MARMARA UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

CSE 3055 – DATABASE SYSTEMS

ITERATION 3 PROJECT REPORT

Hikmet TOPAK – 150121047

Efe YALIM – 150121074

Emirhan ÜNSAL – 150121011

Umut Çağlar ÇELİK – 150121003

1. Project Description

This database is designed to support the operations of a business specializing in the sale and management of various products such as plants, chemicals, and tools. It provides a comprehensive structure for managing key aspects of the business, including inventory tracking, employee management, customer relationships, sales processes, and product categorization.

1. Scope

The scope of this database and its associated triggers, views, and operations includes the management and tracking of sales, customer transactions, inventory levels, and product performance within a retail or e-commerce environment. This system primarily supports the following processes:

* 1. Included Processes
     1. Customer and Transaction Management
        + Updating customer balances based on payment status (triggered by changes in payment status).
        + Tracking of customer performance metrics, such as total transactions, average transaction value, and pending payments.
        + Handling customer debt and paid costs based on sales and payment statuses.
     2. Sales and Inventory Updates
        + Automatically updating the inventory when a sale is made (triggered by a new sale insertion).
        + Adjusting the stock levels after each product sale to maintain an accurate inventory.
        + Updating customer statistics, including sales and payment-related data, to ensure customer records reflect the latest transaction status.
     3. Product and Stock Monitoring
        + Generating alerts for low stock levels, allowing users to monitor products that are nearing their minimum stock levels.
        + Tracking products with an impending expiration date, especially for chemical goods, and alerting the system when such products are nearing expiry.
        + Identifying expired chemical products that need to be removed from stock.
     4. Reporting and Analytics
        + Viewing detailed customer performance data, including sales, payment status, and pending payments.
        + Providing a comprehensive view of customers with outstanding debts.
        + Displaying detailed reports on expired chemical products and products with low stock levels.
  2. Excluded Processes
     1. Order Cancellations and Returns
        + The system does not currently support the management or tracking of order cancellations, returns, or refunds. There are no triggers or views dedicated to these actions.
     2. Customer Account Management
        + While the system tracks customer transactions, it does not include processes related to customer account creation, deletion, or modification of personal information (e.g., updating customer contact information, deleting customer accounts).
     3. Advanced Inventory Management
        + The database does not support more complex inventory management operations such as product restocking, supplier management, or handling back orders.
     4. Dynamic Pricing or Discounts
        + The database does not incorporate features for managing dynamic pricing, promotional discounts, or price changes based on conditions such as customer loyalty or seasonality.
     5. Transaction Dispute Resolution
        + There are no features for handling disputes or conflicts related to transactions, including partial payments or customer complaints.

1. Data and Requirements Analysis for the Database and Business Processes

The database is designed to support core business processes related to sales management, inventory control, customer tracking, and product performance monitoring. The analysis of data and requirements focuses on identifying key entities, relationships, and constraints that ensure the system meets its intended purpose effectively.

* 1. Data Analysis
     1. Key Entities and Attributes
        + Customer: Includes essential information such as CustomerID, CustomerName, ContactInfo, PaidCost, Debt, NoOfSelledProduct, and TotalBalance. These attributes facilitate the tracking of customer transactions and balances.
        + Sale: Contains data like SaleID, CustomerID, ProductID, PaymentStatus, TotalAmount, and SaleDate. This entity forms the foundation for managing transactions and their financial implications.
        + Inventory: Tracks product quantities and stock status through attributes like ProductID, Quantity, MinStockLevel, and StockStatus. It ensures that inventory levels are monitored and managed effectively.
        + Product: Covers general product information such as ProductID, ProductType, and Price, supporting categorization and pricing.
        + ChemicalGood, Plant, Tool: Specialized subcategories of products with specific attributes like ExpirationDate, PlantType, and ToolName.
     2. Relationships
        + Customer and Sale: A one-to-many relationship exists, where each customer can have multiple sales. This relationship enables detailed tracking of customer activity.
        + Sale and Product: A many-to-one relationship links sales to products, ensuring product-specific data (e.g., pricing and type) is accessible for transaction analysis.
        + Product and Inventory: A one-to-one relationship maintains accurate stock levels for each product.
     3. Derived Data and Aggregates
        + Customer performance metrics, such as TotalTransactions, AverageTransactionValue, and PendingPayments, are calculated using aggregate functions in views.
        + Inventory alerts, including Low Stock and Upcoming Expiration, are derived from thresholds and conditions in the data.
  2. Requirements Analysis
     1. Functional Requirements
        + Triggers

-Automate updates to customer balances when payment statuses change.

-Adjust inventory levels and customer statistics upon the insertion of new sales records.

* + - * Views

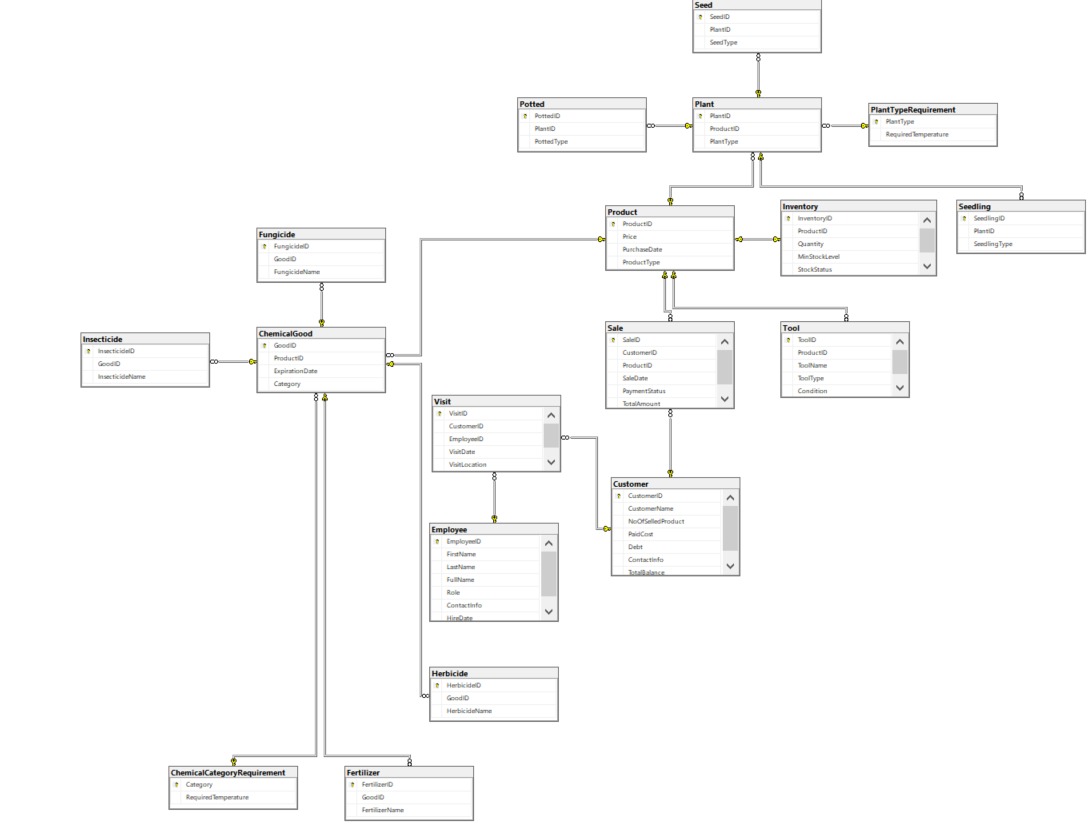
-Provide analytical insights into customer performance, customers with outstanding debts, and stock-related alerts.

-Generate reports on expired chemical products and inventory nearing critical thresholds.

* + 1. Non-Functional Requirements
       - Performance: Queries, especially those involving large datasets (e.g., inventory and sales records), must be executed efficiently.
       - Accuracy: Triggers and views must ensure data consistency across all operations (e.g., updates to balances and inventory).
       - Scalability: The database should accommodate an increasing number of customers, products, and transactions as the business grows.
    2. Constraints and Assumptions
       - Data Integrity: All triggers and views are designed to prevent anomalies, such as negative debt values or stock levels below zero.
       - Exclusions: Advanced features like order cancellations, dynamic pricing, and supplier management are beyond the current scope.
       - Business-Specific Rules: Debt and payment status management are tailored to align with the business's financial policies.
    3. Gaps and Potential Enhancements
       - While the system effectively supports sales and inventory processes, it does not currently manage supplier relationships or track promotional activities.
       - Integration with external systems, such as payment gateways or logistics platforms, could further enhance the database’s utility.

1. Diagram of Whole Database diyagram, metin, teknik çizim, plan içeren bir resim

   Açıklama otomatik olarak oluşturuldu



1. Tables
   1. Customer
      1. Fields/Columns
         * CustomerID
         * CustomerName
         * ContactInfo
         * NoOfSelledProduct
         * PaidCost
         * Debt
         * TotalBalance
      2. Definition: This table stores customer information, including personal details, financial statistics (such as PaidCost and Debt), and purchase activity.
      3. Data Types
         * CustomerID: INT
         * CustomerName: VARCHAR(100)
         * ContactInfo: VARCHAR(255)
         * NoOfSelledProduct: INT
         * PaidCost: DECIMAL(10, 2)
         * Debt: DECIMAL(10, 2)
         * TotalBalance: DECIMAL(10, 2)
      4. Indexes, Primary Key, Foreign Key
         * Primary Key: CustomerID
      5. Uniques, Identity, Constraints
         * CustomerID is an identity column, auto-incremented.
         * Check Constraint: Debt >= 0 to ensure no negative debt values.
      6. Triggers
         * Trigger trg\_UpdateCustomerBalance adjusts PaidCost and Debt based on changes in payment status in the Sale table.
   2. Sale
      1. Fields/Columns
         * SaleID
         * CustomerID
         * ProductID
         * TotalAmount
         * SaleDate
         * PaymentStatus
      2. Definition: Tracks individual sales transactions, including the customer involved, product purchased, total amount, and payment status.
      3. Data Types
         * SaleID: INT
         * CustomerID: INT
         * ProductID: INT
         * TotalAmount: DECIMAL(10, 2)
         * SaleDate: DATE
         * PaymentStatus: VARCHAR(20)
      4. Indexes, Primary Key, Foreign Key
         * Primary Key: SaleID
         * Foreign Key: CustomerID references Customer(CustomerID)
         * Foreign Key: ProductID references Product(ProductID)

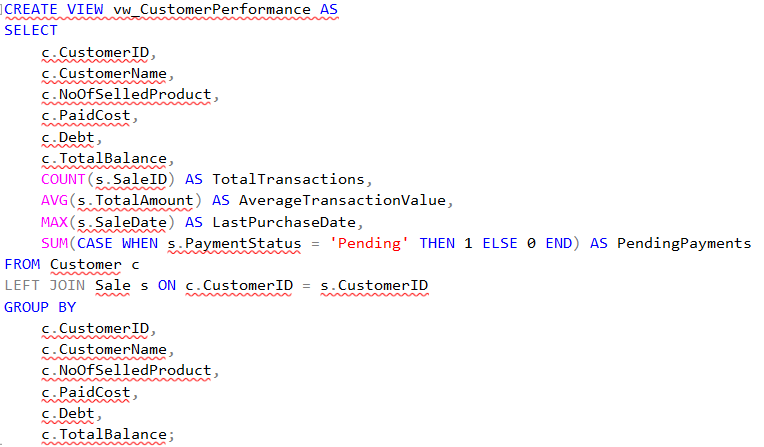
- Index: idx\_sale\_date on SaleDate

* + 1. Uniques, Identity, Constraints
       - SaleID is an identity column.
       - CHECK constraints:

-PaymentStatus allows only 'Pending', 'Completed', or 'Cancelled'.

- TotalAmount must be greater than 0.

* + 1. Triggers:
       - Trigger trg\_UpdateCustomerBalance updates customer statistics (PaidCost and Debt) when PaymentStatus changes to 'Completed'.
       - Trigger trg\_UpdateInventoryAfterSale adjusts inventory quantity and updates customer statistics after a sale is recorded.
  1. Product
     1. Fields/Columns:
        + ProductID
        + ProductType
        + Price
        + PurchaseDate
     2. Definition: This table stores information about products available for sale, including type and pricing details.
     3. Data Types:
        + ProductID: INT
        + ProductType: VARCHAR(50)
        + Price: DECIMAL(10, 2)
        + PurchaseDate: DATE
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: ProductID
        + Index: idx\_product\_type on ProductType
     5. Uniques, Identity, Constraints
        + ProductID is an identity column.
        + CHECK constraint on ProductType (Chemical, Tool, Plant)
        + CHECK constraint on Price (Price > 0)
     6. Triggers
        + None.
  2. PlantTypeRequirement
     1. Fields/Columns
        + PlantType
        + RequiredTemperature
     2. Definition: Stores plant type-specific temperature requirements.
     3. Data Types
        + PlantType: VARCHAR(50)
        + RequiredTemperature: DECIMAL(4, 1)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: PlantType
     5. Uniques, Identity, Constraints
        + CHECK constraint on PlantType to allow only 'Seed', 'Seedling', or 'Potted'.
        + RequiredTemperature has a default value of 20.0.
     6. Triggers:
        + None.
  3. ChemicalCategoryRequirement
     1. Fields/Columns
        + Category
        + RequiredTemperature
     2. Definition: Stores plant type-specific temperature requirements.
     3. Data Types
        + Category: VARCHAR(50)
        + RequiredTemperature: DECIMAL(4, 1)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: Category
     5. Uniques, Identity, Constraints
        + CHECK constraint on Category (Fertilizer, Insecticide, Fungicide, Herbicide)
        + DEFAULT value for RequiredTemperature (20.0)
     6. Triggers
        + None.
  4. Employee
     1. Fields/Columns
        + EmployeeID
        + FirstName
        + LastName
        + FullName
        + Role
        + ContactInfo
        + HireDate
        + Password
     2. Definition: This table stores details about employees, including roles, names, and contact information.
     3. Data Types
        + EmployeeID: INT
        + FirstName: VARCHAR(50)
        + LastName: VARCHAR(50)
        + FullName: Computed column (FirstName + ' ' + LastName)
        + Role: VARCHAR(50)
        + ContactInfo: VARCHAR(100)
        + HireDate: DATE
        + Password: VARCHAR(50)
        + Primary Key: Category
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: EmployeeID
     5. Uniques, Identity, Constraints
        + UNIQUE constraint on ContactInfo
        + CHECK constraint on Role (Manager, Salesperson, Consultant, Worker)
        + EmployeeID is an identity column.
        + Computed column: FullName (PERSISTED)
     6. Triggers
        + None.
  5. Plant
     1. Fields/Columns
        + PlantID
        + ProductID
        + PlantType
     2. Definition: Stores details about plants, including their product ID and plant type.
     3. Data Types
        + PlantID: INT
        + ProductID: INT
        + PlantType: VARCHAR(50)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: PlantID
        + Foreign Key: ProductID references Product(ProductID)
        + Foreign Key: PlantType references PlantTypeRequirement (PlantType)
     5. Uniques, Identity, Constraints
        + PlantID is an identity column.
     6. Triggers
        + None.
  6. Seed
     1. Fields/Columns
        + SeedID
        + PlantID
        + SeedType
     2. Definition: Stores information about seeds, linked to a specific plant.
     3. Data Types
        + SeedID: INT
        + PlantID: INT
        + SeedType: VARCHAR(50)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: SeedID
        + Foreign Key: PlantID references Plant(PlantID)
        + Index: idx\_seed\_type on SeedType
     5. Uniques, Identity, Constraints
        + SeedID is an identity column.
     6. Triggers
        + None.
  7. Seedling
     1. Fields/Columns
        + SeedlingID
        + PlantID
        + SeedlingType
     2. Definition: Stores information about seedlings, linked to a specific plant.
     3. Data Types
        + SeedlingID: INT
        + PlantID: INT
        + SeedlingType: VARCHAR(50)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: SeedlingID
        + Foreign Key: PlantID references Plant(PlantID)
     5. Uniques, Identity, Constraints
        + SeedlingID is an identity column.
     6. Triggers
        + None.
  8. Potted
     1. Fields/Columns
        + PottedID
        + PlantID
        + PottedType
     2. Definition: Stores information about potted plants, including their type and association with a plant record.
     3. Data Types
        + PottedID: INT
        + PlantID: INT
        + PottedType: VARCHAR(50)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: PottedID
        + Foreign Key: PlantID references Plant(PlantID)
     5. Uniques, Identity, Constraints
        + PottedID is an identity column.
     6. Triggers
        + None.
  9. ChemicalGood
     1. Fields/Columns
        + GoodID
        + ProductID
        + ExpirationDate
        + Category
     2. Definition: Stores details about chemical goods, including expiration dates and their categories.
     3. Data Types
        + GoodID: INT
        + ProductID: INT
        + ExpirationDate: DATE
        + Category: VARCHAR(50)
     4. Indexes, Primary Key, Foreign Key
        + Primary Key: GoodID
        + Foreign Key: ProductID references Product(ProductID)
        + Foreign Key: Category references ChemicalCategoryRequirement(Category)
        + Index: idx\_expiration on ExpirationDate
     5. Uniques, Identity, Constraints
        + GoodID is an identity column.
     6. Triggers
        + None.
  10. Fertilizer
      1. Fields/Columns
         + FertilizerID
         + GoodID
         + FertilizerName
      2. Definition: Stores information about fertilizers, including their names and associated chemical goods.
      3. Data Types
         + FertilizerID: INT
         + GoodID: INT
         + FertilizerName: VARCHAR(100)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: FertilizerID
         + Foreign Key: GoodID references ChemicalGood(GoodID)
      5. Uniques, Identity, Constraints
         + FertilizerID is an identity column.
         + UNIQUE constraint on FertilizerName
      6. Triggers
         + None.
  11. Fungicide
      1. Fields/Columns
         + FungicideID
         + GoodID
         + FungicideName
      2. Definition: Stores information about fungicides, including their names and associated chemical goods.
      3. Data Types
         + FungicideID: INT
         + GoodID: INT
         + FungicideName: VARCHAR(100)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: FungicideID
         + Foreign Key: GoodID references ChemicalGood(GoodID)
      5. Uniques, Identity, Constraints
         + FungicideID is an identity column.
         + UNIQUE constraint on FungicideName
      6. Triggers
         + None.
  12. Herbicide
      1. Fields/Columns
         + HerbicideID
         + GoodID
         + HerbicideName
      2. Definition: Stores information about herbicides, including their names and associated chemical goods.
      3. Data Types
         + HerbicideID: INT
         + GoodID: INT
         + HerbicideName: VARCHAR(100)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: HerbicideID
         + Foreign Key: GoodID references ChemicalGood(GoodID)
      5. Uniques, Identity, Constraints
         + HerbicideID is an identity column.
         + UNIQUE constraint on HerbicideName
      6. Triggers
         + None.
  13. Insecticide
      1. Fields/Columns
         + InsecticideID
         + GoodID
         + InsecticideName
      2. Definition: Stores information about insecticides, including their names and associated chemical goods.
      3. Data Types
         + InsecticideID: INT
         + GoodID: INT
         + InsecticideName: VARCHAR(100)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: InsecticideID
         + Foreign Key: GoodID references ChemicalGood(GoodID)
      5. Uniques, Identity, Constraints
         + InsecticideID is an identity column.
         + UNIQUE constraint on InsecticideName
      6. Triggers
         + None.
  14. Tool
      1. Fields/Columns
         + ToolID
         + ProductID
         + ToolName
         + ToolType
         + Condition
      2. Definition: Stores details about tools, including their names, types, and condition status.
      3. Data Types
         + ToolID: INT
         + ProductID: INT
         + ToolName: VARCHAR(100)
         + ToolType: VARCHAR(50)
         + Condition: VARCHAR(20)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: ToolID
         + Foreign Key: ProductID references Product(ProductID)
      5. Uniques, Identity, Constraints
         + ToolID is an identity column.
         + UNIQUE constraint on ToolName
         + CHECK constraint on Condition to allow only values: 'New', 'Good', 'Fair', 'Poor'
      6. Triggers
         + None.
  15. Inventory
      1. Fields/Columns
         + InventoryID
         + ProductID
         + Quantity
         + MinStockLevel
         + StockStatus
      2. Definition: Tracks inventory details, including product quantities and stock status based on minimum levels.
      3. Data Types
         + InventoryID: INT
         + ProductID: INT
         + Quantity: INT
         + MinStockLevel: INT
         + StockStatus: Computed Column (VARCHAR)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: InventoryID
         + Foreign Key: ProductID references Product(ProductID)
      5. Uniques, Identity, Constraints
         + InventoryID is an identity column.
         + CHECK constraint on Quantity to ensure non-negative values
         + StockStatus is a computed column: 'Low' if Quantity ≤ MinStockLevel, otherwise 'OK'.
      6. Triggers
         + None.
  16. Visit
      1. Fields/Columns
         + VisitID
         + CustomerID
         + EmployeeID
         + VisitDate
         + VisitLocation
      2. Definition: Records information about customer visits made by employees, including the date, location, and involved parties.
      3. Data Types
         + VisitID: INT
         + CustomerID: INT
         + EmployeeID: INT
         + VisitDate: DATE
         + VisitLocation: VARCHAR(100)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: VisitID
         + Foreign Key: CustomerID references Customer(CustomerID)
         + Foreign Key: EmployeeID references Employee(EmployeeID)
           1. Index: idx\_visit\_date on VisitDate
      5. Uniques, Identity, Constraints
         + VisitID is an identity column.
      6. Triggers
         + None.
  17. ChemicalCategoryRequirement
      1. Fields/Columns
         + Category
         + RequiredTemperature
      2. Definition: Defines temperature requirements for different chemical categories.
      3. Data Types
         + Category: VARCHAR(50)
         + RequiredTemperature: DECIMAL(4, 1)
      4. Indexes, Primary Key, Foreign Key
         + Primary Key: Category
      5. Uniques, Identity, Constraints
         + CHECK constraint on Category to allow only 'Fertilizer', 'Insecticide', 'Fungicide', or 'Herbicide'.
         + RequiredTemperature has a default value of 20.0.
      6. Triggers
         + None.

1. Views
   1. vw\_CustomerPerformance
      1. Definition: This view provides a detailed performance analysis of each customer. It includes customer details such as the total number of transactions, average transaction value, the date of the last purchase, and the number of pending payments. This view is designed to facilitate customer performance evaluations.
      2. Screenshots

Before:

* 1. vw\_CustomersWithDebt
     1. Definition**:** This view lists all customers who have outstanding debt. It displays customer information such as their name, contact details, paid cost, debt amount, and total balance. This view helps identify customers with financial obligations.
     2. Screenshots

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

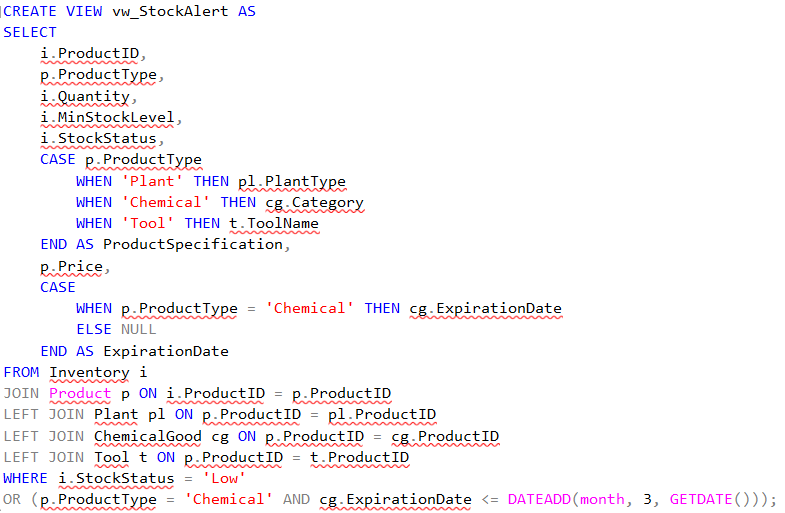
* 1. vw\_ExpiredChemicalProducts
     1. Definition**:** This view identifies expired chemical products in the inventory. It includes details such as the product ID, price, product type, expiration date, and category. This view ensures that expired items can be easily tracked and removed from stock.
     2. Screenshots

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

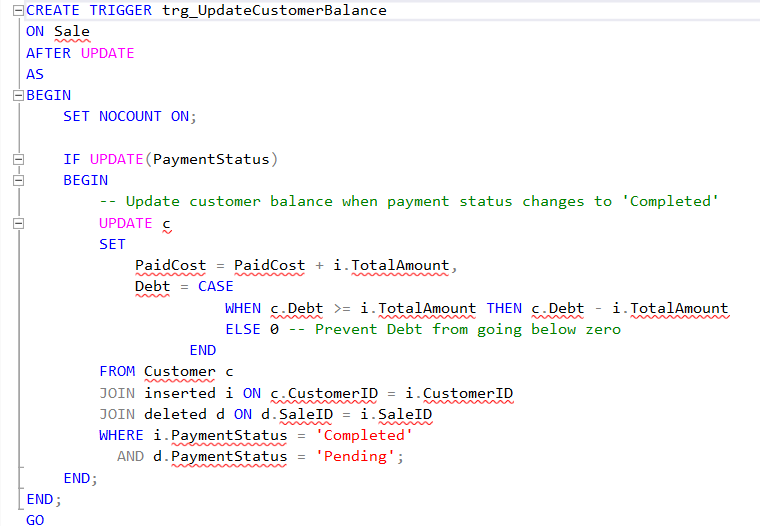
* 1. vw\_StockAlert
     1. Definition**:** This view highlights products in the inventory that are either low in stock or have a chemical expiration date approaching within three months. It provides details such as the product type, quantity, minimum stock level, stock status, product specification, price, and expiration date (if applicable). This view aids in managing inventory efficiently.
     2. Screenshots



1. Triggers
   1. trg\_UpdateCustomerBalance
      1. Definition

This trigger is executed after an update on the Sale table. It checks if the PaymentStatus column has been updated. If the PaymentStatus changes from 'Pending' to 'Completed', the customer's PaidCost and Debt fields are updated in the Customer table. The PaidCost increases by the TotalAmount of the sale, and the Debt is decreased accordingly, ensuring that it does not drop below zero.

* + 1. When/How it Works
       - Triggered after an update on the Sale table.
       - Specifically works when the PaymentStatus field is updated to 'Completed'.
    2. ScreenShots



Before:

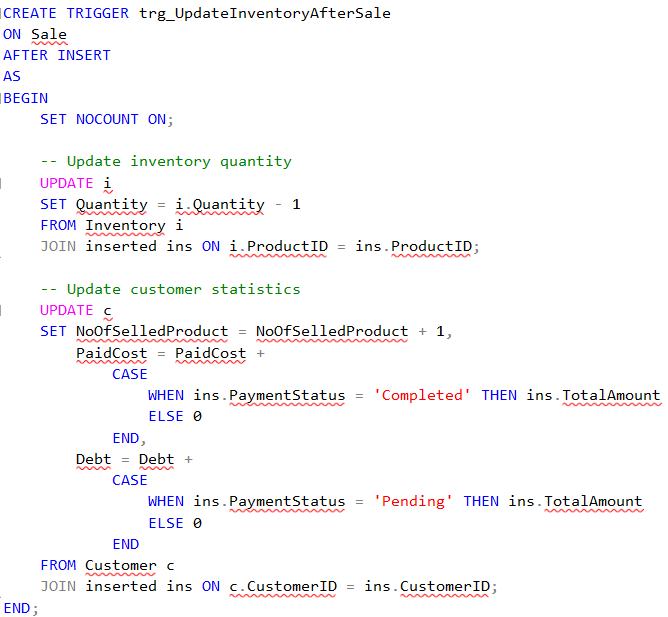
metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. trg\_UpdateInventoryAfterSale
     1. Definition: This trigger is executed after an insert on the Sale table.
        + Updates the Quantity in the Inventory table by reducing it by 1 for the corresponding product.
        + Updates customer statistics in the Customer table, such as incrementing NoOfSelledProduct, adjusting PaidCost, and increasing Debt if the payment status of the sale is 'Pending'.
     2. When/How it Works
        + Triggered after an insert on the Sale table.
        + Updates both inventory and customer statistics immediately after a new sale record is added.
     3. Screenshots

Before:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Usage:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Store Procedure
   1. AddNewSeed
      1. Definition: This stored procedure is used to add a new seed product to the database.
      2. Screenshots

metin, ekran görüntüsü, doküman, belge içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before: metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddCustomer
     1. Definition: This stored procedure is used to add a new customer to the Customer table.
     2. ScreenShots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewSeedling
     1. Definition: This stored procedure is designed to add a new seedling product to the database.
     2. Screenshotsmetin, ekran görüntüsü, doküman, belge içeren bir resim

        Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewPotted
     1. Definition**:** This procedure adds a new potted plant to the database by inserting its details into the Product, Plant, and Potted tables.
     2. Screenshotsmetin, ekran görüntüsü, doküman, belge içeren bir resim

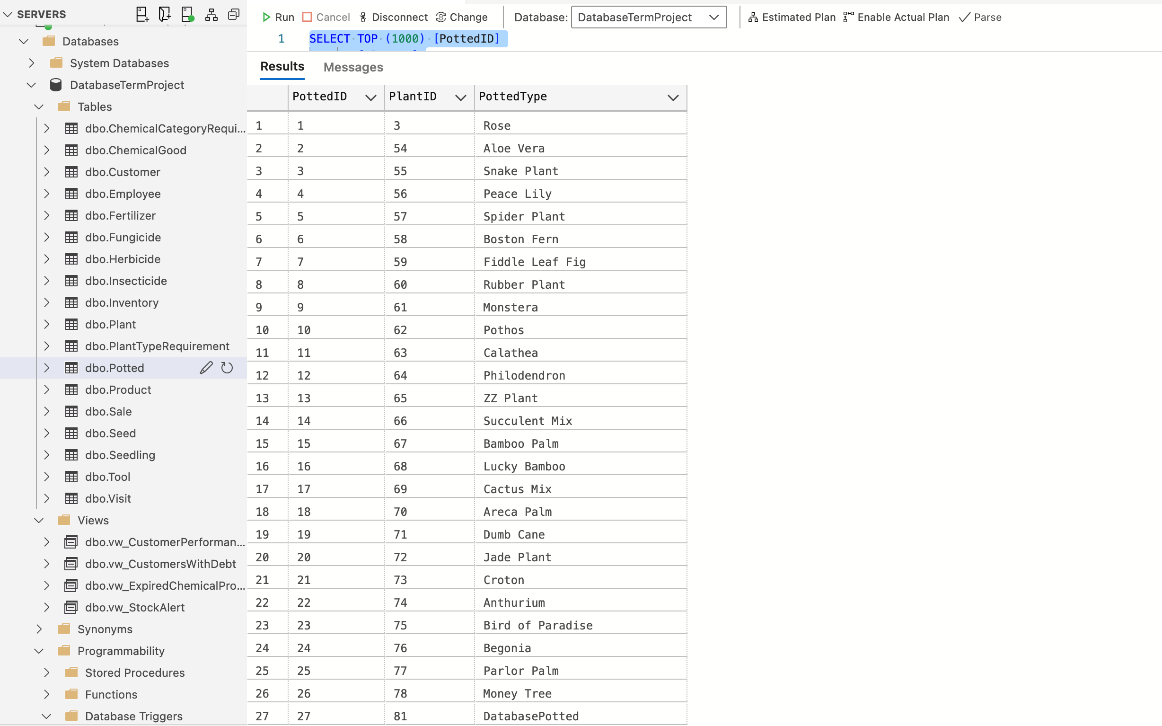
        Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:



* 1. AddNewFertilizer
     1. Definition**:** This procedure is used to add a new fertilizer product to the database. It inserts the fertilizer's information into the Product, ChemicalGood, and Fertilizer tables.
     2. Screenshots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewFungicide
     1. Definition**:** This procedure facilitates the addition of a new fungicide product to the database. It performs insertions into the Product, ChemicalGood, and Fungicide tables.
     2. Screenshots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewHerbicide
     1. Definition**:** This procedure facilitates the addition of a new herbicide product to the database.
     2. Screenshots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewInsecticide
     1. Definition**:** This procedure facilitates the addition of a new insecticide product to the database.
     2. Screenshotsmetin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

        Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewSale
     1. Definition**:** This procedure records a new sale in the database.
     2. Screenshots

metin, ekran görüntüsü, yazı tipi içeren bir resim

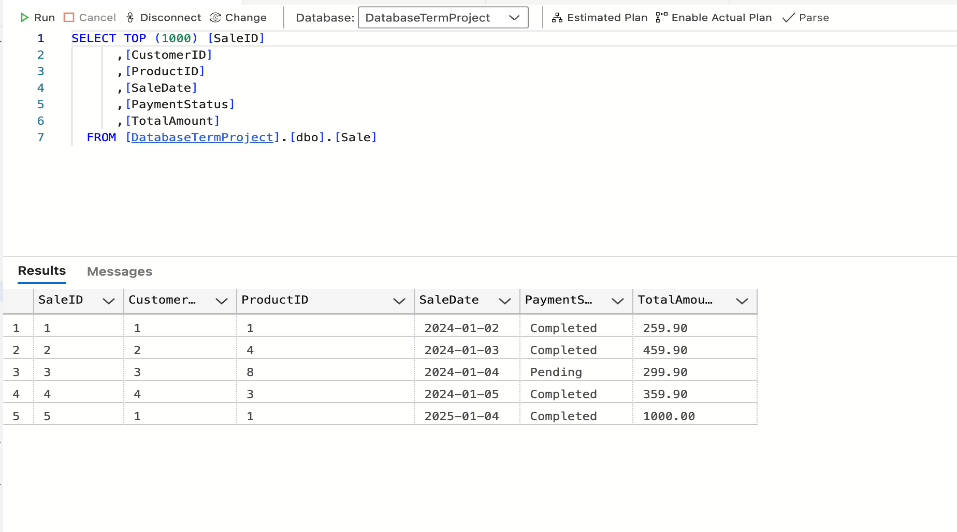
Açıklama otomatik olarak oluşturuldu

Before:

metin, sayı, numara, ekran görüntüsü, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:



* 1. AddNewTool
     1. Definition**:** This procedure adds a new tool to the database.
     2. Screenshots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewVisit
     1. Definition**:** This procedure allows recording a new customer visit.
     2. Screenshots

metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, çizgi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. AddNewEmployee
     1. Definition**:** This procedure facilitates the addition of a new employee to the system.
     2. Screenshots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. DeleteProduct
     1. Definition**:** This procedure deletes a product and all its related data from the database.
     2. Screenshots

metin, ekran görüntüsü, doküman, belge, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Before:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

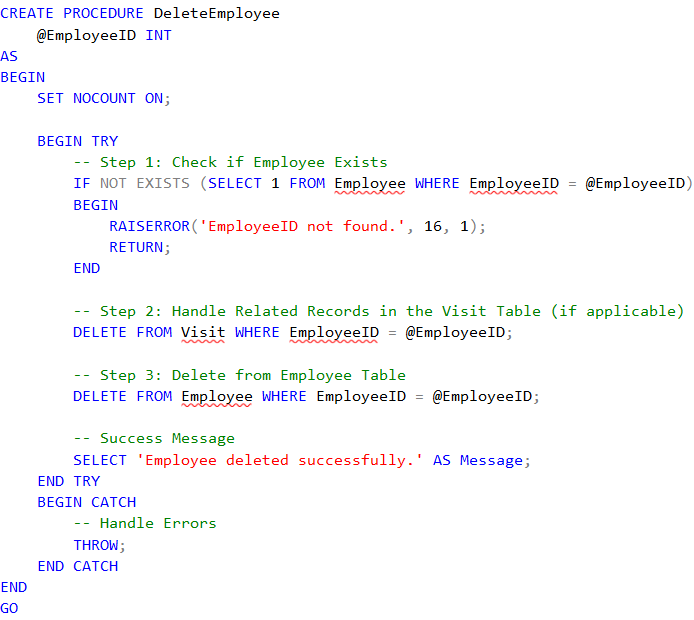
Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. DeleteEmployee
     1. Definition: This procedure deletes a employee and all its related data from the database.
     2. Screenshots



Before:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

After:

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu