)` para recuperar la sumatoria del monto total facturado en el año y sucursal.

- `SUM(CASE FormaPagoId WHEN 1 THEN 1 ELSE 0 END)` para sumar la cantidad de facturas pagadas en efectivo. Luego en la aplicación se muestran los porcentajes de facturas en efectivo y en cuotas en ese año y sucursal.
- `MAX(Total)` se busca la factura con mayor monto en ese año y sucursal.

Para buscar el mayor deudor se realiza un sub-select sobre la tabla de facturas con un left join a la tabla de pagos para buscar todos los pagos que se haya realizado dentro del año en que se realizó la factura. Se realiza la sumatoria de todos los pagos agrupando por factura, luego se realiza la diferencia entra el monto total comprado por el cliente y el monto total de sus pagos, el cliente con una diferencia más grande es el mayor deudor de la sucursal y año seleccionado.

Para la búsqueda del producto con más unidades vendidas se realizó un sub-select sobre la tabla de facturas con la tabla de productos de facturas realizando la sumatoria de cantidades vendidas agrupadas por producto.

- GetMejoresCategorias

  Para el desarrollo de este reporte se realizó una vista, llamada ventas de productos, sobre la tabla de facturas con un join con productos de facturas y otro sobre productos. Seleccionando los campos id de producto, id de categoría, precio unitario de la venta, cantidad vendida, fecha de la venta, id de la sucursal y DNI del empleado vendedor. Esta vista tiene un registro por cada venta de un producto.

  En el select principal se parte de un sub-select con el id de las mayores categorías y la cantidad de sub-categorías. Luego se realizan varios sub-selects sobre la tabla temporal definida mediante la sentencia WITH la cual especifica un conjunto de resultados temporal que se define en el ámbito de ejecución de una sola sentencia select (también puede ser update, insert, delete, etc...). Se definió como resultado temporal a los registros devueltos por la consulta sobre la vista de ventas de productos filtrado por el año y sucursal seleccionados.
- AddProducto

  Registra un producto nuevo siendo todos sus campos requeridos.
- GetProducto

  Retorna un único producto por su id.
- GetProductos

  Retorna todos los productos.
- GetProductosByFiltros

  Retorna todos los productos filtrando por Id, Nombre y Marca, precio desde, precio hasta y id de categoría. Siendo todos estos filtros opcionales.
- SetEnableProducto

  Actualiza el estado de baja lógica del producto.
- UpdateProducto

  Actualiza los campos modificables del producto.
- AddCliente

  Registra un cliente siendo todos sus campos requeridos.

- GetCliente
  Retorna un único cliente por su DNI.
- GetClientes
  Retorna todos los clientes.
- GetClientesByFiltros
  Retorna todos los clientes por DNI, Nombre, Apellido, id de provincia, bit de baja lógica. Siendo todos estos filtros opcionales.
- SetEnableCliente
  Actualiza el estado de baja lógica del producto.
- UpdateCliente
  Actualiza los campos modificables del cliente.
- AddEmpleado
  Registra un empleado siendo todos sus campos requeridos.
- GetEmpleado
  Retorna un único empleado por su DNI.
- GetEmpleados
  Retorna todos los clientes.
- GetEmpleadosByFiltros
  Retorna todos los empleados por DNI, Nombre, Apellido, id de tipo de empleado, id de provincia, id de sucursal y bit de baja lógica. Siendo todos los filtros opcionales.
- SetEnableEmpleado
  Actualiza el estado de baja lógica del empleado.
- UpdateEmpleado
  Actualiza los campos modificables del empleado.
- AddUsuario
  Registra un nuevo usuario siendo todos sus campos requeridos.
- GetUsuario
  Retorna un único usuario por el DNI del empleado asociado.
- GetUsuarios
  Retorna todos los usuarios.
- GetUsuariosByFiltros
  Retorna todos los empleados por DNI, Nombre, Apellido, id de tipo de empleado, id de provincia, id de sucursal. Siendo todos los filtros opcionales.
- SetEnableUsuario
  Actualiza el estado de baja lógica del usuario.
- UpdateUsuario
  Actualiza todos los campos modificables del usuario.
- AddRol
  Registra un rol en el sistema y sus funcionalidades asociadas.
- GetRol
  Retorna un único rol por su Id.

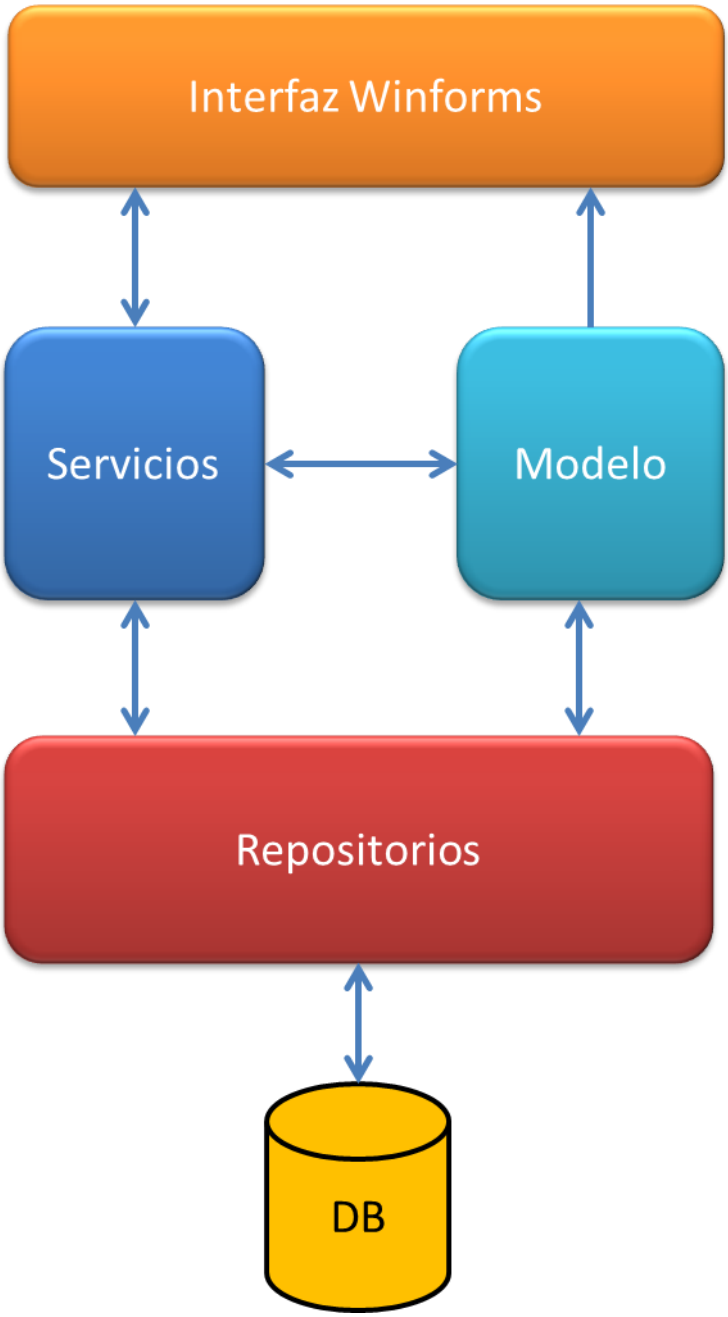
- GetRoles
  Retorna todos los roles.
- GetRolesByFiltros
  Retorna todos los roles por Id y Nombre. Siendo todos los filtros opcionales.
- SetEnableRol
  Actualiza el estado de baja lógica del rol.
- UpdateRol
  Actualiza todos los datos modificables del rol y sus funcionalidades asociadas.
- GetUsuarioByUsername
  Retorna un único usuario por su username. La condición de unicidad la garantiza un índice unique.
- GetFactura
  Retorna una factura por su número.
- GetFacturasImpagas
  Retorna todas las facturas con pagos pendientes para el cliente y sucursal solicitados.
- GetMarcas
  Retorna todas las marcas.
- GetFuncionalidades
  Retorna toda la lista de funcionalidades.
- GetProvincias
  Retorna todas las provincias.
- GetSucursales
  Retorna todas las sucursales.
- GetTipoEmpleados
  Retorna todos los tipos de empleados.
- GetProductoStocks
  Retorna la cantidad de stock del código de producto especificado por cada sucursal.
- GetProductoStocksBySucursal
  Retorna la cantidad de stock del id de sucursal indicado y los códigos de productos de una string separados por comas.
- AddStock
  Registra el ingreso de stock de un producto en la sucursal indicada.
- AddFacturaProducto
  Registra el Producto con su cantidad comprada en una factura.
- AddFacturaPago
  Registra el pago de una cuota de una factura. O el pago total de una factura en efectivo.
- GetPermisosByUsuario
  Retorna la lista de ids de todas las funcionalidades a las que tiene acceso el usuario indicado por su DNI.

Triggers

Tabla	Accion	Descripcion
IngresoStocks	After insert	Actualiza la tabla ProductoStock sumándole al producto que corresponda la cantidad ingresada
FacturaProductos	After insert	Actualiza la tabla ProductoStock restándole al producto que corresponda la cantidad vendida

Arquitectura de la Solución

La solución se dividió en dos proyectos para mantener una buena abstracción entre las capas. El proyecto Core incluye toda la capa de Servicios, el modelo de dominio y la capa de repositorios, mientras que el proyecto Desktop solo se encarga de la interacción con el usuario y presentar los datos, interacturando con el Core de la aplicación mediante los servicios expuestos.



Los repositorios son las únicas clases que pueden interactuar con repositorios de datos externos a la aplicación (base de datos, archivos de texto, web services, etc.).

Las entidades del modelo de dominio son las que contienen las reglas de negocio y validaciones solicitadas en las especificaciones.

Los servicios tienen como fin abstraer la interfaz de usuario de la lógica de negocio de la aplicación.

La interfaz de usuario es la encargada de presentar los datos al usuario, de una forma totalmente abstraída a las validaciones del negocio.

Microsoft Enterprise Library – Data Access Application Block

Esta librería le permite a la aplicación abstraerse de la implementación de clases utilizada para conectar a la base de datos, se encarga de seleccionar de forma automática el proveedor de datos en base al atributo providerName definido en la connection string en el App.config.

Al evitar que la aplicación utilice objetos tales como SqlDataReader o SqlCommand se mantiene una abstracción con la implementación del proveedor de base de datos utilizado, por ejemplo: con solo cambiar el providerName en la connection string a Oracle se podría utilizar los mismos stored procedures implementados en una base de datos Oracle.

Autofac

Esta librería se encarga de resolver las referencias a las implementaciones de las interfaces, esto es conocido como Inversion of Control (IoC). Se implementó en esta aplicación mediante un Service Locator (instancia singleton), este localizador utiliza un contenedor con todos los servicios registrados mediante autofac para resolver las implementaciones de las interfaces solicitadas.

Esto permite intercambiar las implementaciones de los servicios utilizados por las vistas, o los repositorios utilizados por los servicios y las entidades sin necesidad de modificar las clases que utilizan esas interfaces solamente modificando la registración de servicios, evitando el acoplamiento entre las capas de arquitectura y facilitando la realización de pruebas unitarias.

Resolución de Problemas

Parseo de categorías de productos

Se realizó mediante una función recursiva, la cual recibe un delimitador, y una cadena separada por ese delimitador y un flag que indica si lo está llamando el usuario, o se está llamando recursivamente

La función opera de la siguiente manera:

1) Busca las 2 primeras ocurrencias del delimitador, y en base a estos obtiene las primeras 2 categorías de la cadena. La primera es el padre, la segunda es el hijo (de acuerdo a como esta en la tabla maestra).
2) Si estoy en la primera iteración (en la cual la primera categoría va a ser la raíz) y la raíz no está en la tabla de categorías, la inserto, con padre nulo.
3) Luego, de no existir en la tabla de categorías la combinación padre/hijo hallada en 1) la inserto
4) Si en la cadena tenía más de 2 categorías, le saco a la cadena que me ingreso como parámetro la primera categoría y llamo recursivamente a la función con la nueva cadena y el flag en 0 (indicando que está siendo llamado recursivamente).

Datos faltantes de Clientes y Empleados

Al no haber datos sobre el teléfono, la dirección y la provincia de los clientes almacenados antes de la migración se decidió dejarles una cadena vacía a los primeros dos y al id de la provincia en la tabla de clientes dejarlo en uno. Luego en la próxima compra se actualizarán sus datos.

Al no estar almacenado el teléfono de los empleados antes de la migración se dejó ese campo con una cadena vacía.

Pagos de Facturas y Cuotas

Se considera que todas las facturas deben tener pago, incluso las que tienen forma de pago en efectivo. Si la factura es en efectivo y no tiene un registro en la tabla de pagos se considera que la factura esta impaga.

Todas las facturas migradas que tenían menos pagos que cantidad de cuotas convenidas se consideran impagas y con cuotas pendientes de pago por parte de los clientes deudores.

Numeración de Facturas

Como antes de la migración todas las facturas tenían un numero de pago consecutivo se siguió manteniendo este tipo de numeración centralizada independientemente de la sucursal que emita la factura.

Facturación y Registración de Pagos

Esta funcionalidad sólo la pueden utilizar los empleados de tipo vendedor que tengan el permiso necesario. Los analistas no pueden utilizar esta funcionalidad ni aun teniendo el permiso necesario para poder hacerlo.

Usuario Administrador General

Se creó un empleado con número de DNI 1 para poder dar de alta a este usuario que tiene todos los permisos posibles en el sistema de venta de electrodomésticos. Cuyo nombre de usuario y contraseña están definidos en el documento Enunciado.PDF.

Este empleado es de tipo analista por lo que no puede facturar ni registrar pagos.