Bases de Datos II

Leonel Martínez

Integrantes.

Angel Abraham Lopez Delgado

Hazael Gabriel Vásquez

José Ramón Figueroa Roma

Rene Antonio Arostegui Valle

Jeffry Javier Reyes Miranda

Base de datos princicpal:

Descripción General

La base de datos TallerCuero está diseñada para gestionar las operaciones de un taller de cuero, especializado en productos como bolsas, carteras, cinturones, zapatos y accesorios de cuero personalizados. La base de datos incluye tablas para manejar información de empleados, productos, proveedores, clientes, ventas, producción, inventario y seguridad, utilizando múltiples relaciones entre las tablas para asegurar la integridad y consistencia de los datos.

Tablas y Relaciones

1. Tabla Cat_Paises

- **Descripción:** Almacena los países disponibles.
- **Columnas:**
  - id_pais (INT, PRIMARY KEY): Identificador único del país.

- `nombre` (NVARCHAR(30)): Nombre del país.
- `abreviatura` (NVARCHAR(5)): Abreviatura del país.
- `codigoEntrada` (NVARCHAR(5)): Código de entrada del país.
- `nacionalidad` (NVARCHAR(30)): Nacionalidad correspondiente al país.

2. Tabla Departamentos

- **Descripción**: Almacena los departamentos asociados a un país.
- **Columnas**:
  - `id_departamento` (INT, PRIMARY KEY): Identificador único del departamento.
  - `departamento` (NVARCHAR(45)): Nombre del departamento.
  - `abreviatura` (NVARCHAR(5)): Abreviatura del departamento.
  - `pais` (INT, FOREIGN KEY): Referencia a `Cat_Paises(id_pais)`.

3. Tabla Municipios

- **Descripción**: Almacena los municipios asociados a un departamento.
- **Columnas**:
  - `id_municipio` (INT, PRIMARY KEY): Identificador único del municipio.
  - `municipio` (NVARCHAR(35)): Nombre del municipio.
  - `abreviatura` (NVARCHAR(5)): Abreviatura del municipio.
  - `departamento` (INT, FOREIGN KEY): Referencia a `Departamentos(id_departamento)`.

4. Tabla Ubicaciones

- **Descripción**: Almacena las ubicaciones geográficas.
- **Columnas**:
  - `id_ubicacion` (INT, PRIMARY KEY): Identificador único de la ubicación.
  - `direccion` (NVARCHAR(50)): Dirección de la ubicación.
  - `municipio` (INT, FOREIGN KEY): Referencia a `Municipios(id_municipio)`.
  - `codigoPostal` (INT): Código postal de la ubicación.
  - `calle` (NVARCHAR(35)): Nombre de la calle.

5. Tabla Areas

- **Descripción**: Almacena las áreas de trabajo dentro de la empresa.
- **Columnas**:

- id_area (INT, PRIMARY KEY): Identificador único del área.
- area (NVARCHAR(30)): Nombre del área.

6. Tabla Cargos

- **Descripción**: Almacena los cargos disponibles en la empresa.
- **Columnas**:
  - id_cargo (INT, PRIMARY KEY): Identificador único del cargo.
  - area (INT, FOREIGN KEY): Referencia a Areas(id_area).
  - cargo (VARCHAR(30)): Nombre del cargo.
  - abreviaturaCargo (NVARCHAR(5)): Abreviatura del cargo.
  - descripcion (NVARCHAR(100)): Descripción del cargo.

7. Tabla Sucursales

- **Descripción**: Almacena la información de las sucursales.
- **Columnas**:
  - id_sucursal (INT, PRIMARY KEY): Identificador único de la sucursal.
  - nombreSucursal (NVARCHAR(45)): Nombre de la sucursal.
  - RUC (NVARCHAR(20)): Registro Único de Contribuyentes de la sucursal.
  - FechaConstitucion (DATE): Fecha de constitución de la sucursal.
  - ubicacion (INT, FOREIGN KEY): Referencia a Ubicaciones(id_ubicacion).

8. Tabla Cat_TipoIdentificaciones

- **Descripción**: Almacena los tipos de identificaciones aceptados.
- **Columnas**:
  - id_tipoIdentificacion (INT, PRIMARY KEY): Identificador único del tipo de identificación.
  - tipoID (NVARCHAR(30)): Nombre del tipo de identificación.

9. Tabla Identificaciones

- **Descripción**: Almacena las identificaciones de los empleados.
- **Columnas**:
  - id_identificacion (INT, PRIMARY KEY): Identificador único de la identificación.
  - numeroIdentificacion (NVARCHAR(30)): Número de identificación.

- `tipoIdentificacion` (INT, FOREIGN KEY): Referencia a `Cat_TipoIdentificaciones(id_tipoIdentificacion)`.
- `fechavencimiento` (DATE): Fecha de vencimiento de la identificación.

10. Tabla Cat_Generos

- **Descripción**: Almacena los géneros.
- **Columnas**:
  - `id_genero` (INT, IDENTITY(1,1), PRIMARY KEY): Identificador único del género.
  - `genero` (NVARCHAR(10)): Nombre del género.

11. Tabla Empleados

- **Descripción**: Almacena la información de los empleados.
- **Columnas**:
  - `id_empleado` (INT, IDENTITY(1,1), PRIMARY KEY): Identificador único del empleado.
  - `nombres` (VARCHAR(20)): Nombres del empleado.
  - `apellidos` (VARCHAR(15)): Apellidos del empleado.
  - `inss` (VARCHAR(10)): Número de seguridad social del empleado.
  - `identificacion` (INT, FOREIGN KEY): Referencia a `Identificaciones(id_identificacion)`.
  - `genero` (INT, FOREIGN KEY): Referencia a `Cat_Generos(id_genero)`.

12. Tabla Cat_Telefonos

- **Descripción**: Almacena la información de los teléfonos.
- **Columnas**:
  - `id_telefono` (INT, PRIMARY KEY): Identificador único del teléfono.
  - `numeroArea` (NVARCHAR(6)): Número de área.
  - `numeroTelefonico` (NVARCHAR(18)): Número telefónico.
  - `tipo` (NVARCHAR(45)): Tipo de teléfono.
  - `habilitado` (BIT): Estado de habilitación del teléfono.

13. Tabla Empleados_Telefonos

- **Descripción**: Relaciona los empleados con sus teléfonos.
- **Columnas**:
  - `id_empleado` (INT, FOREIGN KEY): Referencia a `Empleados(id_empleado)`.

- id_telefono (INT, FOREIGN KEY): Referencia a Cat_Telefonos(id_telefono).
- PRIMARY KEY (id_empleado, id_telefono).

14. Tabla DetallesEmpleado

- **Descripción**: Almacena los detalles adicionales de los empleados.
- **Columnas**:
  - empleado (INT, FOREIGN KEY): Referencia a Empleados(id_empleado).
  - estadoCivil (NVARCHAR(20)): Estado civil del empleado.
  - ubicacion (INT, FOREIGN KEY): Referencia a Ubicaciones(id_ubicacion).
  - estado (BIT): Estado del empleado (activo/inactivo).
  - cargo (INT, FOREIGN KEY): Referencia a Cargos(id_cargo).
  - sucursal (INT, FOREIGN KEY): Referencia a Sucursales(id_sucursal).
  - correoElectronico (NVARCHAR(25)): Correo electrónico del empleado.

15. Tabla DetallesSucursales

- **Descripción**: Almacena los detalles de las sucursales.
- **Columnas**:
  - sucursal (INT, FOREIGN KEY): Referencia a Sucursales(id_sucursal).
  - horaApertura (TIME): Hora de apertura de la sucursal.
  - horaCierre (TIME): Hora de cierre de la sucursal.
  - encargadoSucursal (INT, FOREIGN KEY): Referencia a Empleados(id_empleado).
  - Telefono (INT, FOREIGN KEY): Referencia a Cat_Telefonos(id_telefono).

16. Tabla HistorialCargosEmpleados

- **Descripción**: Almacena el historial de cargos de los empleados.
- **Columnas**:
  - empleado (INT, FOREIGN KEY): Referencia a Empleados(id_empleado).
  - cargo (INT, FOREIGN KEY): Referencia a Cargos(id_cargo).
  - fechaAsignacion (DATETIME): Fecha de asignación del cargo.
  - fechaRevocacion (DATETIME): Fecha de revocación del cargo.

- sucursal (INT, FOREIGN KEY): Referencia a Sucursales(id_sucursal).

17. Tabla HistorialEmpleados

- **Descripción**: Almacena el historial de los empleados en la empresa.
- **Columnas**:
  - empleado (INT, FOREIGN KEY): Referencia a Empleados(id_empleado).
  - sucursal (INT, FOREIGN KEY): Referencia a Sucursales(id_sucursal).
  - cargo (INT, FOREIGN KEY): Referencia a Cargos(id_cargo).
  - estado (BIT): Estado del empleado (activo/inactivo).
  - fechaContratacion (DATE): Fecha de contratación del empleado.
  - fechaRetiro (DATE): Fecha de retiro del empleado.
  - observaciones (NVARCHAR(150)): Observaciones sobre el empleado.

18. Tabla Clientes

- **Descripción**: Almacena la información de los clientes.
- **Columnas**:
  - id_cliente (INT, PRIMARY KEY): Identificador único del cliente.
  - nombres (NVARCHAR(30)): Nombres del cliente.
  - apellidos (NVARCHAR(25)): Apellidos del cliente.
  - direccion (INT, FOREIGN KEY): Referencia a Ubicaciones(id_ubicacion).
  - identificacion (INT, FOREIGN KEY): Referencia a Identificaciones(id_identificacion).
  - genero (INT, FOREIGN KEY): Referencia a Cat_Generos(id_genero).
  - fechaNacimiento (DATE): Fecha de nacimiento del cliente.
  - estado (BIT): Estado del cliente (activo/inactivo).

19. Tabla Proveedores

- **Descripción**: Almacena la información de los proveedores.
- **Columnas**:
  - id_proveedor (INT, PRIMARY KEY): Identificador único del proveedor.
  - nombre (NVARCHAR(45)): Nombre del proveedor.
  - ruc (NVARCHAR(20)): Registro Único de Contribuyentes del proveedor.

- direccion (INT, FOREIGN KEY): Referencia a Ubicaciones(id_ubicacion).
- telefono (INT, FOREIGN KEY): Referencia a Cat_Telefonos(id_telefono).
- correo (NVARCHAR(45)): Correo electrónico del proveedor.

20. Tabla MateriaPrima

- **Descripción**: Almacena la información de la materia prima.
- **Columnas**:
  - id_materiaPrima (INT, PRIMARY KEY): Identificador único de la materia prima.
  - nombre (NVARCHAR(45)): Nombre de la materia prima.
  - descripcion (NVARCHAR(100)): Descripción de la materia prima.
  - unidadMedida (NVARCHAR(10)): Unidad de medida de la materia prima.
  - estado (BIT): Estado de la materia prima (disponible/no disponible).

21. Tabla Inventario

- **Descripción**: Almacena la información del inventario.
- **Columnas**:
  - id_inventario (INT, PRIMARY KEY): Identificador único del inventario.
  - materiaPrima (INT, FOREIGN KEY): Referencia a MateriaPrima(id_materiaPrima).
  - cantidad (FLOAT): Cantidad disponible en inventario.
  - sucursal (INT, FOREIGN KEY): Referencia a Sucursales(id_sucursal).

22. Tabla Productos

- **Descripción**: Almacena la información de los productos.
- **Columnas**:
  - id_producto (INT, PRIMARY KEY): Identificador único del producto.
  - nombre (NVARCHAR(45)): Nombre del producto.
  - descripcion (NVARCHAR(100)): Descripción del producto.
  - precio (DECIMAL(10, 2)): Precio del producto.
  - stock (INT): Cantidad disponible en stock.
  - estado (BIT): Estado del producto (disponible/no disponible).

23. Tabla Ventas

- **Descripción**: Almacena la información de las ventas.
- **Columnas**:
  - id_venta (INT, PRIMARY KEY): Identificador único de la venta.
  - cliente (INT, FOREIGN KEY): Referencia a Clientes(id_cliente).
  - fecha (DATETIME): Fecha de la venta.
  - montoTotal (DECIMAL(10, 2)): Monto total de la venta.
  - empleado (INT, FOREIGN KEY): Referencia a Empleados(id_empleado).

24. Tabla DetallesVenta

- **Descripción**: Almacena los detalles de cada venta.
- **Columnas**:
  - id_detalleVenta (INT, PRIMARY KEY): Identificador único del detalle de venta.
  - venta (INT, FOREIGN KEY): Referencia a Ventas(id_venta).
  - producto (INT, FOREIGN KEY): Referencia a Productos(id_producto).
  - cantidad (INT): Cantidad de producto vendido.
  - precioUnitario (DECIMAL(10, 2)): Precio unitario del producto.
  - subtotal (DECIMAL(10, 2)): Subtotal del detalle de venta.

25. Tabla Produccion

- **Descripción**: Almacena la información de la producción.
- **Columnas**:
  - id_produccion (INT, PRIMARY KEY): Identificador único de la producción.
  - producto (INT, FOREIGN KEY): Referencia a Productos(id_producto).
  - materiaPrima (INT, FOREIGN KEY): Referencia a MateriaPrima(id_materiaPrima).
  - cantidad (FLOAT): Cantidad producida.
  - fechaProduccion (DATETIME): Fecha de producción.
  - empleado (INT, FOREIGN KEY): Referencia a Empleados(id_empleado).

Implementación SQL

Creación de la Base de Datos

```
sql
Copy code
USE master;
GO
```

Esta documentación proporciona una visión completa de la estructura de la base de datos `TallerCuero_final`, detallando las tablas, sus columnas y relaciones. Esta base de datos está diseñada para apoyar las operaciones de un taller de cuero, facilitando la gestión eficiente de empleados, productos, proveedores, clientes, inventario y procesos de producción y ventas.

Documentación del Proyecto: Configuración de Replicación de Base de Datos

Descripción General

El objetivo de este proyecto es configurar y documentar la replicación de bases de datos en un entorno SQL Server. La replicación permite copiar y distribuir datos y objetos de bases de datos de una base de datos a otra y mantener la coherencia entre ellas. En este proyecto, configuramos el servidor de SQL como distribuidor y publicador, así como la base de datos de distribución necesaria para la replicación.

Pasos de Configuración

1. Instalación del Servidor como Distribuidor

El servidor `DESKTOP-SQQF78L` se configura como distribuidor. Un distribuidor es un servidor que contiene la base de datos de distribución y almacena metadatos e historial de las transacciones y comandos replicados.

```
sql
Copy code
```

```
use master;
exec sp_adddistributor @distributor = N'DESKTOP-SQQF78L',
@password = N'';
GO
```

- **Parámetros**:
  - `@distributor`: Especifica el nombre del servidor que se está configurando como distribuidor.
  - `@password`: Contraseña para la conexión del distribuidor.

2. Creación de la Base de Datos de Distribución

Se crea la base de datos `Distribuidotaller` que almacenará los datos de distribución necesarios para la replicación.

```
sql
Copy code
exec sp_adddistributiondb @database = N'Distribuidotaller',
    @data_folder = N'D:\Program Files\Microsoft SQL
Server\MSSQL16.MSSQLSERVER\MSSQL\Data',
    @log_folder = N'D:\Program Files\Microsoft SQL
Server\MSSQL16.MSSQLSERVER\MSSQL\Data',
    @log_file_size = 2,
    @min_distretention = 0,
    @max_distretention = 72,
    @history_retention = 48,
    @deletebatchsize_xact = 5000,
    @deletebatchsize_cmd = 2000,
    @security_mode = 1;
GO
```

- **Parámetros**:
  - `@database`: Nombre de la base de datos de distribución.
  - `@data_folder`: Ruta del directorio de datos.
  - `@log_folder`: Ruta del directorio de logs.
  - `@log_file_size`: Tamaño del archivo de log.
  - `@min_distretention`: Retención mínima de la distribución en horas.
  - `@max_distretention`: Retención máxima de la distribución en horas.
  - `@history_retention`: Retención del historial en horas.

- `@deletebatchsize_xact`: Tamaño del lote de transacciones a eliminar.
- `@deletebatchsize_cmd`: Tamaño del lote de comandos a eliminar.
- `@security_mode`: Modo de seguridad (1 indica autenticación de Windows).

3. Configuración de Propiedades Extendidas

Configurar propiedades extendidas en la base de datos de distribución.

```
sql
Copy code
use [Distribuidotaller];
if (not exists (select * from sysobjects where name =
'UIProperties' and type = 'U '))
    create table UIProperties(id int);

if (exists (select * from
::fn_listextendedproperty('SnapshotFolder', 'user', 'dbo',
'table', 'UIProperties', null, null)))
    EXEC sp_updateextendedproperty N'SnapshotFolder',
N'D:\replic', 'user', dbo, 'table', 'UIProperties';
else
    EXEC sp_addextendedproperty N'SnapshotFolder', N'D:\replic',
'user', dbo, 'table', 'UIProperties';
GO
```

- **Propiedad Extendida**:
  - `SnapshotFolder`: Directorio donde se almacenarán los snapshots de la replicación.

4. Agregar el Publicador

Configurar el servidor `DESKTOP-SQQF78L` como publicador.

```
sql
Copy code
exec sp_adddistpublisher @publisher = N'DESKTOP-SQQF78L',
    @distribution_db = N'Distribuidotaller',
    @security_mode = 1,
    @working_directory = N'D:\replic',
    @trusted = N'false',
```

```
    @thirdparty_flag = 0,
    @publisher_type = N'MSSQLSERVER';
GO
```

- **Parámetros**:
    - `@publisher`: Nombre del servidor que se está configurando como publicador.
    - `@distribution_db`: Nombre de la base de datos de distribución.
    - `@security_mode`: Modo de seguridad (1 indica autenticación de Windows).
    - `@working_directory`: Directorio de trabajo para la replicación.
    - `@trusted`: Indicador de confianza (false indica que no se confía automáticamente).
    - `@thirdparty_flag`: Indicador para terceros (0 indica que no es un publicador de terceros).
    - `@publisher_type`: Tipo de publicador (MSSQLSERVER indica SQL Server).

Conclusión

Este documento detalla los pasos necesarios para configurar la replicación en un entorno SQL Server. Se cubren aspectos como la instalación del distribuidor, la creación de la base de datos de distribución, la configuración de propiedades extendidas y la adición de un publicador. La replicación mejora la disponibilidad y escalabilidad de los datos, lo que es esencial para mantener la continuidad del negocio y la eficiencia operativa.

Documentación del Modelo de Base de Datos Orientado a Objetos

Cat_Paises

- **Descripción**: Almacena información sobre países.
- **Atributos**:
    - `idPais`: Identificador único del país.
    - `nombre`: Nombre del país.
    - `abreviatura`: Abreviatura del nombre del país.
    - `codigoEntrada`: Código de entrada del país.
    - `nacionalidad`: Nacionalidad del país.

Departamentos

- **Descripción**: Almacena información sobre departamentos dentro de un país.
- **Atributos**:
  - `idDepartamento`: Identificador único del departamento.
  - `departamento`: Nombre del departamento.
  - `abreviatura`: Abreviatura del nombre del departamento.
  - `pais`: Referencia al país al que pertenece el departamento.

Empleados

- **Descripción**: Almacena información sobre empleados.
- **Atributos**:
  - `idEmpleado`: Identificador único del empleado.
  - `nombres`: Nombres del empleado.
  - `apellidos`: Apellidos del empleado.
  - `inss`: Número de Seguro Social del empleado.
  - `identificacion`: Referencia a la identificación del empleado.
  - `genero`: Referencia al género del empleado.

Consideraciones Adicionales

- **Tipos de Datos**: Define tipos de datos personalizados para mantener consistencia y facilitar el manejo de datos.
- **Relaciones**: Explica las relaciones entre clases (tablas) como asociaciones entre objetos.
- **Indices y Restricciones**: Documenta cualquier índice o restricción importante para asegurar la integridad de los datos y mejorar el rendimiento de consultas.

Conclusiones

Esta documentación proporciona una visión clara de la estructura de tu base de datos orientada a objetos, facilitando su comprensión y mantenimiento a lo largo del tiempo.

Documentación del Modelo de Base de Datos Orientado a Objetos

Cat_Paises

- **Descripción**: Representa un país.
- **Atributos**:
  - `idPais`: `int` - Identificador único del país.
  - `nombre`: `string` - Nombre del país.
  - `abreviatura`: `string` - Abreviatura del nombre del país.
  - `codigoEntrada`: `string` - Código de entrada del país.
  - `nacionalidad`: `string` - Nacionalidad del país.

Departamentos

- **Descripción**: Representa un departamento dentro de un país.
- **Atributos**:
  - `idDepartamento`: `int` - Identificador único del departamento.
  - `departamento`: `string` - Nombre del departamento.
  - `abreviatura`: `string` - Abreviatura del nombre del departamento.
  - `pais`: `Cat_Paises` - País al que pertenece el departamento.

Municipios

- **Descripción**: Representa un municipio dentro de un departamento.
- **Atributos**:
  - `idMunicipio`: `int` - Identificador único del municipio.
  - `municipio`: `string` - Nombre del municipio.
  - `abreviatura`: `string` - Abreviatura del nombre del municipio.
  - `departamento`: `Departamentos` - Departamento al que pertenece el municipio.

Ubicaciones

- **Descripción**: Representa una ubicación física.
- **Atributos**:
  - `idUbicacion`: `int` - Identificador único de la ubicación.
  - `direccion`: `string` - Dirección física.
  - `municipio`: `Municipios` - Municipio donde se encuentra la ubicación.
  - `codigoPostal`: `int` - Código postal de la ubicación.
  - `calle`: `string` - Nombre de la calle.

Areas

- **Descripción**: Representa áreas funcionales de la organización.
- **Atributos**:
  - `idArea`: `int` - Identificador único del área.
  - `area`: `string` - Nombre del área.

Cargos

- **Descripción**: Representa cargos o roles dentro de la organización.
- **Atributos**:
  - `idCargo`: `int` - Identificador único del cargo.
  - `area`: `Areas` - Área a la que pertenece el cargo.
  - `cargo`: `string` - Nombre del cargo.
  - `abreviaturaCargo`: `string` - Abreviatura del cargo.
  - `descripcion`: `string` - Descripción del cargo.

Sucursales

- **Descripción**: Representa las sucursales de la organización.
- **Atributos**:
  - `idSucursal`: `int` - Identificador único de la sucursal.
  - `nombreSucursal`: `string` - Nombre de la sucursal.
  - `RUC`: `string` - Registro único del contribuyente de la sucursal.
  - `fechaConstitucion`: `DateTime` - Fecha de constitución de la sucursal.
  - `ubicacion`: `Ubicaciones` - Ubicación física de la sucursal.

Cat_TipoIdentificaciones

- **Descripción**: Representa tipos de identificaciones.
- **Atributos**:
  - `idTipoIdentificacion`: `int` - Identificador único del tipo de identificación.
  - `tipoID`: `string` - Nombre o descripción del tipo de identificación.

Identificaciones

- **Descripción**: Representa las identificaciones de personas.
- **Atributos**:
  - `idIdentificacion`: `int` - Identificador único de la identificación.

    - numeroIdentificacion: `string` - Número de identificación.
    - `tipoIdentificacion`: `Cat_TipoIdentificaciones` - Tipo de identificación.
    - `fechaVencimiento`: `DateTime` - Fecha de vencimiento de la identificación.

Cat_Generos

- **Descripción**: Representa géneros (masculino, femenino, etc.).
- **Atributos**:
    - `idGenero`: `int` - Identificador único del género.
    - `genero`: `string` - Nombre del género.

Empleados

- **Descripción**: Representa empleados de la organización.
- **Atributos**:
    - `idEmpleado`: `int` - Identificador único del empleado.
    - `nombres`: `string` - Nombres del empleado.
    - `apellidos`: `string` - Apellidos del empleado.
    - `inss`: `string` - Número de Seguro Social del empleado.
    - `identificacion`: `Identificaciones` - Identificación del empleado.
    - `genero`: `Cat_Generos` - Género del empleado.

Cat_Telefonos

- **Descripción**: Representa tipos de teléfonos.
- **Atributos**:
    - `idTelefono`: `int` - Identificador único del teléfono.
    - `numeroArea`: `string` - Número de área del teléfono.
    - `numeroTelefonico`: `string` - Número telefónico.
    - `tipo`: `string` - Tipo de teléfono.
    - `habilitado`: `bool` - Indica si el teléfono está habilitado.

Empleados_Telefonos

- **Descripción**: Asocia empleados con números de teléfono.
- **Atributos**:
    - `idEmpleado`: `int` - Identificador único del empleado.
    - `idTelefono`: `int` - Identificador único del teléfono.

DetallesEmpleado

- **Descripción**: Detalles adicionales de los empleados.
- **Atributos**:
  - `empleado`: `Empleados` - Empleado al que pertenecen los detalles.
  - `estadoCivil`: `string` - Estado civil del empleado.
  - `ubicacion`: `Ubicaciones` - Ubicación del empleado.
  - `estado`: `bool` - Estado del empleado (activo/inactivo).
  - `cargo`: `Cargos` - Cargo del empleado.
  - `sucursal`: `Sucursales` - Sucursal a la que está asignado el empleado.
  - `correoElectronico`: `string` - Correo electrónico del empleado.

DetallesSucursales

- **Descripción**: Detalles específicos de las sucursales.
- **Atributos**:
  - `sucursal`: `Sucursales` - Sucursal a la que pertenecen los detalles.
  - `horaApertura`: `Time` - Hora de apertura de la sucursal.
  - `horaCierre`: `Time` - Hora de cierre de la sucursal.
  - `encargadoSucursal`: `Empleados` - Empleado encargado de la sucursal.
  - `telefono`: `Cat_Telefonos` - Teléfono de contacto de la sucursal.

HistorialCargosEmpleados

- **Descripción**: Historial de cargos de los empleados en diferentes sucursales.
- **Atributos**:
  - `empleado`: `Empleados` - Empleado al que pertenece el historial.
  - `cargo`: `Cargos` - Cargo del empleado.
  - `fechaAsignacion`: `DateTime` - Fecha de asignación del cargo.
  - `fechaRevocacion`: `DateTime` - Fecha de revocación del cargo.
  - `sucursal`: `Sucursales` - Sucursal donde se asignó el cargo.

HistorialEmpleados

- **Descripción**: Historial de empleados en diferentes roles y sucursales.
- **Atributos**:
  - `empleado`: `Empleados` - Empleado al que pertenece el historial.
  - `sucursal`: `Sucursales` - Sucursal donde trabajó el empleado.

    - `cargo`: `Cargos` - Cargo que ocupó el empleado.
    - `fechaRegistro`: `DateTime` - Fecha de registro del historial.
    - `fechaIngreso`: `DateTime` - Fecha de ingreso del empleado.
    - `fechaSalida`: `DateTime` - Fecha de salida del empleado.

Cat_Medidas

- **Descripción**: Representa unidades de medida para productos.
- **Atributos**:
    - `idUnidadMedida`: `int` - Identificador único de la unidad de medida.
    - `nombreUnidad`: `string` - Nombre de la unidad de medida.
    - `abreviatura`: `string` - Abreviatura de la unidad de medida.

Cat_Colores

- **Descripción**: Representa colores disponibles para productos.
- **Atributos**:
    - `idColor`: `int` - Identificador único del color.
    - `nombreColor`: `string` - Nombre del color.
    - `codigoHex`: `string` - Código hexadecimal del color.

Productos

- **Descripción**: Representa productos disponibles en la tienda.
- **Atributos**:
    - `idProducto`: `int` - Identificador único del producto.
    - `nombreProducto`: `string` - Nombre del producto.
    - `descripcion`: `string` - Descripción del producto.
    - `precioUnitario`: `decimal` - Precio unitario del producto.
    - `unidadMedida`: `Cat_Medidas` - Unidad de medida del producto.
    - `color`: `Cat_Colores` - Color del producto.

DetallesProductos

- **Descripción**: Detalles específicos de los productos.
- **Atributos**:
    - `producto`: `Productos` - Producto al que pertenecen los detalles.
    - `codigoProducto`: `string` - Código único del producto.
    - `fechaRegistro`: `DateTime` - Fecha de registro del producto.
    - `ultimaActualizacion`: `DateTime` - Fecha de la última actualización del producto.

- `imagen`: `byte[]` - Imagen del producto en formato binario.

StockProductos

- **Descripción**: Gestiona el inventario de productos en las sucursales.
- **Atributos**:
  - `producto`: `Productos` - Producto al que pertenece el stock.
  - `sucursal`: `Sucursales` - Sucursal donde se gestiona el stock del producto.
  - `cantidad`: `int` - Cantidad de unidades en stock.

DetalleVenta

- **Descripción**: Detalle de las ventas realizadas.
- **Atributos**:
  - `venta`: `Ventas` - Venta a la que pertenece el detalle.
  - `producto`: `Productos` - Producto vendido.
  - `cantidad`: `int` - Cantidad vendida.
  - `precioUnitario`: `decimal` - Precio unitario al momento de la venta.

Ventas

- **Descripción**: Registro de las ventas realizadas.
- **Atributos**:
  - `idVenta`: `int` - Identificador único de la venta.
  - `fechaVenta`: `DateTime` - Fecha y hora de la venta.
  - `cliente`: `Clientes` - Cliente que realizó la compra.
  - `sucursal`: `Sucursales` - Sucursal donde se realizó la venta.
  - `totalVenta`: `decimal` - Total de la venta.

Clientes

- **Descripción**: Representa clientes que realizan compras.
- **Atributos**:
  - `idCliente`: `int` - Identificador único del cliente.
  - `nombres`: `string` - Nombres del cliente.
  - `apellidos`: `string` - Apellidos del cliente.
  - `identificacion`: `Identificaciones` - Identificación del cliente.
  - `telefono`: `Cat_Telefonos` - Teléfono de contacto del cliente.
  - `correoElectronico`: `string` - Correo electrónico del cliente.

DetalleCompra

- **Descripción**: Detalle de las compras realizadas.
- **Atributos**:
  - `compra`: `Compras` - Compra a la que pertenece el detalle.
  - `producto`: `Productos` - Producto comprado.
  - `cantidad`: `int` - Cantidad comprada.
  - `precioUnitario`: `decimal` - Precio unitario al momento de la compra.

Compras

- **Descripción**: Registro de las compras realizadas.
- **Atributos**:
  - `idCompra`: `int` - Identificador único de la compra.
  - `fechaCompra`: `DateTime` - Fecha y hora de la compra.
  - `proveedor`: `Proveedores` - Proveedor que realizó la venta.
  - `sucursal`: `Sucursales` - Sucursal donde se realizó la compra.
  - `totalCompra`: `decimal` - Total de la compra.

Proveedores

- **Descripción**: Representa proveedores de productos.
- **Atributos**:
  - `idProveedor`: `int` - Identificador único del proveedor.
  - `nombreProveedor`: `string` - Nombre del proveedor.
  - `direccion`: `string` - Dirección del proveedor.
  - `telefono`: `Cat_Telefonos` - Teléfono de contacto del proveedor.
  - `correoElectronico`: `string` - Correo electrónico del proveedor.

Almacenamiento y Transacciones

1. Introducción

Este documento detalla la estructura de almacenamiento y las operaciones de transacciones implementadas en el sistema de gestión para el taller de cuero 'Arte en Piel'. Se abordan aspectos clave como la gestión de datos, almacenamiento en bases de datos relacionales y operaciones transaccionales para garantizar la integridad y eficiencia del sistema.

2. Almacenamiento de Datos

2.1. Bases de Datos Relacionales

El sistema utiliza una base de datos relacional para almacenar datos estructurados de manera organizada. Las principales entidades y sus relaciones se detallan en el esquema relacional.

- **Tablas Principales:**
  - **Clientes:** Almacena información detallada de los clientes, incluyendo nombres, apellidos, contactos y medidas específicas.
  - **Empleados:** Registra datos de los empleados, incluyendo información personal, identificaciones y detalles de contacto.
  - **Productos:** Contiene información detallada sobre los productos fabricados, como nombre, color, cuero utilizado y precios.
  - **Proveedores:** Mantiene registros de los proveedores, incluyendo información de contacto y detalles de transacciones.
  - **Inventario:** Gestiona el stock de productos y materiales utilizados en la producción.
  - **Ventas:** Registra las transacciones de ventas, incluyendo detalles de productos vendidos, precios y clientes.

2.2. Operaciones de Base de Datos

Las operaciones de base de datos aseguran la gestión eficiente y segura de los datos:

- **Inserción:** Permite la adición de nuevos registros de clientes, empleados, productos, proveedores y transacciones de ventas.
- **Actualización:** Modifica información existente como detalles de productos, inventario y datos de contacto.
- **Eliminación:** Gestiona la remoción de registros obsoletos o no deseados, manteniendo la integridad de los datos relacionales.

3. Transacciones y Gestión de Datos

3.1. Operaciones Transaccionales

Las transacciones se gestionan para asegurar la integridad y consistencia de los datos en las operaciones diarias.

- **Transacciones ACID:** Cumple con las propiedades ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) para garantizar transacciones seguras y completas.
- **Control de Concurrencia:** Gestiona el acceso simultáneo de múltiples usuarios para prevenir conflictos y asegurar la coherencia de los datos.

3.2. Seguridad y Control de Acceso

Se implementan políticas de seguridad para proteger los datos sensibles y restringir el acceso no autorizado.

- **Autenticación y Autorización:** Utilización de sistemas de gestión de usuarios y grupos para controlar el acceso a diferentes niveles de datos y funcionalidades del sistema.
- **Auditoría y Registro:** Registro de todas las actividades críticas, como modificaciones de datos y accesos, para cumplir con requisitos de auditoría y mantener la trazabilidad.

4. Conclusiones

Esto ofrece una visión general del almacenamiento y las transacciones implementadas en el sistema de gestión del taller de cuero 'Arte en Piel'. La estructura relacional y las operaciones transaccionales aseguran la eficiencia operativa y facilitan análisis detallados para la toma de decisiones estratégicas y tácticas.

Proceso de Llenado Masivo de Datos

1. Introducción

Este documento describe el proceso de importación masiva de datos desde archivos CSV a las tablas de la base de datos relacional del taller de cuero 'Arte en Piel'. Se utilizó la instrucción BULK INSERT en SQL Server para este propósito.

2. Esquema de Tablas y Archivos CSV

A continuación se detallan las tablas en la base de datos 'TallerCuero_final' y los archivos CSV utilizados para el llenado masivo de datos:

2.1. Tablas

- **Cat_TipoIdentificaciones**
- **Identificaciones**
- **Cat_Generos**
- **Empleados**
- **Cat_Colores**
- **Cat_Medidas**
- **Cat_Paises**
- **Departamentos**
- **Municipios**
- **Ubicaciones**
- **Areas**
- **Cargos**
- **Sucursales**
- **Cat_Telefonos**
- **DetallesEmpleado**
- **DetallesSucursales**
- **HistorialCargosEmpleados**
- **HistorialEmpleados**
- **Cat_Cueros**
- **Categorias**
- **subcategorias**
- **Impuestos**
- **PresentacionProducto**
- **Productos**
- **Cat_Monedas**
- **Precios**
- **DetallesProductos**
- **Proveedores**
- **detalle_proveedor**
- **Clientes**
- **Detalle_Cliente**
- **PlanesProduccionProductos**
- **Materiales**
- **MaterialesUsadosPlanesProduccion**
- **Inventario**
- **venta**
- **detalleventa**
- **controlPago**
- **Cajas**
- **AperturasCajas**
- **DetalleAperturaCajas**
- **ArqueoCaja**

- CierresCajas
- Transacciones

2.2. Archivos CSV

- CATID.csv
- Identificaciones.csv
- genero.csv
- Empleados.csv
- CATcolor.csv
- Cat_Medidas.csv
- Cat_Paises.csv
- Departamentos.csv
- Municipios.csv
- Ubicaciones.csv
- Areas.csv
- Cargos.csv
- Sucursales.csv
- Cat_Telefonos.csv
- DetallesEmpleado.csv
- DetallesSucursales.csv
- HistorialCargosEmpleados.csv
- HistorialEmpleados.csv
- Cat_Cueros.csv
- Categorias.csv
- Subcategorias.csv
- Impuestos.csv
- Presentacion.csv
- Productos.csv
- Monedas.csv
- Precios.csv
- Detallesprod.csv
- Provedor.csv
- Detalleprov.csv
- Clientes.csv
- DetalleCliente.csv
- Produccion.csv
- Materiales.csv
- MaterialesUsados.csv
- Inventario.csv
- venta.csv
- detalleventa.csv
- controlpago.csv
- Cajas.csv

- **AperturasCajas.csv**
- **DetalleAperturaCajas.csv**
- **ArqueoCaja.csv**
- **CierresCajas.csv**
- **Transacciones.csv**

3. Proceso de Llenado Masivo

3.1. Configuración de Importación

Se configuraron las instrucciones BULK INSERT en SQL Server con los siguientes parámetros para cada archivo CSV:

```
sql
Copy code
-- Ejemplo para la tabla Cat_TipoIdentificaciones desde CATID.csv
BULK INSERT Cat_TipoIdentificaciones
FROM 'C:\Users\Jose\Desktop\Excel\CATID.csv'
WITH
(
    FIELDTERMINATOR = ';',
    ROWTERMINATOR = '\n',
    FIRSTROW = 2  -- Opcional, si el archivo tiene encabezado
);
```

3.2. Ejecución del Proceso

Cada instrucción BULK INSERT se ejecutó dentro del contexto de la base de datos 'TallerCuero_final', asegurando que los datos se cargaran correctamente en las tablas correspondientes.

4. Resultados y Verificación

Se verificó que los datos importados estuvieran correctos mediante consultas SQL SELECT para cada tabla, asegurando la integridad y consistencia de los datos en la base de datos.

5. Conclusión

El proceso de llenado masivo de datos mediante BULK INSERT facilita la carga eficiente y rápida de grandes volúmenes de datos desde archivos CSV a la base de datos del taller de cuero 'Arte en Piel'. Esta metodología asegura que la información esencial para las operaciones diarias esté actualizada y sea precisa.

Matriz de responsabilidad

Rol	Responsabilidades
Planificación	Propietario: Define la visión y objetivos del taller. Supervisa el progreso y toma decisiones estratégicas. Gerente administrativo: Supervisa los recursos financieros y humanos. Asegura que se asignen recursos adecuados para la planificación.
Presupuesto	Propietario: Aprueba el presupuesto general del taller y los proyectos específicos. Gerente administrativo: Elabora y supervisa el presupuesto general del taller y los proyectos. Asegura que los gastos estén dentro de los límites establecidos. Contadores : Realiza seguimiento de los gastos y asegura la precisión contable.
Diseño	Propietario: Aprueba el presupuesto general del taller y los proyectos específicos. Diseñador jefe: Lidera el proceso de diseño de productos de cuero. Supervisa la creación de prototipos y la selección de materiales. Diseñadores de Producto : Trabaja en estrecha colaboración con el diseñador jefe para dar vida a los diseños.

Ventas	Propietario: Participa en la identificación de oportunidades de venta y desarrollo de estrategias de marketing. Gerente de Ventas : Lidera el equipo de ventas. Establece objetivos de ventas y estrategias para alcanzarlos. Representantes de Ventas : Realiza visitas a clientes potenciales y promueve los productos del taller.
Producción	Propietario: Supervisa y coordina todas las actividades de producción. Asegura que se cumplan los estándares de calidad y los plazos de entrega. Gerente de Producción: Lidera el equipo de producción. Coordina la asignación de tareas y recursos para optimizar la eficiencia y calidad de la producción. Artesanos : Trabaja en la fabricación de productos de cuero siguiendo las especificaciones y estándares establecidos.
Control de Calidad	Artesanos : Realiza controles de calidad en cada etapa del proceso de fabricación.

Proceso de Migración de Datos

1. Introducción

Este documento describe el proceso de migración de datos ejecutado en la base de datos 'TallerCuero_final' para el taller de cuero 'Arte en Piel'. El proceso incluye la inserción de datos en diversas tablas para establecer la estructura inicial necesaria para las operaciones del negocio.

2. Objetivo

El objetivo principal de este proceso de migración es poblar la base de datos con información esencial sobre países, departamentos, municipios, empleados, sucursales, productos y otros elementos cruciales para el funcionamiento del taller.

3. Detalles del Proceso

A continuación se detalla la inserción de datos en cada tabla relevante, siguiendo un orden lógico de acuerdo con las relaciones establecidas en el esquema de la base de datos.

3.1. Tabla Cat_Paises

Se insertaron datos de países relevantes para la operación:

```
sql
Copy code
insert into Cat_Paises (id_pais, nombre, abreviatura,
codigoEntrada, nacionalidad)
values
(1, 'Nicaragua', 'NIC', '+505', 'Nicaragüense'),
(2, 'Costa Rica', 'CRC', '+506', 'Costarricense');
```

3.2. Tabla Departamentos

Se insertaron datos de departamentos correspondientes a los países:

```
sql
Copy code
insert into Departamentos (id_departamento, departamento,
abreviatura, pais)
values
(1, 'Managua', 'MGA', 1),
(2, 'San José', 'SJO', 2);
```

3.3. Tabla Municipios

Se insertaron datos de municipios bajo cada departamento:

```
sql
Copy code
insert into Municipios (id_municipio, municipio, abreviatura,
departamento)
values
(1, 'Managua', 'MGA', 1),
```

```
(2, 'Escazú', 'ESC', 2);
```

3.4. Tabla Ubicaciones

Se insertaron datos de ubicaciones físicas:

```
sql
Copy code
insert into Ubicaciones (id_ubicacion, direccion, municipio,
codigoPostal, calle)
values
(1, 'Avenida Bolivar', 1, 10000, 'Calle 1'),
(2, 'Barrio Escalante', 2, 20000, 'Calle 2');
```

3.5. Otras Tablas Principales

Se realizaron inserciones similares en otras tablas fundamentales como:

- **Areas, Cargos, Sucursales, Cat_TipoIdentificaciones, Identificaciones, Cat_Generos, Empleados**: Para definir áreas funcionales, cargos, sucursales, tipos de identificación, datos personales de empleados y géneros.
- **Cat_Telefonos, Empleados_Telefonos**: Para manejar información de contacto y asignación de teléfonos a empleados.
- **DetallesEmpleado, DetallesSucursales**: Para detalles específicos de empleados y sucursales.

4. Verificación de Datos

Después de cada inserción, se verificó la consistencia y validez de los datos mediante consultas SELECT para asegurar que los registros se hubieran añadido correctamente.

5. Conclusiones

El proceso de migración de datos ha sido completado exitosamente, estableciendo una base sólida de información para las operaciones diarias del taller de cuero 'Arte en Piel'. Este documento sirve como referencia para futuras consultas y mantenimiento del sistema.

Plan de mantenimiento y respaldo de la base de datos

INFORME DEL Mantenimiento		
Mantenimiento de: Base de datos Taller de cuero		
Tipo de respaldo: Respaldo Total Taller.		
Periodo: Semanal	**Hora Inicio:** 1:00 pm	**Hora Final:** 1:30pm
Responsable del Mantenimiento:	**Agente SQL Server**	
Responsable del Supervisión:	**IT Infrastructure Manager**	
Dia: Sábado		
Descripción de Mantenimiento		
1.Recompilar Índice **2.Reorganizar índice** **3.Comprobar Integridad De la base de datos** **4.Copia de seguridad de la base de datos (Total)**		
Descripción de la información a respaldar:		
El respaldo de la base de datos del taller de cuero incluiría información detallada sobre clientes, preferencias de productos, compras, ventas, transacciones y registro de producción y abastecimiento de bodegas.		

INFORME DEL Mantenimiento		
Mantenimiento de: Base de datos Taller de cuero		
Tipo de respaldo: Respaldo Diferencial Taller.		
Periodo: Diario	**Hora Inicio:** 5:00 pm	**Hora Final:** 5:30pm
Responsable del Mantenimiento:	**Agente SQL Server**	
Responsable del Supervisión:	**IT Infrastructure Manager**	
Dia: Todos		
Descripción de Mantenimiento		
1.Recompilar Índice **2.Reorganizar índice** **3.Comprobar Integridad De la base de datos** **4.Copia de seguridad de la base de datos (Diferencial)**		
Descripción de la información a respaldar:		
El respaldo de la base de datos del taller de cuero diferencial incluiría información detallada sobre clientes, preferencias de productos, compras, ventas, transacciones y registro de producción y abastecimiento de bodegas.		

INFORME DEL Mantenimiento		
Mantenimiento de: Base de datos Taller de cuero		
Tipo de respaldo: Respaldo **transaccional** Taller.		
Periodo: 6 Horas	**Hora Inicio:** 8:00 pm	**Hora Final:** 6:30pm
Responsable del Mantenimiento:	**Agente SQL Server**	
Responsable del Supervisión:	**IT Infrastructure Manager**	
Dia: todos		
Descripción de Mantenimiento		
1.Comprobar Integridad De la base de datos **2.Copia de seguridad de la base de datos (transaccional)**		
Descripción de la información a respaldar:		
El respaldo de la base de datos del taller de cuero incluiría información detallada sobre compras, ventas, transacciones.		

Procedimientos Almacenados y su Importancia

Introducción

Los procedimientos almacenados son componentes fundamentales en sistemas de gestión de bases de datos, incluyendo el diseño y mantenimiento de bases de datos para aplicaciones comerciales como el taller de cuero "Arte en Piel". Estos procedimientos ofrecen una forma estructurada y eficiente de ejecutar operaciones dentro de la base de datos, proporcionando beneficios significativos en términos de rendimiento, seguridad y mantenibilidad del sistema.

¿Qué son los Procedimientos Almacenados?

Los procedimientos almacenados son conjuntos de instrucciones SQL precompiladas y almacenadas en la base de datos. Se utilizan para realizar tareas específicas, como insertar, actualizar, eliminar y recuperar datos, así como para

ejecutar lógica de negocio compleja. Cada procedimiento almacenado puede aceptar parámetros de entrada y devolver resultados a través de parámetros de salida o conjuntos de resultados.

Importancia de los Procedimientos Almacenados

1. **Optimización del Rendimiento:**
    - **Reducción de la carga de red:** Al ejecutar lógica en el servidor de base de datos, se minimiza el tráfico de red, mejorando la velocidad de ejecución de las consultas.
    - **Plan de ejecución optimizado:** Los procedimientos almacenados son precompilados y almacenados en caché, lo que optimiza el plan de ejecución de las consultas y mejora el tiempo de respuesta de las aplicaciones.
2. **Seguridad:**
    - **Control de acceso:** Los procedimientos almacenados permiten controlar quién puede ejecutar qué operaciones en la base de datos, reduciendo riesgos de seguridad al limitar el acceso directo a las tablas.
3. **Consistencia de Datos:**
    - **Integridad de los datos:** Al centralizar la lógica de negocio en procedimientos almacenados, se asegura que las operaciones sobre los datos mantengan la integridad y consistencia de la base de datos.
4. **Facilidad de Mantenimiento:**
    - **Modularidad:** Separar la lógica de negocio en procedimientos almacenados facilita el mantenimiento y la evolución del sistema, permitiendo cambios en la implementación sin afectar la aplicación cliente.
    - **Reutilización de código:** Los procedimientos almacenados pueden ser compartidos y reutilizados por múltiples aplicaciones, promoviendo una implementación coherente y reduciendo la duplicación de código.
5. **Mejora en la Administración del Sistema:**
    - **Monitoreo y diagnóstico:** Al utilizar procedimientos almacenados para tareas administrativas recurrentes (como respaldos, auditorías), se simplifica la administración del sistema y se facilita el monitoreo del rendimiento y la salud de la base de datos.

Conclusión

En resumen, los procedimientos almacenados son esenciales para el diseño de sistemas robustos de gestión de bases de datos como el utilizado en "Arte en Piel". Proporcionan beneficios significativos en términos de rendimiento, seguridad, mantenibilidad y administración del sistema, contribuyendo a la eficiencia operativa y la toma de decisiones informadas basadas en datos precisos y consistentes. Integrar y utilizar adecuadamente procedimientos almacenados en la arquitectura de la base de datos es clave para optimizar el funcionamiento y la eficacia de cualquier aplicación empresarial.

Para una documentación detallada sobre la configuración de seguridad a nivel de servidor en SQL Server, especialmente enfocada en logins, usuarios y permisos, puedes estructurarla de la siguiente manera:

Documentación de Seguridad a Nivel de Servidor en SQL Server

Introducción

Esta documentación describe los procedimientos y mejores prácticas para gestionar la seguridad a nivel de servidor en SQL Server. Se centra en la creación de logins, asignación de usuarios a bases de datos y la gestión de permisos para diferentes roles dentro de la organización.

Objetivos

- Establecer una política de seguridad robusta que garantice el acceso controlado a los datos sensibles.
- Asignar permisos adecuados a los usuarios según sus roles y responsabilidades.
- Documentar los pasos necesarios para crear logins, usuarios y asignar permisos en SQL Server.

Configuración de Logins

Creación de Logins

Los logins son las credenciales de acceso al servidor SQL Server. Cada login está asociado a un principal de seguridad de SQL Server.

Ejemplo:

```
sql
Copy code
USE [master]
GO

CREATE LOGIN [Gerente administrativo] WITH PASSWORD=N'1234',
DEFAULT_DATABASE=[tallercuero], CHECK_EXPIRATION=OFF,
CHECK_POLICY=OFF;
-- Crear otros logins según los roles necesarios
```

Configuración de Usuarios

Asignación de Usuarios a Bases de Datos

Los usuarios son asociados a logins y tienen permisos específicos en bases de datos particulares.

Ejemplo:

```
sql
Copy code
USE [tallercuero]
GO

CREATE USER [Gerente administrativo] FOR LOGIN [Gerente
administrativo];
-- Asignar otros usuarios según los logins creados
```

Asignación de Permisos

Gestión de Permisos

Los permisos definen las operaciones que los usuarios pueden realizar en objetos específicos dentro de la base de datos.

Ejemplo:

```
sql
Copy code
USE [tallercuero]
GO

GRANT SELECT, UPDATE, INSERT, DELETE ON [dbo].[UsuariosGrupos]
TO [Gerente administrativo];
-- Otorgar permisos según las necesidades específicas de cada
rol
```

Mantenimiento y Auditoría

Consideraciones de Seguridad

- **Contraseñas:** Asegurarse de que las contraseñas sean robustas y cumplan con las políticas de seguridad.
- **Auditoría:** Revisar regularmente los permisos y realizar auditorías de seguridad para detectar posibles vulnerabilidades.

Conclusiones

Esta documentación proporciona una guía básica para la configuración de seguridad a nivel de servidor en SQL Server. Se recomienda revisar y actualizar periódicamente los permisos y políticas de seguridad para mantener un entorno seguro y protegido.

El objetivo principal es establecer logins seguros que permitan a diferentes usuarios acceder a la base de datos 'tallercuero' según sus roles y responsabilidades dentro del taller de cuero.

Proceso de Creación de Logins

6. **Gerente Administrativo**
    - **Login:** Gerente administrativo
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** El gerente administrativo necesita acceso completo a la base de datos para gestionar operaciones, personal y recursos financieros.
7. **Diseñador Jefe**
    - **Login:** Diseñador Jefe
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** El diseñador jefe requiere acceso para supervisar y gestionar el diseño de productos y proyectos.
8. **Gerente de Ventas**
    - **Login:** Gerente de ventas
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** El gerente de ventas necesita acceso para monitorear las ventas, las estrategias de marketing y la relación con los clientes.
9. **Gerente de Producción**
    - **Login:** Gerente producción
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** El gerente de producción requiere acceso para supervisar la fabricación de productos, gestión de inventarios y planificación de la producción.
10. **Contadores**
    - **Login:** Contadores
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** Los contadores necesitan acceso para gestionar las transacciones financieras, contabilidad y auditorías.
11. **Diseñadores de Productos**
    - **Login:** Diseñadores de productos
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** Los diseñadores de productos necesitan acceso para trabajar en el diseño y desarrollo de nuevos productos.
12. **Representantes de Ventas**
    - **Login:** Representantes de ventas
    - **Password:** 1234

- **Base de datos predeterminada:** tallercuero
- **Razón:** Los representantes de ventas necesitan acceso para gestionar pedidos, seguimiento de ventas y atención al cliente.

13. **Artesanos**
    - **Login:** Artesanos
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** Los artesanos necesitan acceso para registrar su trabajo, actualizar el estado de las órdenes y colaborar en la producción de productos.
14. **Ayudantes de Producción**
    - **Login:** Ayudantes de producción
    - **Password:** 1234
    - **Base de datos predeterminada:** tallercuero
    - **Razón:** Los ayudantes de producción necesitan acceso para asistir en las tareas de fabricación y cumplimiento de órdenes.
15. **Otros Logins Específicos**
    - **Logins adicionales:** resventa1, resventa2, Vendedor1, Vendedor2, Vendedor3, Vendedor4, Caja1, Caja2, Bodega1
    - **Razón:** Estos logins adicionales se crearon para permitir acceso específico a usuarios como vendedores, cajas registradoras y personal de bodega según las necesidades operativas del taller.

Conclusiones La creación de logins con contraseñas seguras y políticas de seguridad adecuadas es esencial para garantizar la integridad y la confidencialidad de los datos en la base de datos 'tallercuero'. Cada login se ha configurado para proporcionar acceso adecuado a los recursos necesarios según los roles y responsabilidades del usuario dentro del taller de cuero.

Recomendaciones

- Monitorear regularmente los accesos y actividades de los usuarios para detectar cualquier actividad inusual o potencial violación de seguridad.
- Implementar políticas de contraseña sólidas y realizar auditorías periódicas de seguridad para mantener la integridad del sistema.

Este informe proporciona una visión general detallada del proceso de creación de logins en la base de datos 'tallercuero', asegurando que cada usuario tenga acceso apropiado y seguro a los recursos necesarios para realizar sus funciones dentro del taller.

Creación de Usuario y Asignación de Permisos para Vendedores

Introducción En este documento se detalla el proceso de creación de un usuario específico para los vendedores en la base de datos 'tallercuero', junto con la asignación de permisos necesarios para realizar sus funciones dentro del sistema.

Proceso de Creación de Usuario y Asignación de Permisos

Creación del Usuario

Usuario: Vendedores

Login Asociado: Vendedor1

Base de datos: tallercuero

Razón: El usuario 'Vendedores' se crea para permitir que los vendedores accedan a la base de datos utilizando el login 'Vendedor1', facilitando así el seguimiento y la gestión de sus actividades de venta.

sql

Copy code

```
USE [tallercuero]
GO

CREATE USER [Vendedores] FOR LOGIN [Vendedor1]
GO
```

Asignación de Permisos

Permisos de Actualización en venta:

Razón: Los vendedores necesitan poder actualizar registros de ventas para reflejar transacciones y cambios en el sistema.

sql

Copy code

```
USE [tallercuero]
GO
```

```
GRANT UPDATE ON [dbo].[venta] TO [Vendedores]
GO
```

Permisos de Inserción en venta:

Razón: Los vendedores deben poder registrar nuevas ventas en el sistema.

sql

Copy code

```
USE [tallercuero]
GO

GRANT INSERT ON [dbo].[venta] TO [Vendedores]
GO
```

Permisos de Actualización en Cajas:

Razón: Los vendedores necesitan actualizar el estado de las cajas según las transacciones realizadas.

sql

Copy code

```
USE [tallercuero]
GO

GRANT UPDATE ON [dbo].[Cajas] TO [Vendedores]
GO
```

Permisos de Inserción en Cajas:

Razón: Los vendedores deben poder registrar nuevas transacciones de caja en el sistema.

sql

Copy code

```
USE [tallercuero]
GO

GRANT INSERT ON [dbo].[Cajas] TO [Vendedores]
GO
```

Permisos de Actualización en detalleventa:

Razón: Los vendedores necesitan actualizar detalles específicos de las ventas para mantener registros precisos y actualizados.

sql

Copy code

```
USE [tallercuero]
GO

GRANT UPDATE ON [dbo].[detalleventa] TO [Vendedores]
GO
```

Permisos de Inserción en detalleventa:

Razón: Los vendedores deben poder agregar nuevos detalles de ventas relacionados con las transacciones.

sql

Copy code

```
USE [tallercuero]
GO

GRANT INSERT ON [dbo].[detalleventa] TO [Vendedores]
GO
```

Conclusiones La creación del usuario 'Vendedores' y la asignación de permisos específicos en la base de datos 'tallercuero' aseguran que los vendedores tengan acceso adecuado para cumplir con sus responsabilidades de registro y gestión de ventas. Estos permisos están diseñados para equilibrar la seguridad con la funcionalidad necesaria para el equipo de ventas.

Recomendaciones

Monitorear regularmente el uso de permisos y actividades de los usuarios para mantener la integridad y seguridad de la base de datos.

Revisar periódicamente los permisos asignados para asegurarse de que sigan siendo adecuados y necesarios para las funciones de cada usuario.

Este documento proporciona una guía completa sobre cómo se implementa y justifica la creación de usuarios y asignación de permisos específicos para los vendedores en la base de datos 'tallercuero'.

Triggers en la Base de Datos 'tallercuero'

1. Trigger `tr_AfterInsert_HistorialSesiones` en la tabla `HistorialSesiones`

Descripción: Este trigger se dispara después de que se inserta una nueva fila en la tabla `HistorialSesiones`. Su objetivo es imprimir los detalles de la sesión recién insertada.

Código:

```
sql
Copy code
CREATE TRIGGER tr_AfterInsert_HistorialSesiones
ON HistorialSesiones
AFTER INSERT
AS
BEGIN
    DECLARE @id_session INT, @Usuario INT, @FechaInicio
DATETIME, @FechaSalida DATETIME, @IPOrigen NVARCHAR(45),
@Resultado NVARCHAR(20);

    SELECT @id_session = id_Session, @Usuario = Usuario,
@FechaInicio = FechaInicio, @FechaSalida = FechaSalida,
@IPOrigen = IPOrigen, @Resultado = Resultado
    FROM inserted;

    PRINT 'Se insertaron los siguientes datos en la tabla
HistorialSesiones:';
    PRINT 'ID de Sesión: ' + CAST(@id_session AS NVARCHAR(10));
```

```
    PRINT 'Usuario: ' + CAST(@Usuario AS NVARCHAR(10));
    PRINT 'Fecha de Inicio: ' + CONVERT(NVARCHAR(30),
@FechaInicio, 120);
    PRINT 'Fecha de Salida: ' + ISNULL(CONVERT(NVARCHAR(30),
@FechaSalida, 120), 'N/A');
    PRINT 'IP de Origen: ' + @IPOrigen;
    PRINT 'Resultado: ' + @Resultado;
END;
```

Justificación: Este trigger proporciona un registro detallado de cada sesión iniciada en el sistema, capturando información crucial como el usuario, las fechas de inicio y salida, la dirección IP de origen y el resultado de la sesión. Esto es útil para auditorías de seguridad y seguimiento de actividades.

2. Trigger `tr_AfterInsert_Usuarios` en la tabla `Usuarios`

Descripción: Este trigger se activa después de insertar datos en la tabla `Usuarios`. Su función es imprimir un mensaje indicando que se han insertado datos en la tabla.

Código:

```
sql
Copy code
CREATE TRIGGER tr_AfterInsert_Usuarios
ON Usuarios
AFTER INSERT
AS
BEGIN
    PRINT 'Se han insertado datos en la tabla Usuario.';
END;
```

Justificación: Este trigger es útil para registrar cada inserción de datos en la tabla de usuarios, lo cual es útil para seguimiento y auditoría de cambios en los registros de usuarios del sistema.

3. Trigger `ActualizarInventarioDespuesDeVenta` en la tabla `detalleventa`

Descripción: Este trigger se ejecuta después de que se inserta una nueva fila en la tabla `detalleventa`. Su objetivo es actualizar automáticamente el inventario disminuyendo la cantidad de productos vendidos.

Código:

```
sql
Copy code
CREATE TRIGGER ActualizarInventarioDespuesDeVenta
ON detalleventa
AFTER INSERT
AS
BEGIN
    DECLARE @id_venta INT;
    DECLARE @id_producto INT;
    DECLARE @cantidad_vendida INT;

    -- Cursor para iterar sobre cada fila insertada
    DECLARE inserted_cursor CURSOR FOR
    SELECT id_venta, id_producto, cantidad_vendida
    FROM inserted;

    OPEN inserted_cursor;

    FETCH NEXT FROM inserted_cursor INTO @id_venta,
@id_producto, @cantidad_vendida;

    WHILE @@FETCH_STATUS = 0
    BEGIN
        -- Actualizar la cantidad en el inventario solo si es un
producto
        UPDATE Inventario
        SET cantidad = cantidad - @cantidad_vendida
        WHERE id_entidad = @id_producto AND tipo_entidad =
'Producto';

        FETCH NEXT FROM inserted_cursor INTO @id_venta,
@id_producto, @cantidad_vendida;
    END

    CLOSE inserted_cursor;
    DEALLOCATE inserted_cursor;
```

```
END;
```

Justificación: Este trigger asegura que cada vez que se registra una venta, el inventario se actualiza automáticamente para reflejar la cantidad de productos vendidos, manteniendo así la precisión de las existencias.

4. Trigger `trg_InsertUpdatePrecios` en la tabla `Precios`

Descripción: Este trigger se activa después de que se inserta o actualiza un registro en la tabla `Precios`. Su función es insertar un nuevo registro en la tabla `HistorialPrecios` y actualizar el campo `id_historial_precio` en la tabla `Precios`.

Código:

```
sql
Copy code
CREATE TRIGGER trg_InsertUpdatePrecios
ON Precios
AFTER INSERT, UPDATE
AS
BEGIN
    DECLARE @id_precio INT;
    DECLARE @fechaInicio DATETIME;
    DECLARE @fechaFin DATETIME;
    DECLARE @precio DECIMAL(10, 2);
    DECLARE @tipoMoneda NVARCHAR(10);
    DECLARE @id_producto INT;

    SELECT @id_precio = id_precio,
           @fechaInicio = fechaInicio,
           @fechaFin = fechaFin,
           @precio = precio,
           @tipoMoneda = tipoMoneda
    FROM inserted;

    -- Insertar el nuevo registro en HistorialPrecios
    INSERT INTO HistorialPrecios (id_producto, precio,
fechaInicio, fechaFin, tipoMoneda)
    VALUES (@id_precio, @precio, @fechaInicio, NULL,
@tipoMoneda);
```

```
    -- Actualizar el campo id_historial_precio en Precios
    UPDATE Precios
    SET id_historial_precio = SCOPE_IDENTITY()
    WHERE id_precio = @id_precio;
END;
```

Justificación: Este trigger asegura que cada vez que se actualiza o inserta un nuevo precio para un producto, se registra automáticamente en el historial de precios (`HistorialPrecios`). Esto proporciona un registro histórico completo de los cambios de precios a lo largo del tiempo, lo cual es esencial para análisis y auditoría.

Conclusiones Estos triggers proporcionan funcionalidades automáticas y registros importantes en la base de datos 'tallercuero', mejorando la gestión de sesiones, usuarios, inventario y precios. Cada uno está diseñado para optimizar la operación del sistema y asegurar la precisión y el seguimiento de datos críticos.

Cubos OLAP en SQL Server

¿Qué es un Cubo OLAP?

Un cubo OLAP (Procesamiento Analítico en Línea) es una estructura multidimensional que permite analizar grandes cantidades de datos de manera eficiente. Está diseñado para proporcionar una vista multidimensional de los datos, facilitando análisis complejos y consultas rápidas.

Propósito y Funcionalidad

16. **Análisis Multidimensional:** Los cubos OLAP permiten analizar datos desde múltiples perspectivas, incluyendo diferentes dimensiones y medidas. Esto es esencial para descubrir patrones, tendencias y

relaciones en los datos que no son fácilmente identificables en estructuras de datos tradicionales.

17. **Rendimiento Optimizado:** Al precalcular resúmenes y agregaciones, los cubos OLAP mejoran significativamente el rendimiento de las consultas analíticas complejas. Esto se logra mediante la creación de estructuras de datos precalculadas que permiten respuestas rápidas a consultas ad hoc y de alto nivel.
18. **Soporte para Grandes Volúmenes de Datos:** Ideal para entornos donde se manejan grandes volúmenes de datos históricos o transaccionales. Los cubos OLAP pueden manejar eficientemente millones de registros al proporcionar un acceso rápido y optimizado a los datos agregados.
19. **Interactividad y Exploración de Datos:** Facilitan la exploración interactiva de los datos, permitiendo a los usuarios navegar a través de diferentes niveles de detalle y realizar análisis detallados sobre diversas dimensiones. Esto ayuda a los usuarios a comprender mejor el comportamiento de los datos y a tomar decisiones informadas.
20. **Compatibilidad con Herramientas de BI:** Integra fácilmente con herramientas de Business Intelligence (BI) como Microsoft Power BI, Tableau, y Excel. Esto permite a los usuarios crear informes, dashboards y visualizaciones basadas en datos provenientes de cubos OLAP, mejorando así la capacidad de generar insights y comunicar hallazgos clave.

Componentes Principales

Los cubos OLAP en SQL Server generalmente consisten en los siguientes componentes principales:

- **Dimensiones:** Son las categorías o ejes por los cuales se puede analizar y filtrar los datos. Ejemplos comunes incluyen tiempo, ubicación, producto, cliente, etc.
- **Medidas:** Son los valores numéricos o cuantitativos que se analizan, como ingresos, cantidad vendida, costo, etc.
- **Hechos:** Son las tablas en la base de datos que contienen las medidas numéricas.
- **Agregaciones:** Son los resúmenes precalculados de los datos que permiten un acceso más rápido y eficiente a las consultas.

Implementación en SQL Server

En SQL Server, los cubos OLAP se implementan típicamente utilizando SQL Server Analysis Services (SSAS). SSAS permite la creación, administración y despliegue

de cubos OLAP, proporcionando herramientas para definir dimensiones, medidas, jerarquías y relaciones entre datos.

Beneficios Clave

- **Mejora del rendimiento de consultas complejas.**
- **Análisis multidimensional eficiente.**
- **Soporte para grandes volúmenes de datos.**
- **Facilita la toma de decisiones estratégicas.**

Conclusión

Los cubos OLAP en SQL Server son fundamentales para transformar datos transaccionales en información estratégica, permitiendo a las organizaciones explorar y analizar sus datos de manera eficiente y efectiva. Su capacidad para proporcionar vistas multidimensionales y respuestas rápidas a consultas complejas los convierte en una herramienta poderosa para cualquier entorno empresarial que requiera análisis de datos avanzado.