

**THE DATA WIZARDS GROUP**

Prepared for

Enterprise Data Management

University of Arizona

Apurva Atalkar, Apoorva Barot, Priyal Dani, Utsav Parekh, Rohan Pudipeddi, Sneha Thomas

# TABLE OF CONTENTS

[TABLE OF CONTENTS 2](#_Toc152719853)

[Chapter 1: Requirements Analysis 3](#_Toc152719854)

[Chapter: Conceptual Schema 5](#_Toc152719855)

[Data Dictionary(Conceptual / for ER Modeling) 6](#_Toc152719856)

[Chapter 3: Relational Schema 15](#_Toc152719858)

[Relational Schema 15](#_Toc152719859)

[Data Dictionary (Relational) 18](#_Toc152719860)

[DDL Appendix 40](#_Toc152719900)

[Chapter 4: SQL Queries 49](#_Toc152719901)

[Chapter 6: User Interface 58](#_Toc152719904)

[Chapter 7: Implementation Plan 63](#_Toc152719905)

[Appendix A 64](#_Toc152719906)

[References 64](#_Toc152719907)

# Chapter 1: Requirements Analysis

In the context of Nestle, an effective data management framework is crucial to proficiently handle diverse facets of its supply chain and retail operations. Nestle, as a global leader in the food and beverage industry, confronts the complexity of managing a multifaceted supply chain and retail operations. The organization engages with various entities, each possessing distinctive attributes and intricate interrelationships, necessitating the development of a comprehensive data management framework.

At the core of Nestle's operations is supplier management, where suppliers play a crucial role in providing raw materials. For instance, suppliers like XYZ Farms (SupplierID: 001) contribute dairy products essential for Nestle's ice cream production. These suppliers are meticulously tracked, featuring details such as SupplierName, ContactName, ContactDesignation, Email, and Phone, along with their physical address details comprising Street, City, State, Country, and PostalCode.

The meticulous management of supply orders is exemplified by instances such as a particular SupplyDetailsId (S123) representing an order from ABC Ingredients Inc. These orders, featuring SupplyPrice, TotalValue, ItemName, and Quantity, are crucial for effective inventory management.These supply orders find their way into multiple inventories, exemplified by the SupplyInvID (SI456) associated with the "Main Warehouse." Here, inventory details like UnitsInStock, InventoryOwner (IOFirstName, IOLastName), and re-order levels are tracked for optimal product availability.

Raw materials supplied by these inventories are utilized by manufacturing units, such as UnitID 101, located at Nestle's facility in Cityville. Here, details like UnitName, location, production capacity, and facility size are maintained.Nestle's diverse product offerings are exemplified by the ProductId (P789) representing "Chocolate Crunch Bar." These products, identified by attributes like ProductName, Description, UnitPrice, Weight, and linked to ProductCategory, exemplify Nestle's commitment to efficient organization.

In managing these products, warehouses like WarehouseId 201, known as "Central Warehouse," play a pivotal role. Attributes like WarehouseName, Location, Capacity, UnitsInOrder, UnitsInStock, and Discontinued status are crucial for effective warehouse management. Shipments, such as ShipmentID (SH678) facilitated by CarrierName "Global Logistics," are integral to Nestle's logistics. ShipmentDetails, including tracking information, shipment weight, and arrival date, ensure effective monitoring.

Nestle's diverse workforce, exemplified by EmployeeId 301, includes various worker types, such as Packer, Shipping Clerk, and Supervisor. The WarehouseWorkerTeam, featuring an inventory manager, quality control inspectors, and other worker categories, ensures efficient warehouse operations. Departments within Nestle, like DepartmentId 401 representing "Marketing," are uniquely identified and managed with attributes like DepartmentName and DepartmentHead.

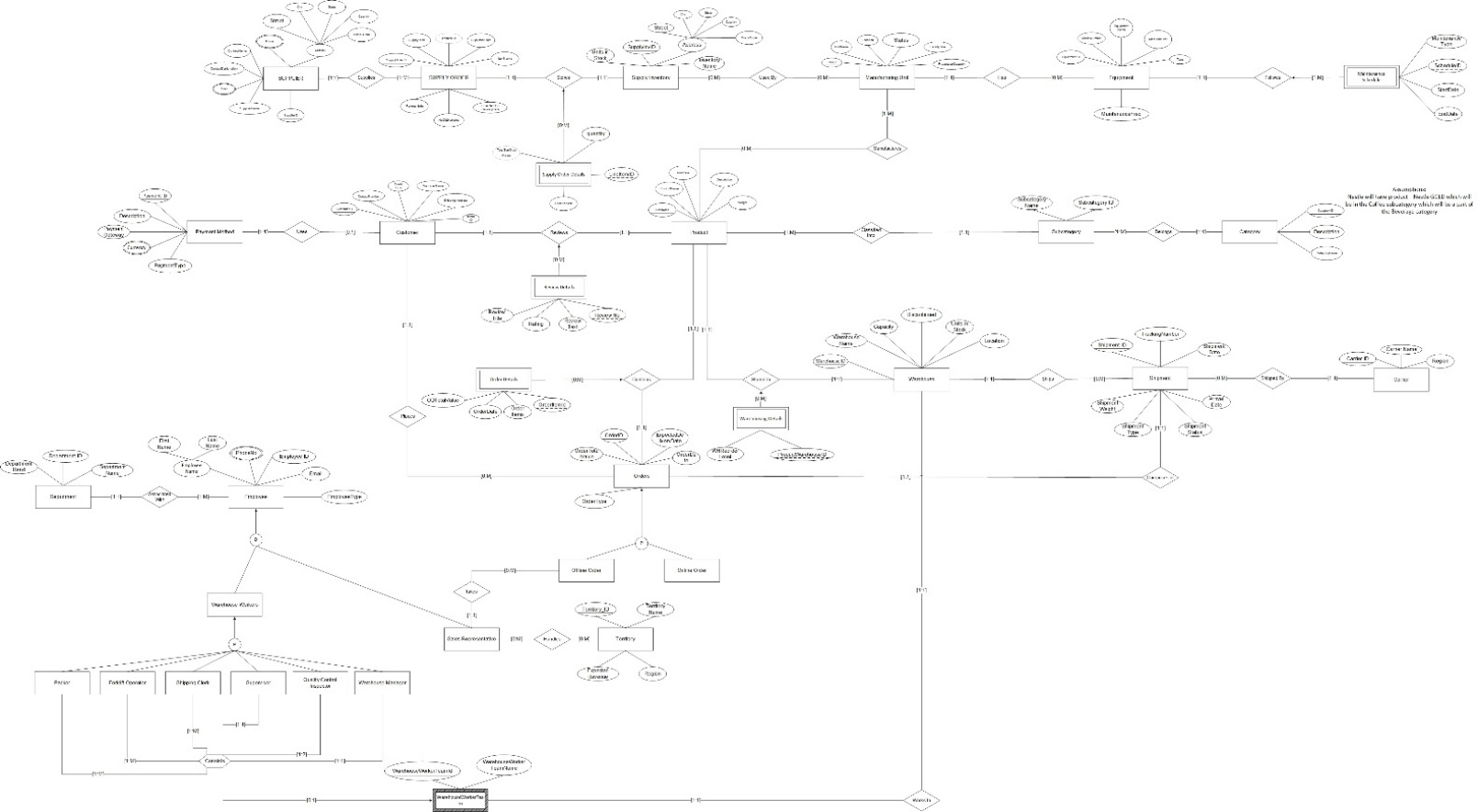
Sales representatives, like SalesRepId 501 associated with "North Region," are responsible for managing customer relationships, as seen in CustomerId 601 representing "ABC Retailers." Customer details, including CreditLimit and ShippingAddress, are stored alongside attributes like CustomerName and ContactNumber. Orders placed by customers, like OrderId 701, are uniquely identified, with offline orders managed by salespersons. Supply order details, such as SupplyOrderId 801, encompass critical attributes like PaymentInfo, PricePerUnit, and ExpectedDeliveryDate.

Customers utilize various payment methods, exemplified by PaymentID 901 for "Credit Card." Inventory and warehousing details, like InvReorderLevel and WhReorderLevel, ensure optimal inventory levels and reordering.Manufacturing units, identified by EquipmentId 1001, manage different equipment with attributes like MaintenanceSchedule ensuring timely maintenance for optimal production.

To fulfill the key requirement articulated by our client, Nestle, the focus is on ensuring a robust inventory of Nestle's diverse range of products at authorized retailers and points of sale. Therefore, a strategic implementation of a database system is imperative for Nestle to effectively track the movement of specific raw materials, store supplies, and finished products throughout our extensive supply chain. This database-driven approach is designed to empower Nestle's management to closely monitor material flow, promptly identify potential bottlenecks, and swiftly address any supply-related challenges. The primary aim is to automate the systematic tracking and storage of data at every critical juncture in Nestle's intricate supply chain, spanning from sourcing raw materials to delivering final products to retailers and distribution points. This comprehensive system ensures a streamlined and efficient supply chain management process, aligning with Nestle's commitment to maintaining optimal inventory levels and proactively managing supply chain dynamics with agility.

In addition to inventory management, a second critical requirement for Nestle is the implementation of a robust Database Management System (DBMS) and comprehensive tracking mechanisms to gain insights into sales performance and assess its impact across the entire supply chain. This entails the systematic recording and analysis of sales data, including product sales volumes, customer preferences, and regional variations in demand. The DBMS should enable real-time tracking of sales transactions, allowing Nestle's management to derive meaningful analytics and make data-driven decisions.

# Chapter 2: Conceptual Schema



## Data Dictionary (Conceptual / for ER Modeling)

1. Supplier

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| SupplierID | Unique value identifying supplier | Identifying attribute |
| SupplierName | Name of Supplier |  |
| ContactName | Point of Contact |  |
| ContactDesignation | Designation of Point of Contact |  |
| Email | Email of the supplier | Multivalued Attribute |
| Phone | Phone of the supplier | Multivalued Attribute |
| Address | Address of the supplier | Composite Attribute |
| Street | Street of the supplier |  |
| City | City of the supplier |  |
| State | State of the supplier |  |
| Country | Country of the supplier |  |
| PostalCode | PostalCode of the supplier |  |

1. SupplyOrder

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| SupplyorderID | Unique value identifying supply order | Identifying attribute |
| SupplyPrice | Supply Price in ($) |  |
| TotalValue | Total value of the suppliers order | Derived attribute from quantity and supply price |
| SupOrderDate | Date of the order |  |
| ItemName | Name of the item |  |
| PaymentInfo | Mode by which the payment was made |  |
| NoofLineitems | Number of each product purchased |  |

1. SupplyInventory

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| SupplyInvID | Unique value identifying supply inventory | Identifying attribute |
| InventoryName | InventoryName of the supply Inventory |  |
| Address | Address of the supply Inventory | Composite Attribute |
| Street | Street of the supply Inventory |  |
| City | City of the supply Inventory |  |
| State | State of the supply Inventory |  |
| Country | Country of the supply Inventory |  |
| PostalCode | PostalCode of the supply Inventory |  |
| UnitsInStock | Total Units in Inventory |  |

1. Product

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| ProductID | Unique value identifying products | Identifying attribute |
| ProductName | Name of the product |  |
| Description | Description of Product |  |
| UnitPrice | Unit Price in ($) |  |
| Weight | Weight in lb |  |

1. Category

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| CategoryID | Unique value identifying Category | Identifying attribute |
| Description | Description of Product Category |  |
| CategoryName | Name of the Category |  |

1. SubCategory

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| SubCategoryID | Unique value identifying subcategory | Identifying attribute |
| SubCategoryName | Name of the Category |  |

1. ReviewDetails

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| ReviewNo | Unique value identifying review details | Partial identifier |
| ReviewTitle | Title of the review |  |
| Rating | Rating of the review |  |
| ReviewText | Text of the review |  |

1. Warehouse

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| WarehouseId | Unique value identifying warehouse | Identifying attribute |
| WarehouseName | Name of the warehouse |  |
| Location | Location of the warehouse |  |
| Capacity | Capacity of the warehouse |  |
| UnitsInOrder | No. of units placed in order |  |
| Discontinued | Discontinued warehouse |  |

1. Shipment

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| ShipmentID | Unique value identifying shipment | Identifying attribute |
| TrackingNumber | Unique number for tracking |  |
| ShipmentDate | Date of Shipment |  |
| ShipmentStatus | Status of the Shipment |  |
| ShipmentType | Type of the Shipment |  |
| ShipmentWeight | Weight in lbs |  |
| ArrivalDate | Date of arrival of shipment |  |

1. Carrier

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| CarrierID | Unique value identifying carrier | Identifying attribute |
| CarrierName | Name of Carrier |  |
| Region | Region of Carrier |  |

1. MaintenanceSchedule

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| ScheduleID | Unique value identifying maintenance schedule | Identifying attribute |
| MaintenanceType |  |  |
| StartDate | Start date of maintenance schedule |  |
| EndDate | End date of maintenance schedule |  |

1. Employee

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| EmployeeID | Unique value identifying employee | Identifying attribute |
| EmployeeName | Full Name of Employee | Composite attribute |
| FirstName | First Name of the employee |  |
| LastName | Last Name of the employee |  |
| PhoneNo | Contact no. of employee | Multi-valued attribute |
| Email | Contact email of employee |  |
| EmployeeType | Type of Employee | E.g. Sales Representative, Warehouse Workers |

**Subclass of Employees (based on EmployeeType):**

1. Packer
2. Shipping Clerk
3. Forklift Operator
4. Supervisor
5. Quality Control Inspector
6. Inventory Manager
7. Sales Representative
8. WarehouseWorkerTeam

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| WarehouseWorkerTeamID | Unique value identifying worker team | Identifying attribute |
| WarehouseWorkerTeamName | Name of the Worker Team |  |

1. Department

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| DepartmentId | Unique value identifying Department | Identifying attribute |
| DepartmentName | Name of the department |  |
| DepartmentHead | Name of the department head |  |

1. PaymentMethod

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| PaymentID | Unique value identifying payment made | Identifying attribute |
| Description | Description related to Payment |  |
| PaymentGateway | Gateway Used |  |
| Currency | Currency of Transaction | Multi-valued attribute |
| PaymentType | Type of Payment | E.g, cash, credit, debit etc. |

1. Customer

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| CustomerID | Unique value identifying the customer | Identifying attribute |
| CustomerName | Name of Client |  |
| ContactNumber | Contact no. of client |  |
| CreditLimit | Credit Limit for each Client |  |
| ShippingAddress | Address of Client |  |
| EmailID | Email ID of Client |  |

1. Manufacturing Unit

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| UnitID | Unique value identifying the unit | Identifying attribute |
| UnitName | Name of Unit |  |
| Location | Geographical Location |  |
| Status | Information whether unit is currently active or not |  |
| ProductionCapacity | Total Capacity of unit |  |
| FacilitySize | Size in square meters |  |

1. Equipment

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| EquipmentID | Unique value identifying equipment | Identifying attribute |
| EquipmentName | Name of Equipment |  |
| Manufacturer | Name of Manufacturer |  |
| Type | Type of Equipment | e.g. mixer, grinder |
| MaintenanceFreq | How frequently the equipment should be checked for maintenance | e.g. biweekly, monthly |
| ModelNumber | Serial Number of Model of equipment |  |

1. Orders

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| OrderID | Order identifying id | Identifying attribute |
| OrderDate | Date when order was placed |  |
| OrderTotalValue | Total value of the order |  |
| OrderType | Type of order | e.g. |
| ExpectedDeliveryDate | Expected delivery date of the order |  |

**Subclass of orders**:

1. Offline Orders
2. Online Orders
3. Territory

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| TerritoryID | Territory identifying id managed by Sales Representative | Identifying attribute |
| TerritoryName | Name of Territory |  |
| Region | Region in which Territory falls |  |
| ExpectedRevenue | FY revenue anticipated |  |

1. SupplyOrderDetails

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| TotaSupOrderValue | Total value of the supply order($) | Partial identifier |
| PricePerUnit | Price of a unit in ($) |  |
| LineItemID | Individual item id in order |  |
| Quantity | Quantity of each item ordered |  |

1. OrderDetails

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| OrderItemId | Unique id to identify order | Partial identifier |
| ODTotalValue | Total order value of order details |  |
| OrderItems | List of items ordered |  |
| OrderDate | Order Date |  |

1. WarehousingDetails

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Construct Description** | **Other Information** |
| WhReorderLevel | Minimum threshold to restock units | Format: Integer |
| ProductWarehouseId | To identify product and warehouse combination | Partial Identifier |

### Relationships

|  |  |
| --- | --- |
| **Relation** | **Description** |
| Supplies | Relationship that models the supplier supplies supply order |
| Stores | Relationship that models the supply order that is stored in supply inventory |
| Used By | Relationship that models the supply inventory that is used by manufacturing unit |
| Has | Relationship that models the manufacturing unit which houses the equipment |
| Follows | Relationship that models the equipment following a maintenance schedule |
| Manufactures | Relationship that models the manufacturing unit created the product |
| Classifedinto | Relationship that models that product are classified into sub category |
| Belongs | Relationship that models that sub categories belong to certain product categories |
| Reviews | Relationship that models the customer reviews a product |
| Uses | Relationship that models the customer uses a payment method to make payments |
| Places | Relationship that models the customer placing an order |
| Contains | Relationship that models the order contains product |
| Stored in | Relationship that models that products are store in warehouses |
| Ships | Relationship that models the warehouse ships shipment |
| Shipped by | Relationship that models that shipments are shipped by carriers |
| Comprises of | Relationship that models that shipments contain order products placed by customers |
| Associated With | Relationship that models the employee is associated with a department |
| Takes | Relationship that models the sales representative takes offline orders |
| Handles | Relationship that shows that sales representatives manage certain territories |
| Consists | Relationship that models warehouse workers consists of packer, operator, shipping clerk, supervisor, quality control inspector, and inventory manager |
| Works in | Relationship that shows that the warehouse worker team works in warehouses |

# Chapter 3: Relational Schema

## Relational Schema

1. **SUPPLIER** (SupplierID, SupplierName, Street, City, State, Country, PostalCode, ContactName, ContactDesignation)

* 1. **SUPPLIER\_PHONE** (SupplierID, PhoneNo)

FOREIGN KEY (SupplierID) references **SUPPLIER** (SupplierID)

* 1. **SUPPLIER\_EMAIL** (SupplierID, Email)

FOREIGN KEY (SupplierID) references **SUPPLIER (**SupplierID)

2. **SUPPLYORDER** (SupplyorderID, SupplyPrice, TotalValue, SupOrderDate, ItemName, PaymentInfo, NoofLineitems, ExpectedSupDeliveryDate, SupplierID)

FOREIGN KEY (SupplierID) references **SUPPLIER**(SupplierID)

3. **SUPPLYINVENTORY** (SupplyInvID, InventoryName, UnitsInStock, Street, City, State, Country, PostalCode)

4. **SUPPLYORDERDETAILS** (SupplyOrderID, SupplyInvID, LineItemID, Quantity, TotalSupOrderValue, PricePerUnit)

FOREIGN KEY (SupplyInvID) references **SUPPLY\_INVENTORY**(SupplyInvID)

FOREIGN KEY (SupplyorderID) references **SUPPLYORDER**(SupplyorderID)

5. **USEDBY** (SuppInvID, UnitID)

FOREIGN KEY (SupplyInvID) references **SUPPLY\_INVENTORY**(SupplyInvID)

FOREIGN KEY(UnitID) references **MANUFACTURING UNIT**(UnitID)

6.**MANUFACTURINGUNIT** (UnitID, UnitName, Location, FacilitySize, ProductionCapacity, Status)

7.**MANUFACTURES** (UnitID, ProductID)

FOREIGN KEY(UnitID) references **MANUFACTURING UNIT**(UnitID)

FOREIGN KEY(ProductID) references **PRODUCTS**(ProductID)

8.**PRODUCT** (ProductID, ProductName, UnitPrice, Description, Weight, subCategoryId)

FOREIGN KEY (subCategoryId) references **SUBCATEGORY (**subCategoryID**)**

9.**SUBCATEGORY** (subCategoryID,CategoryID,subCategoryName)

FOREIGN KEY (CategoryID) references **CATEGORY** (CategoryID)

11. **CATEGORY** (CategoryID, CategoryName. Description)

12.**WAREHOUSE** (WarehouseId, UnitsInOrder, WarehouseName, Capacity, Discontinued, Location, WarehouseWorkerTeamID)

Foreign key (WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID)

13.**WAREHOUSINGDETAILS** (ProductId, WarehouseId,ProductWarehouseId, WhReorderLevel)

Foreign key PRODUCT (ProductId) references **PRODUCT** (ProductId )

Foreign key WAREHOUSE (WarehouseId) references **WAREHOUSE**(WarehouseId)

14. **SHIPMENT** (ShipmentID, WarehouseID, CarrierID, TrackingNumber, ShipmentDate, ArrivalDate, ShipmentStatus, ShipmentType, ShipmentWeight)

Foreign key (CarrierID) references **CARRIER** (CarrierID)

Foreign key (WarehouseId) references **WAREHOUSE**(WarehouseId)

15.**CARRIER** (CarrierID, CarrierName, Region)

16.**EQUIPMENT** (EquipmentID, ModelNumber, EquipmentName, Manufacturer, Type, MaintenanceFreq, UnitId)

FOREIGN KEY (UnitId) references **MANUFACTURINGUNIT** (UnitID)

17.**MAINTENANCESCHEDULE** (ScheduleID, EquipmentID, MaintenanceType, StartDate, EndDate)

Foreign key (EquipmentID) references **EQUIPMENT** (EquipmentID)

18.**CUSTOMER** (CustomerID, CustomerName, ContactNumber, EmailID, CreditLimit, ShippingAddress, PaymentID)

Foreign key (PaymentID) references **PAYMENTMENTHOD** (PaymentID)

19.**REVIEWDETAILS** (ProductID, CustomerID, ReviewNo, ReviewText, ReviewTitle, Rating)

Foreign key (ProductID) references **PRODUCT** (ProductID)

Foreign key (CustomerID) references **CUSTOMER** (CustomerID)

20. **PAYMENTMETHOD** (PaymentID, Description, PaymentGateway, PaymentType)

**a.** **PAYMENTCURRENCY** (PaymentID, Currency)

Foreign key (PaymnetID) references **PAYMENTMETHOD**(PaymentID)

21. **ORDERS** (OrderID, OrderTotalValue, ExpectedDeliveryDate, OrderDate, OrderType, CustomerID)

Foreign key CustomerID references **CUSTOMER** (CustomerID)

* 1. **OFFLINEORDER** (OrderID, EmployeeID)

Foreign key (OrderID) references **ORDERS**(OrderID)

Foreign key (EmployeeID) references **SALESREPRESENTATIVE**(EmployeeID)

* 1. **ONLINEORDER** (OrderID)

Foreign key (OrderID) references **ORDERS**(OrderID)

22. **ORDERDETAILS** (ProductID, OrderID, OrderItemID, OrderItems, OrderDate, ODTotalValue)

Foreign key (OrderID) references **ORDERS**(OrderID)

Foreign key (ProductID) references **PRODUCT** (ProductID)

23.**TERRITORY** (TerritoryID, TerritoryName, ExpectedRevenue, Region)

24.**EMPLOYEE** (EmployeeID, Email, FirstName,LastName,DepartmentID, EmployeeType)

Foreign key (DepartmentID) references **DEPARTMENT**(DepartmentID)

1. **SALESREPRESENTATIVE** (EmployeeID)

Foreign key EmployeeID) references **EMPLOYEES**(EmployeeID)

1. **WAREHOUSEWORKERS** (EmployeeID)

Foreign key EmployeeID) references **EMPLOYEES**(EmployeeID)

c. **EMPLOYEEPHONE** (EmployeeID,EmpPhone)

Foreign key (EmployeeID) references **EMPLOYEE** (EmployeeID)

25.**DEPARTMENT**(DepartmentID,DepartmentName,DepartmentHead)

26. **CONSISTS\_PACKER** (EmployeeID, WarehouseWorkerTeamID)

FOREIGN KEY(WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID)

27.**CONSISTS\_FORKLIFTOPERATOR** ( EmployeeID, WarehouseWorkerTeamID)

FOREIGN KEY(WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID)

28.**CONSISTS\_SHIPPINGCLERK** ( EmployeeID, WarehouseWorkerTeamID)

FOREIGN KEY(WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID )

29.**CONSISTS\_SUPERVISOR**( EmployeeID, WarehouseWorkerTeamID)

FOREIGN KEY(WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID )

30.**CONSISTS\_QUALITYCONTROLINSPECTOR** (EmployeeID, WarehouseWorkerTeamID)

FOREIGN KEY(WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID )

31.**CONSISTS\_WAREHOUSEMANAGER(** EmployeeID, WarehouseWorkerTeamID**)**

FOREIGN KEY(WarehouseWorkerTeamID) references **WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID )

32.**WAREHOUSEWORKERTEAM** (WarehouseWorkerTeamID, WarehouseWorkerTeamName)

33.**HANDLES** (EmployeeID, TerritoryID)

FOREIGN KEY(EmployeeID) references **SALESREPRESENTATIVE** (EmployeeID)

FOREIGN KEY(TerritoryID) references **TERRITORY**(TerritoryID)

## Data Dictionary (Relational)

### CARRIER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| CarrierID | VARCHAR2 | Primary Key  Format begins with CR followed by 3 digits |
| CarrierName | VARCHAR2 | Not Null |
| Region | VARCHAR2 |  |
| FD:CarrierID->CarrierName, Region | | |

### CATEGORY

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| CategoryID | VARCHAR2 | Primary Key  Format begins with C followed by 3 digits |
| CategoryName | VARCHAR2 | Not Null |
| Description | VARCHAR2 |  |
| FD:CategoryID->CategoryName, Description | | |

### CONSISTS\_PACKER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |

### CONSISTS\_FORKLIFTOPERATOR

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |

### CONSISTS\_SHIPPINGCLERK

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |

### CONSISTS\_SUPERVISOR

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |

### CONSISTS\_QUALITYCONTROLINSPECTOR

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |

### CONSISTS\_WAREHOUSEMANAGER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |

### CUSTOMER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| CustomerID | VARCHAR2 | Primary Key  Format begins with CUST followed by 6 digits |
| CustomerName | VARCHAR2 | Not Null |
| ContactNumber | VARCHAR2 |  |
| EmailID | VARCHAR2 |  |
| CreditLimit | NUMBER |  |
| ShippingAddress | VARCHAR2 |  |
| PaymentID | VARCHAR2 | Foreign Key  References PAYMENTMENTHOD(PaymentID) |
| FD:CustomerID->CustomerName, ContactNumber, EmailID, CreditLimit, ShippingAddress, PaymentID | | |

### DEPARTMENT

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| DepartmentID | VARCHAR2 | Primary Key  Format begins with DEPT followed by 3 digits |
| DepartmentName | VARCHAR2 | Not Null |
| DepartmentHead | VARCHAR2 |  |
| FD:DepartmentID->DepartmentName, DepartmentHead | | |

### EMPLOYEE

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Format begins with EMP followed by 8 digits |
| Email | VARCHAR2 | Not Null |
| FirstName | VARCHAR2 | Not Null |
| LastName | VARCHAR2 | Not Null |
| DepartmentID | VARCHAR2 | Foreign Key  References DEPARTMENT(DepartmentID) |
| EmployeeType | VARCHAR2 |  |
| FD:EmployeeID->Email, FirstName, LastName, DepartmentID, EmployeeType | | |

### EMPLOYEEPHONE

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Foreign Key  References EMPLOYEE(EmployeeID) |
| EmpPhone | VARCHAR2 |  |
| FD:EmployeeID,EmpPhone->EmployeeID, EmpPhone | | |

### SALESREPRESENTATIVE

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Foreign Key  References EMPLOYEE (EmployeeID) |

### WAREHOUSEWORKERS

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References EMPLOYEE(EmployeeID) |

### EQUIPMENT

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EquipmentID | VARCHAR2 | Primary Key  Format begins with EQ followed by 3 digits |
| ModelNumber | VARCHAR2 |  |
| EquipmentName | VARCHAR2 | Not Null |
| Manufacturer | VARCHAR2 |  |
| Type | VARCHAR2 |  |
| MaintenanceFreq | VARCHAR2 |  |
| UnitId | VARCHAR2 | Foreign Key  References MANUFACTURINGUNIT(UnitID) |
| FD:EquipmentID->ModelNumber, EquipmentName, Manufacturer, Type, MaintenanceFreq, UnitId | | |

1. HANDLES

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| EmployeeID | VARCHAR2 | Primary Key  Foreign Key  References SALESREPRESENTATIVE (EmployeeID) |
| TerritoryID | VARCHAR2 | Primary Key  Foreign Key  References TERRITORY(TerritoryID) |

### MAINTENANCESCHEDULE

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| ScheduleID | VARCHAR2 | Primary Key  Format begins with SCD followed by 4 digits |
| EquipmentID | VARCHAR2 | Primary Key  Foreign Key  References EQUIPMENT(EquipmentID) |
| MaintenanceType | VARCHAR2 | Check (Type IN (“Monthly”, “Weekly”, “Yearly”, “Emergency”)) |
| StartDate | DATE |  |
| EndDate | DATE |  |
| FD:ScheduleID,EquipmentID->MaintenanceType, StartDate, EndDate | | |

### MANUFACTURINGUNIT

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| UnitID | VARCHAR2 | Primary Key  Format begins with MF followed by 3 digits |
| UnitName | VARCHAR2 | NOT NULL |
| Location | VARCHAR2 |  |
| FacilitySize | VARCHAR2 |  |
| ProductionCapacity | NUMBER |  |
| Status | VARCHAR2 |  |
| FD:UnitID->UnitName, Location, FacilitySize, ProductionCapacity, Status | | |

### MANUFACTURES

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| UnitID | VARCHAR2 | Primary Key  Foreign Key  References MANUFACTURINGUNIT(UnitID) |
| ProductID | VARCHAR2 | Primary Key  Foreign Key  References PRODUCT(ProductID) |
| FD:UnitID,ProductID->UnitID, ProductID | | |

### ORDERS

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| OrderID | VARCHAR2 | Primary Key  Format begins with OR followed by 8 digits |
| OrderTotalValue | NUMBER |  |
| ExpectedDeliveryDate | DATE |  |
| OrderDate | DATE |  |
| OrderType | VARCHAR2 |  |
| CustomerID | VARCHAR2 | Foreign Key  References CUSTOMER(CustomerID) |
| FD: OrderID->OrderTotalValue, ExpectedDeliveryDate, OrderDate, OrderType, CustomerID | | |

### OFFLINEORDER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| OrderID | VARCHAR2 | Primary Key  Foreign Key  References ORDER(OrderID) |
| EmployeeID | VARCHAR2 | Foreign Key  References SALESREPRESENTATIVE(EmployeeID) |
| FD:OrderID,EmployeeID->OrderID, EmployeeID | | |

### ONLINEORDER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| OrderID | VARCHAR2 | Primary Key  Foreign Key  References ORDER(OrderID) |

### ORDERDETAILS

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| ProductID | VARCHAR2 | Foreign Key  References PRODUCT(ProductID) |
| OrderID | VARCHAR2 | Foreign Key  References ORDER(OrderID) |
| OrderItemID | VARCHAR2 | Primary Key |
| OrderItems | NUMBER | Not Null |
| OrderDate | DATE |  |
| ODTotalValue | NUMBER |  |
| FD:ProductID,OrderID,OrderItemID->OrderItems, OrderDate, ODTotalValue | | |

### PAYMENTMETHOD

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| PaymentID | VARCHAR2 | Primary Key  Format begins with PY followed by 8 digits |
| Description | VARCHAR2 |  |
| PaymentGateway | VARCHAR2 |  |
| PaymentType | VARCHAR2 | Check (Type IN (“Credit”, “Debit”, “Wallet”, “Cash”)) |
| FD:PaymentID→Description, PaymentGateway, PaymentType | | |

### PAYMENTCURRENCY

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| PaymentID | VARCHAR2 | Foreign Key  References PAYMENTMETHOD(PaymentID) |
| Currency | VARCHAR2 | Primary Key |
| FD: PaymentID -> Currency | | |

### PRODUCT

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| ProductID | VARCHAR2 | Primary Key  Format begins with P followed by 4 digits |
| ProductName | VARCHAR2 | NOT NULL |
| UnitPrice | NUMBER |  |
| Description | VARCHAR2 |  |
| Weight | NUMBER |  |
| subCategoryID | VARCHAR2 | Foreign Key  References SUBCATEGORY(subCategoryID) |
| FD:ProductID->ProductName, UnitPrice, Description,Weight, subCategoryID | | |

### REVIEWDETAILS

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| ProductID | VARCHAR2 | Primary Key  Foreign Key  References PRODUCT(ProductID) |
| CustomerID | VARCHAR2 | Primary Key  Foreign Key  References CUSTOMER(CustomerID) |
| ReviewNo | VARCHAR2 | Primary Key |
| ReviewText | VARCHAR2 |  |
| ReviewTitle | VARCHAR2 |  |
| Rating | NUMBER |  |
| FD:ProductID,CustomerID,ReviewNo->ReviewText, ReviewTitle, Rating | | |

### SHIPMENT

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| ShipmentID | VARCHAR2 | Primary Key  Format begins with SH followed by 8 digits |
| WarehouseID | VARCHAR2 | Foreign Key  References WAREHOUSE(WarehouseId) |
| CarrierID | VARCHAR2 | Foreign Key  References CARRIER(CarrierID) |
| TrackingNumber | VARCHAR2 |  |
| ShipmentDate | DATE |  |
| ArrivalDate | DATE |  |
| ShipmentStatus | VARCHAR2 |  |
| ShipmentType | VARCHAR2 |  |
| ShipmentWeight | NUMBER |  |
| FD: ShipmentID->TrackingNumber, ShipmentDate, ArrivalDate, ShipmentStatus, ShipmentType, ShipmentWeight,WarehouseID,CarrierID | | |

### SUBCATEGORY

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| subCategoryID | VARCHAR2 | Primary Key  Format begins with SC followed by 3 digits |
| CategoryID | VARCHAR2 | Foreign Key  References Category(CategoryID) |
| subCategoryName | VARCHAR2 | NOT NULL |
| FD:subCategoryID->CategoryID, subCategoryName | | |

### SUPPLIER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SupplierID | VARCHAR2 | Primary Key Format begins with S followed by 3 digits |
| SupplierName | VARCHAR2 | NOT NULL |
| Street | VARCHAR2 |  |
| City | VARCHAR2 |  |
| State | VARCHAR2 |  |
| Country | VARCHAR2 |  |
| PostalCode | VARCHAR2 |  |
| ContactName | VARCHAR2 | NOT NULL |
| ContactDesignation | VARCHAR2 |  |
| FD:SupplierID->SupplierName, Street, City, Country, PostalCode, ContactName, ContactDesignation | | |

### SUPPLIER\_PHONE

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SupplierID | VARCHAR2 | Primary Key  Foreign Key  References SUPPLIER(SupplierID) |
| PhoneNo | VARCHAR2 | Primary Key |
| FD:SupplierID,PhoneNo->SupplierID, PhoneNo | | |

### SUPPLIER\_EMAIL

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SupplierID | VARCHAR2 | Primary Key  Foreign Key  References SUPPLIER(SupplierID) |
| Email | VARCHAR2 | Primary Key |
| FD:SupplierID,Email->SupplierID, Email | | |

### SUPPLYINVENTORY

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SupplyInvID | VARCHAR2 | Primary Key  Format begins with INV followed by 4 digits |
| InventoryName | VARCHAR2 | Not Null |
| UnitsInStock | NUMBER |  |
| Street | VARCHAR2 |  |
| City | VARCHAR2 |  |
| State | VARCHAR2 |  |
| Country | VARCHAR2 |  |
| PostalCode | VARCHAR2 |  |
| FD:SupplyInvID->InventoryName, UnitsInStock, Street, City, State, Country, PostalCode | | |

### SUPPLYORDER

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SupplyorderID | NUMBER | Primary Key  7 digit number |
| SupplyPrice | NUMBER |  |
| TotalValue | NUMBER |  |
| SupOrderDate | DATE |  |
| ItemName | VARCHAR2 |  |
| PaymentInfo | VARCHAR2 |  |
| NoofLineitems | NUMBER |  |
| ExpectedSupDeliveryDate | DATE |  |
| SupplierID | VARCHAR2 | Foreign Key  References SUPPLIER(SupplierID) |
| FD:SupplyorderID->SupplyPrice,TotalValue, SupOrderDate, ItemName,  PaymentInfo, NoofLineitems, ExpectedSupDeliveryDate, SupplierID | | |

### SUPPLYORDERDETAILS

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SupplyOrderID | NUMBER | Primary Key  Foreign Key  References SUPPLYORDER(SupplyOrderID) |
| SupplyInvID | VARCHAR2 | Primary Key  Foreign Key  References SUPPLYINVENTORY(SupplyInvID) |
| LineItemID | NUMBER | Not Null |
| Quantity | NUMBER | Not Null |
| TotalSupOrderValue | NUMBER |  |
| PricePerUnit | NUMBER |  |
| FD:SupplyOrderID,SupplyInvID->LineItemID,TotalSupOrderValue,Quantity,PricePerUnit | | |

### TERRITORY

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| TerritoryID | VARCHAR2 | Primary Key  Format begins with T followed by 4 digits |
| TerritoryName | VARCHAR2 |  |
| ExpectedRevenue | NUMBER |  |
| Region | VARCHAR2 |  |
| FD:TerritoryID-> TerritoryName, ExpectedRevenue | | |

### USEDBY

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| SuppInvID | VARCHAR2 | Primary Key  Foreign Key  References SUPPLYINVENTORY(SupplyInvID) |
| UnitID | VARCHAR2 | Primary Key  Foreign Key  References MANUFACTURINGUNIT(UnitID) |
| FD:SuppInvID,UnitID->SuppInvID,UnitID | | |

### WAREHOUSE

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| WarehouseId | VARCHAR2 | Primary Key  Format begins with WH followed by 4 digits |
| UnitsInOrder | NUMBER | Not Null |
| WarehouseName | VARCHAR2 |  |
| Capacity | NUMBER |  |
| Discontinued | CHAR |  |
| Location | VARCHAR2 |  |
| WarehouseWorkerTeamID | VARCHAR2 | Foreign Key  References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID) |
| FD:WarehouseId->UnitsInOrder, WarehouseName, Capacity, Discontinued, Location, WarehouseWorkerTeamID | | |

### WAREHOUSINGDETAILS

|  |  |  |
| --- | --- | --- |
| **Schema Construct** | **Data Type** | **Constraint** |
| ProductId | VARCHAR2 | Primary Key  Foreign Key  References PRODUCT(ProductId) |
| WarehouseId | VARCHAR2 | Primary Key  Foreign Key  References WAREHOUSE(WarehouseId) |
| ProductWarehouseId | VARCHAR2 | Primary Key  Format begins with PWID followed by 4 digits |
| WhReorderLevel | NUMBER | Not Null |
| FD:ProductId,WarehouseId,ProductWarehouseId->WhReorderLevel | | |

### WAREHOUSEWORKERTEAM

|  |  |  |
| --- | --- | --- |
| Schema Construct | Data Type | Constraint |
| WarehouseWorkerTeamID | VARCHAR2 | Primary Key  Format begins with WT followed by 3 digits |
| WarehouseWorkerTeamName | VARCHAR2 | Not Null |

## DDL Appendix

-- 1. SUPPLIER

CREATE TABLE SUPPLIER (

SupplierID VARCHAR2(4) CONSTRAINT pk\_supplier PRIMARY KEY,

SupplierName VARCHAR2(255) NOT NULL,

Street VARCHAR2(255),

City VARCHAR2(255),

State VARCHAR2(255),

Country VARCHAR2(255),

PostalCode VARCHAR2(255),

ContactName VARCHAR2(255) NOT NULL,

ContactDesignation VARCHAR2(255)

);

-- 2. SUPPLIER\_PHONE

CREATE TABLE SUPPLIER\_PHONE (

SupplierID VARCHAR2(4),

PhoneNo VARCHAR2(255) UNIQUE NOT NULL,

CONSTRAINT pk\_supplier\_phone PRIMARY KEY (SupplierID, PhoneNo),

CONSTRAINT fk\_supplier\_phone\_supplier FOREIGN KEY (SupplierID) REFERENCES SUPPLIER(SupplierID)

);

-- 3. SUPPLIER\_EMAIL

CREATE TABLE SUPPLIER\_EMAIL (

SupplierID VARCHAR2(4),

Email VARCHAR2(255) UNIQUE NOT NULL,

CONSTRAINT pk\_supplier\_email PRIMARY KEY (SupplierID, Email),

CONSTRAINT fk\_supplier\_email\_supplier FOREIGN KEY (SupplierID) REFERENCES SUPPLIER(SupplierID)

);

-- 4. SUPPLYORDER

CREATE TABLE SUPPLYORDER (

SupplyorderID NUMBER(7) CONSTRAINT pk\_supplyorder PRIMARY KEY,

SupplyPrice NUMBER,

TotalValue NUMBER,

SupOrderDate DATE,

ItemName VARCHAR2(255),

PaymentInfo VARCHAR2(255),

NoofLineitems INT,

ExpectedSupDeliveryDate DATE,

SupplierID VARCHAR2(4),

CONSTRAINT fk\_supplyorder\_supplier FOREIGN KEY (SupplierID) REFERENCES SUPPLIER(SupplierID)

);

-- 5. SUPPLYINVENTORY

CREATE TABLE SUPPLYINVENTORY (

SupplyInvID VARCHAR2(8) CONSTRAINT pk\_supplyinventory PRIMARY KEY,

InventoryName VARCHAR2(255) NOT NULL,

UnitsInStock INT NOT NULL,

Street VARCHAR2(255),

City VARCHAR2(255),

State VARCHAR2(255),

Country VARCHAR2(255),

PostalCode VARCHAR2(255)

);

-- 6. SUPPLYORDERDETAILS

CREATE TABLE SUPPLYORDERDETAILS (

SupplyOrderID NUMBER,

SupplyInvID VARCHAR2(8),

LineItemID INT NOT NULL,

Quantity INT NOT NULL,

TotalSupOrderValue NUMBER,

PricePerUnit NUMBER,

CONSTRAINT pk\_supplyorderdetails PRIMARY KEY (SupplyOrderID, SupplyInvID, LineItemID),

CONSTRAINT fk\_supplyorderdetails\_supplyorder FOREIGN KEY (SupplyOrderID) REFERENCES SUPPLYORDER(SupplyOrderID),

CONSTRAINT fk\_supplyorderdetails\_supplyinventory FOREIGN KEY (SupplyInvID) REFERENCES SUPPLYINVENTORY(SupplyInvID)

);

-- 7. MANUFACTURINGUNIT

CREATE TABLE MANUFACTURINGUNIT (

UnitID VARCHAR2(6) CONSTRAINT pk\_manufacturingunit PRIMARY KEY ,

UnitName VARCHAR2(255) NOT NULL,

Location VARCHAR2(255),

FacilitySize VARCHAR2(255),

ProductionCapacity INT,

Status VARCHAR2(255)

);

-- 8. USEDBY

CREATE TABLE USEDBY (

SuppInvID VARCHAR2(8),

UnitID VARCHAR2(6),

CONSTRAINT pk\_usedby PRIMARY KEY (SuppInvID, UnitID),

CONSTRAINT fk\_usedby\_supplyinventory FOREIGN KEY (SuppInvID) REFERENCES SUPPLYINVENTORY(SupplyInvID),

CONSTRAINT fk\_usedby\_manufacturingunit FOREIGN KEY (UnitID) REFERENCES MANUFACTURINGUNIT(UnitID)

);

-- 9. CATEGORY

CREATE TABLE CATEGORY (

CategoryID VARCHAR2(6) CONSTRAINT pk\_category PRIMARY KEY ,

CategoryName VARCHAR2(255) NOT NULL,

Description VARCHAR2(255)

);

-- 10. SUBCATEGORY

CREATE TABLE SUBCATEGORY (

subCategoryID VARCHAR2(6) CONSTRAINT pk\_subcategory PRIMARY KEY ,

CategoryID VARCHAR2(6),

subCategoryName VARCHAR2(255) NOT NULL,

CONSTRAINT fk\_subcategory\_category FOREIGN KEY (CategoryID) REFERENCES CATEGORY(CategoryID)

);

-- 11. PRODUCT

CREATE TABLE PRODUCT (

ProductID VARCHAR2(6) CONSTRAINT pk\_product PRIMARY KEY ,

ProductName VARCHAR2(255) NOT NULL,

UnitPrice NUMBER,

Description VARCHAR2(255),

Weight NUMBER,

subCategoryID VARCHAR2(6),

CONSTRAINT fk\_product\_subcategory FOREIGN KEY (subCategoryID) REFERENCES SUBCATEGORY(subCategoryID)

);

-- 12. MANUFACTURES

CREATE TABLE MANUFACTURES (

UnitID VARCHAR2(6),

ProductID VARCHAR2(6),

CONSTRAINT pk\_manufactures PRIMARY KEY (UnitID, ProductID),

CONSTRAINT fk\_manufactures\_manufacturingunit FOREIGN KEY (UnitID) REFERENCES MANUFACTURINGUNIT(UnitID),

CONSTRAINT fk\_manufactures\_product FOREIGN KEY (ProductID) REFERENCES PRODUCT(ProductID)

);

-- 13. WAREHOUSEWORKERTEAM

CREATE TABLE WAREHOUSEWORKERTEAM (

WarehouseWorkerTeamID VARCHAR2(6) CONSTRAINT pk\_warehouseworkerteam PRIMARY KEY ,

WarehouseWorkerTeamName VARCHAR2(255) NOT NULL

);

-- 14. WAREHOUSE

CREATE TABLE WAREHOUSE (

WarehouseId VARCHAR2(8) CONSTRAINT pk\_warehouse PRIMARY KEY ,

UnitsInOrder INT NOT NULL,

WarehouseName VARCHAR2(255),

Capacity INT,

UnitsInStock INT NOT NULL,

Discontinued CHAR(1),

Location VARCHAR2(255),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT fk\_warehouse\_warehouseworkerteam FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 15. WAREHOUSINGDETAILS

CREATE TABLE WAREHOUSINGDETAILS (

ProductId VARCHAR2(6),

WarehouseId VARCHAR2(8),

ProductWarehouseId VARCHAR2(8) CONSTRAINT pk\_warehousingdetails PRIMARY KEY ,

WhReorderLevel INT NOT NULL,

CONSTRAINT fk\_warehousingdetails\_product FOREIGN KEY (ProductId) REFERENCES PRODUCT(ProductId),

CONSTRAINT fk\_warehousingdetails\_warehouse FOREIGN KEY (WarehouseId) REFERENCES WAREHOUSE(WarehouseId)

);

-- 16. CARRIER

CREATE TABLE CARRIER (

CarrierID VARCHAR2(6) CONSTRAINT pk\_carrier PRIMARY KEY ,

CarrierName VARCHAR2(255) NOT NULL,

Region VARCHAR2(255)

);

-- 17. SHIPMENT

CREATE TABLE SHIPMENT (

ShipmentID VARCHAR2(10) CONSTRAINT pk\_shipment PRIMARY KEY ,

WarehouseID VARCHAR2(8),

CarrierID VARCHAR2(6),

TrackingNumber VARCHAR2(255),

ShipmentDate DATE,

ArrivalDate DATE,

ShipmentStatus VARCHAR2(255),

ShipmentType VARCHAR2(255),

ShipmentWeight NUMBER,

CONSTRAINT fk\_shipment\_warehouse FOREIGN KEY (WarehouseID) REFERENCES WAREHOUSE(WarehouseID),

CONSTRAINT fk\_shipment\_carrier FOREIGN KEY (CarrierID) REFERENCES CARRIER (CarrierID)

);

-- 18. EQUIPMENT

CREATE TABLE EQUIPMENT (

EquipmentID VARCHAR2(6) CONSTRAINT pk\_equipment PRIMARY KEY ,

ModelNumber VARCHAR2(255),

EquipmentName VARCHAR2(255) NOT NULL,

Manufacturer VARCHAR2(255),

Type VARCHAR2(255),

MaintenanceFreq VARCHAR2(255),

UnitId VARCHAR2(6),

CONSTRAINT fk\_equipment\_manufacturingunit FOREIGN KEY (UnitId) REFERENCES MANUFACTURINGUNIT(UnitID)

);

-- 19. MAINTENANCESCHEDULE

CREATE TABLE MAINTENANCESCHEDULE (

ScheduleID VARCHAR2(10) CONSTRAINT pk\_maintenanceschedule PRIMARY KEY ,

EquipmentID VARCHAR2(6),

MaintenanceType VARCHAR2(255) CHECK (MaintenanceType IN ('Monthly', 'Weekly', 'Yearly', 'Emergency')),

StartDate DATE,

EndDate DATE,

CONSTRAINT fk\_maintenanceschedule\_equipment FOREIGN KEY (EquipmentID) REFERENCES EQUIPMENT(EquipmentID)

);

-- 20. PAYMENTMETHOD

CREATE TABLE PAYMENTMETHOD (

PaymentID VARCHAR2(10) CONSTRAINT pk\_paymentmethod PRIMARY KEY,

Description VARCHAR2(255),

PaymentGateway VARCHAR2(255),

PaymentType VARCHAR2(255) CHECK (PaymentType IN ('Credit', 'Debit', 'Wallet', 'Cash'))

);

-- 21. CUSTOMER

CREATE TABLE CUSTOMER (

CustomerID VARCHAR2(12) CONSTRAINT pk\_customer PRIMARY KEY,

CustomerName VARCHAR2(255) NOT NULL,

ContactNumber VARCHAR2(255),

EmailID VARCHAR2(255),

CreditLimit NUMBER,

ShippingAddress VARCHAR2(255),

PaymentID VARCHAR2(10),

CONSTRAINT fk\_customer\_payment FOREIGN KEY (PaymentID) REFERENCES PAYMENTMETHOD(PaymentID)

);

-- 22. REVIEWDETAILS

CREATE TABLE REVIEWDETAILS (

ProductID VARCHAR2(6),

CustomerID VARCHAR2(12),

ReviewNo VARCHAR2(255) PRIMARY KEY,

ReviewText VARCHAR2(255),

ReviewTitle VARCHAR2(255),

Rating INT,

CONSTRAINT fk\_reviewdetails\_product FOREIGN KEY (ProductID) REFERENCES PRODUCT(ProductID),

CONSTRAINT fk\_reviewdetails\_customer FOREIGN KEY (CustomerID) REFERENCES CUSTOMER(CustomerID)

);

-- 23. PAYMENTCURRENCY

CREATE TABLE PAYMENTCURRENCY (

PaymentID VARCHAR2(10),

Currency VARCHAR2(255),

CONSTRAINT pk\_paymentcurrency PRIMARY KEY (PaymentID, Currency),

CONSTRAINT fk\_paymentcurrency\_paymentmethod FOREIGN KEY (PaymentID) REFERENCES PAYMENTMETHOD(PaymentID)

);

-- 24. ORDERS

CREATE TABLE ORDERS (

OrderID VARCHAR2(10) CONSTRAINT pk\_order PRIMARY KEY ,

OrderTotalValue NUMBER,

ExpectedDeliveryDate DATE,

OrderDate DATE,

OrderType VARCHAR2(255),

CustomerID VARCHAR2(12),

CONSTRAINT fk\_order\_customer FOREIGN KEY (CustomerID) REFERENCES CUSTOMER(CustomerID)

);

-- 25. ORDERDETAILS

CREATE TABLE ORDERDETAILS (

ProductID VARCHAR2(6),

OrderID VARCHAR2(10),

OrderItemID INT NOT NULL,

OrderItems INT NOT NULL,

OrderDate DATE,

ODTotalValue NUMBER,

CONSTRAINT pk\_orderdetails PRIMARY KEY (ProductID, OrderID, OrderItemID),

CONSTRAINT fk\_orderdetails\_product FOREIGN KEY (ProductID) REFERENCES PRODUCT(ProductID),

CONSTRAINT fk\_orderdetails\_order FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID)

);

-- 26. DEPARTMENT

CREATE TABLE DEPARTMENT (

DepartmentID VARCHAR2(7) CONSTRAINT pk\_department PRIMARY KEY ,

DepartmentName VARCHAR2(255) NOT NULL,

DepartmentHead VARCHAR2(255)

);

--alter table DEPARTMENT MODIFY DepartmentID VARCHAR2(7);

-- 27. EMPLOYEE

CREATE TABLE EMPLOYEE (

EmployeeID VARCHAR2(12) CONSTRAINT pk\_employee PRIMARY KEY ,

Email VARCHAR2(255) NOT NULL,

FirstName VARCHAR2(255) NOT NULL,

LastName VARCHAR2(255) NOT NULL,

DepartmentID VARCHAR2(6),

EmployeeType VARCHAR2(255),

CONSTRAINT fk\_employee\_department FOREIGN KEY (DepartmentID) REFERENCES DEPARTMENT(DepartmentID)

);

--alter table EMPLOYEE MODIFY DepartmentID VARCHAR2(7);

-- 28. SALESREPRESENTATIVE

CREATE TABLE SALESREPRESENTATIVE (

EmployeeID VARCHAR2(12),

CONSTRAINT pk\_salesrepresentative PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_salesrepresentative\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID)

);

-- 29. OFFLINEORDER

CREATE TABLE OFFLINEORDER (

OrderID VARCHAR2(10),

EmployeeID VARCHAR2(12),

CONSTRAINT pk\_offlineorder PRIMARY KEY (OrderID),

CONSTRAINT fk\_offlineorder\_order FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID),

CONSTRAINT fk\_offlineorder\_employee FOREIGN KEY (EmployeeID) REFERENCES SALESREPRESENTATIVE(EmployeeID)

);

-- 30. ONLINEORDER

CREATE TABLE ONLINEORDER (

OrderID VARCHAR2(10),

CONSTRAINT pk\_onlineorder PRIMARY KEY (OrderID),

CONSTRAINT fk\_onlineorder\_order FOREIGN KEY (OrderID) REFERENCES ORDERS(OrderID)

);

-- 31. TERRITORY

CREATE TABLE TERRITORY (

TerritoryID VARCHAR2(8) CONSTRAINT pk\_territory PRIMARY KEY,

TerritoryName VARCHAR2(255),

ExpectedRevenue NUMBER,

Region VARCHAR2(255)

);

-- 32. EMPLOYEEPHONE

CREATE TABLE EMPLOYEEPHONE (

EmployeeID VARCHAR2(12),

EmpPhone VARCHAR2(255),

CONSTRAINT pk\_employeephone PRIMARY KEY (EmployeeID, EmpPhone),

CONSTRAINT fk\_employeephone\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID)

);

-- 33. WAREHOUSEWORKERS

CREATE TABLE WAREHOUSEWORKERS (

EmployeeID VARCHAR2(12),

CONSTRAINT pk\_warehouseworkers PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_warehouseworkers\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID)

);

-- 34. CONSISTS\_PACKER

CREATE TABLE CONSISTS\_PACKER (

EmployeeID VARCHAR2(12),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT pk\_consists\_packer PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_consists\_packer\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),

CONSTRAINT fk\_consists\_packer\_warehouse FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 35. CONSISTS\_FORKLIFTOPERATOR

CREATE TABLE CONSISTS\_FORKLIFTOPERATOR (

EmployeeID VARCHAR2(12),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT pk\_consists\_forkliftoperator PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_consists\_forkliftoperator\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),

CONSTRAINT fk\_consists\_forkliftoperator\_warehouse FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 36. CONSISTS\_SHIPPINGCLERK

CREATE TABLE CONSISTS\_SHIPPINGCLERK (

EmployeeID VARCHAR2(12),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT pk\_consists\_shippingclerk PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_consists\_shippingclerk\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),

CONSTRAINT fk\_consists\_shippingclerk\_warehouse FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 37. CONSISTS\_SUPERVISOR

CREATE TABLE CONSISTS\_SUPERVISOR (

EmployeeID VARCHAR2(12),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT pk\_consists\_supervisor PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_consists\_supervisor\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),

CONSTRAINT fk\_consists\_supervisor\_warehouse FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 38. CONSISTS\_QUALITYCONTROLINSPECTOR

CREATE TABLE CONSISTS\_QUALITYCONTROLINSPECTOR (

EmployeeID VARCHAR2(12),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT pk\_consists\_qc\_inspector PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_consists\_qc\_inspector\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),

CONSTRAINT fk\_consists\_qc\_inspector\_warehouse FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 39. CONSISTS\_WAREHOUSEMANAGER

CREATE TABLE CONSISTS\_WAREHOUSEMANAGER (

EmployeeID VARCHAR2(12),

WarehouseWorkerTeamID VARCHAR2(6),

CONSTRAINT pk\_consists\_warehousemanager PRIMARY KEY (EmployeeID),

CONSTRAINT fk\_consists\_warehousemanager\_employee FOREIGN KEY (EmployeeID) REFERENCES EMPLOYEE(EmployeeID),

CONSTRAINT fk\_consists\_warehousemanager\_warehouse FOREIGN KEY (WarehouseWorkerTeamID) REFERENCES WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

);

-- 40. HANDLES   
CREATE TABLE HANDLES (  
    EmployeeID VARCHAR2(6) ,  
    TerritoryID VARCHAR2(8) ,  
    CONSTRAINT pk\_handles PRIMARY KEY (EmployeeID, TerritoryID),  
    CONSTRAINT fk\_handles\_employee FOREIGN KEY (EmployeeID) REFERENCES SALESREPRESENTATIVE(EmployeeID),  
    CONSTRAINT fk\_handles\_territory FOREIGN KEY (TerritoryID) REFERENCES TERRITORY(TerritoryID)  
);

TRIGGER FOR SEQUENCE GENERATION

1. **Generate OrderID when inserting a record into the Order table using triggers**

CREATE OR REPLACE TRIGGER OrderIDTriggerGenerator

BEFORE INSERT ON ORDERS

FOR EACH ROW

DECLARE

  dw\_MaxOrderID NUMBER;

BEGIN

  -- Get the maximum existing OrderID

  SELECT MAX(TO\_NUMBER(SUBSTR(OrderID, 3)))

  INTO dw\_MaxOrderID

  FROM ORDERS;

  -- If there are no existing records, set the starting value to 0

  IF dw\_MaxOrderID IS NULL THEN

    dw\_MaxOrderID := 0;

  END IF;

  -- Generate the next OrderID by incrementing the maximum value

  :NEW.OrderID := 'OR' || LPAD(dw\_MaxOrderID + 1, 8, '0');

EXCEPTION

# Chapter 4: SQL Queries

1. **List all products with their total sales, average ratings, and a flag indicating if the warehouse they are stored in are discontinued:**

SELECT

   p.ProductName,

   SUM(ODTotalValue) AS TotalSales,

   AVG(Rating) AS AvgRating,

   CASE WHEN w1.Discontinued = 1 THEN 'Discontinued'

        ELSE 'Not discontinued' END AS DiscontinuedStatus

FROM PRODUCT p

JOIN ORDERDETAILS od ON p.ProductID = od.ProductID

JOIN ORDERS o ON od.OrderID = o.OrderID

JOIN WAREHOUSINGDETAILS w ON p.ProductID = w.ProductId

JOIN WAREHOUSE w1 ON w.warehouseid = w1.warehouseid

JOIN REVIEWDETAILS rd ON p.ProductID = rd.ProductID

GROUP BY p.ProductName, w1.discontinued, ODTotalValue, Rating;

1. **Total sales value generated by each sales representative and identifies the top-performing representatives based on their sales performance:**

WITH SalesData AS (

    SELECT

        s.EmployeeID,

        e.FirstName || ' ' || e.LastName AS SalesRepresentative,

        SUM(o.OrderTotalValue) AS TotalSalesValue

    FROM

        OFFLINEORDER s

    JOIN

        Orders o ON s.OrderID = o.OrderID

    JOIN

        SALESREPRESENTATIVE sr ON s.EmployeeID = sr.EmployeeID

    JOIN

        EMPLOYEE e ON sr.EmployeeID = e.EmployeeID

    GROUP BY

        s.EmployeeID, e.FirstName, e.LastName

)

SELECT

    EmployeeID,

    SalesRepresentative,

    TotalSalesValue

FROM

    SalesData

ORDER BY

    TotalSalesValue DESC;

1. **Find the details of the carrier, no. of shipments they are responsible for, date of next shipment, days between 2 shipments**:

WITH ShipmentCTE AS (

    SELECT

        carrierid,

        TO\_DATE(ShipmentDate, 'DD-MON-YYYY') AS ShipmentDate,

        LEAD(TO\_DATE(ShipmentDate, 'DD-MON-YYYY')) OVER (PARTITION BY carrierid ORDER BY TO\_DATE(ShipmentDate, 'DD-MON-YYYY')) AS NextShipmentDate

    FROM

        SHIPMENT

)

SELECT

    c.CarrierName,

    COUNT(s.ShipmentID) AS ShipmentCount,min(s.ShipmentDate) AS CURRENT\_SHIPMENT\_DATE,

    TO\_CHAR(MAX(sc.NextShipmentDate), 'DD-MON-YYYY') AS Next\_Shipment\_Date,

    COALESCE(TO\_CHAR((MAX(sc.NextShipmentDate) - MIN(TO\_DATE(s.ShipmentDate, 'DD-MON-YYYY'))), '99999'), '') AS Days\_Between\_Shipments

FROM

    ShipmentCTE sc

 JOIN

    carrier c ON sc.carrierid = c.carrierid

LEFT JOIN

    SHIPMENT s ON sc.carrierid = s.carrierid AND sc.NextShipmentDate = TO\_DATE(s.ShipmentDate, 'DD-MON-YYYY')

GROUP BY

    c.CarrierName

ORDER BY

    c.CarrierName;

1. **List the top 3 warehouses with the highest total shipment weights and their respective carriers:**

SELECT w.WarehouseId, w.WarehouseName, c.CarrierName, SUM(s.ShipmentWeight) AS TotalShipmentWeight

FROM WAREHOUSE w

JOIN SHIPMENT s ON w.WarehouseId = s.WarehouseID

JOIN CARRIER c ON s.CarrierID = c.CarrierID

GROUP BY w.WarehouseId, w.WarehouseName, c.CarrierName

ORDER BY TotalShipmentWeight DESC

FETCH FIRST 3 ROWS ONLY;

1. **Identify Customers with Unusual Order Frequency:**

SELECT

c.CustomerID,

c.CustomerName,

COUNT(o.OrderID) AS TotalOrders,

CASE

     WHEN COUNT(o.OrderID) < AVG(COUNT(o.OrderID)) OVER () - STDDEV(COUNT(o.OrderID)) OVER () THEN 'Low Order Frequency'

     WHEN COUNT(o.OrderID) > AVG(COUNT(o.OrderID)) OVER () + STDDEV(COUNT(o.OrderID)) OVER () THEN 'High Order Frequency'

     ELSE 'Normal Order Frequency'

END AS OrderFrequencyStatus

FROM

CUSTOMER c

JOIN

ORDERS o ON c.CustomerID = o.CustomerID

GROUP BY

c.CustomerID, c.CustomerName

ORDER BY

TotalOrders DESC;

1. **Retrieve a list of products, their categories, and the number of orders placed for each product:**

SELECT

    P.ProductID,

    P.ProductName,

    C.CategoryName,

    S.SubCategoryName,

    COUNT(OD.OrderID) AS NumberOfOrders

FROM

    PRODUCT P

JOIN

    SUBCATEGORY S ON P.SubCategoryID = S.SubCategoryID

JOIN

    CATEGORY C ON S.CategoryID = C.CategoryID

LEFT JOIN

    ORDERDETAILS OD ON P.ProductID = OD.ProductID

GROUP BY

    P.ProductID, P.ProductName, C.CategoryName, S.SubCategoryName

ORDER BY

    NumberOfOrders DESC;

1. **Calculate the percentage of units in stock compared to the capacity for each warehouse**:

SELECT w.WarehouseId, w.WarehouseName, round(((SUM(w.UnitsInStock) / w.Capacity) \* 100),2) AS StockPercentage

FROM WAREHOUSE w

GROUP BY w.WarehouseId, w.WarehouseName, w.Capacity;

1. **Find the customers who have exceeded their credit limit:**

SELECT c.CustomerID, c.CustomerName, CreditLimit, SUM(OrderTotalValue) AS TotalOrders

FROM CUSTOMER c

JOIN ORDERS o ON c.CustomerID = o.CustomerID

GROUP BY c.CustomerID, CustomerName, CreditLimit

HAVING SUM(OrderTotalValue) > CreditLimit;

1. **List the top 3 products with the highest average customer ratings:**

SELECT p.ProductID, p.ProductName, AVG(r.Rating) AS AvgRating

FROM PRODUCT p

JOIN REVIEWDETAILS r ON p.ProductID = r.ProductID

GROUP BY p.ProductID, p.ProductName

ORDER BY AvgRating DESC

FETCH FIRST 3 ROWS ONLY;

1. **Find the top 5 suppliers with the highest total supply order values:**

SELECT s.SupplierID, s.SupplierName, SUM(o.TotalValue) AS TotalOrderValue

FROM SUPPLIER s

JOIN SUPPLYORDER o ON s.SupplierID = o.SupplierID

GROUP BY s.SupplierID, s.SupplierName

ORDER BY TotalOrderValue DESC

FETCH FIRST 5 ROWS ONLY;

1. **Find Products with Fluctuating Demand**

WITH MonthlyProductSales AS (

    SELECT

        p.ProductID,

        p.ProductName,

        TO\_CHAR(od.OrderDate, 'YYYY-MM') AS OrderMonth,

        SUM(od.OrderItems) AS MonthlySales

    FROM

        PRODUCT p

    LEFT JOIN

        ORDERDETAILS od ON p.ProductID = od.ProductID

    GROUP BY

        p.ProductID, p.ProductName, TO\_CHAR(od.OrderDate, 'YYYY-MM')

),

SalesFluctuationCTE AS (

    SELECT

        ProductID,

        ProductName,

        OrderMonth,

        MonthlySales,

        LAG(MonthlySales) OVER (PARTITION BY ProductID ORDER BY OrderMonth) AS PreviousMonthSales,

        CASE

            WHEN LAG(MonthlySales) OVER (PARTITION BY ProductID ORDER BY OrderMonth) IS NOT NULL

            THEN MonthlySales - LAG(MonthlySales) OVER (PARTITION BY ProductID ORDER BY OrderMonth)

            ELSE NULL

        END AS SalesFluctuation

    FROM

        MonthlyProductSales

)

SELECT

    ProductID,

    ProductName,

    OrderMonth,

    MonthlySales,

    PreviousMonthSales,

    SalesFluctuation

FROM

    SalesFluctuationCTE

WHERE

    SalesFluctuation IS NOT NULL

ORDER BY

    ProductID, OrderMonth;

1. **Retrieve a list of products and their total sales quantity, grouped by subcategory:**

WITH RankedProducts AS (  
 SELECT  
     p.ProductID,  
     p.ProductName,  
     c.CategoryName,  
     od.OrderItems,  
     RANK() OVER (PARTITION BY c.CategoryID ORDER BY od.OrderItems DESC) AS SalesRank  
 FROM  
     PRODUCT p  
 JOIN  
     SUBCATEGORY sc ON p.SubCategoryID = sc.SubCategoryID  
 JOIN  
     CATEGORY c ON sc.CategoryID = c.CategoryID  
 JOIN  
     ORDERDETAILS od ON p.ProductID = od.ProductID  
 )

SELECT  
 ProductID,  
 ProductName,  
 CategoryName,  
 OrderItems  
 FROM  
 RankedProducts  
 WHERE  
 SalesRank = 1;

# Chapter 5: Triggers and Procedure

## **1.** **Create a procedure that checks whether a customer has exceeded their credit limit before placing an order. If the customer's total outstanding balance (sum of all unpaid orders) exceeds their credit limit, the procedure will raise an exception.**

CREATE OR REPLACE PROCEDURE CreditLimitChecker(p\_CustomerID IN CUSTOMER.CustomerID%TYPE,

p\_OrderTotalValue IN ORDERS.OrderTotalValue%TYPE) IS

  dw\_TotalOutstandingBalance NUMBER(10, 2);

  dw\_CreditLimit NUMBER(10, 2);

BEGIN

  -- Get the total outstanding balance for the customer

  SELECT NVL(SUM(OD.ODTotalValue), 0) INTO dw\_TotalOutstandingBalance

  FROM ORDERS O

  JOIN ORDERDETAILS OD ON O.OrderID = OD.OrderID

  WHERE O.CustomerID = p\_CustomerID

    AND O.OrderType IN ('Online', 'Offline');

  -- Get the credit limit for the customer

  SELECT CreditLimit INTO dw\_CreditLimit

  FROM CUSTOMER

  WHERE CustomerID = p\_CustomerID;

  -- Check if the order exceeds the credit limit

  IF dw\_TotalOutstandingBalance + p\_OrderTotalValue > dw\_CreditLimit THEN

    RAISE\_APPLICATION\_ERROR(-20001, 'Order exceeds credit limit. Cannot proceed with the order.');

  ELSE

    DBMS\_OUTPUT.PUT\_LINE('Credit limit check passed. Order can be placed.');

  END IF;

EXCEPTION

  WHEN NO\_DATA\_FOUND THEN

    DBMS\_OUTPUT.PUT\_LINE('Customer not found.');

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

  RAISE;

END CreditLimitChecker;

/

**2. Create a procedure to check and update shipment status**

CREATE OR REPLACE PROCEDURE CheckAndUpdateshipmentstatus(  
    dw\_ArrivalDate IN DATE, dw\_ShipmentID IN shipment.ShipmentID%TYPE  
) AS  
   -- dw\_ShipmentID shipment.ShipmentID%TYPE;  
    dw\_ShipmentDate shipment.ShipmentDate%TYPE;  
    dw\_shipmentstatus shipment.shipmentstatus%TYPE;  
BEGIN  
    -- Check for shipment with ArrivalDate less than the provided date  
    FOR shipment\_rec IN (SELECT ShipmentID, ShipmentDate, shipmentstatus  
                         FROM shipment  
                         WHERE ArrivalDate < dw\_ArrivalDate)   
    LOOP  
       -- dw\_ShipmentID := shipment\_rec.ShipmentID;  
        dw\_ShipmentDate := shipment\_rec.ShipmentDate;  
        dw\_shipmentstatus := shipment\_rec.shipmentstatus;  
  
        -- Calculate delay in days  
        DECLARE  
            dw\_Delay NUMBER;  
        BEGIN  
            dw\_Delay := dw\_ArrivalDate - dw\_ShipmentDate;  
            IF dw\_Delay > 0 THEN  
                -- Update the shipmentstatus to 'Delayed'  
                UPDATE shipment SET  
               shipmentstatus = 'Delayed'  
                WHERE ShipmentID = dw\_ShipmentID;  
  
                -- Print information about the delay  
                DBMS\_OUTPUT.PUT\_LINE('Shipment ID ' || dw\_ShipmentID || ' has been delayed by ' || dw\_Delay || ' days.');  
            END IF;  
        END;  
    END LOOP;  
END CheckAndUpdateshipmentstatus;  
/

**3. Create a procedure that simulates applying a discount rate for products in a specific category during a special occasion. The procedure will take a CategoryID, a DiscountPercentage for the special occasion, and a StartDate for the special occasion. It will then apply the discount to products in the specified category, but only if the special occasion is ongoing.**

/\*Create a procedure that simulates applying a discount rate for products in a specific category during a special occasion.

The procedure will take a CategoryID, a DiscountPercentage for the special occasion, and a StartDate for the special occasion.

It will then apply the discount to products in the specified category, but only if the special occasion is ongoing.\*/

CREATE OR REPLACE PROCEDURE ApplyDiscountForSpecialOccasion(

  p\_CategoryID IN CATEGORY.CategoryID%TYPE,

  p\_DiscountPercentage IN NUMBER,

  p\_StartDate IN DATE

) IS

  -- Declare cursor

  CURSOR ProductCursor IS

    SELECT P.ProductID, P.ProductName, P.UnitPrice

    FROM PRODUCT P

    JOIN SUBCATEGORY SC ON P.subCategoryID = SC.subCategoryID

    WHERE SC.CategoryID = p\_CategoryID;

  -- Declare variables

  dw\_ProductID PRODUCT.ProductID%TYPE;

  dw\_ProductName PRODUCT.ProductName%TYPE;

  dw\_OriginalUnitPrice PRODUCT.UnitPrice%TYPE;

  dw\_DiscountedUnitPrice PRODUCT.UnitPrice%TYPE;

  -- Variable to check if the category exists

  dw\_CategoryExists NUMBER := 0;

BEGIN

  -- Check if the provided CategoryID exists

  SELECT COUNT(\*)

  INTO dw\_CategoryExists

  FROM CATEGORY

  WHERE CategoryID = p\_CategoryID;

  -- If the CategoryID doesn't exist, raise an exception

  IF dw\_CategoryExists = 0 THEN

    RAISE\_APPLICATION\_ERROR(-20002, 'Invalid or non-existent CategoryID: ' || p\_CategoryID);

  END IF;

  -- Open the cursor

  OPEN ProductCursor;

  -- Fetch the first row

  FETCH ProductCursor INTO dw\_ProductID, dw\_ProductName, dw\_OriginalUnitPrice;

  -- Loop through the cursor

  WHILE ProductCursor%FOUND LOOP

    -- Check if the special occasion is ongoing

    IF p\_StartDate <= SYSDATE THEN

      -- Calculate the discounted unit price

      dw\_DiscountedUnitPrice := dw\_OriginalUnitPrice - (dw\_OriginalUnitPrice \* p\_DiscountPercentage / 100);

      -- Update the product's unit price

      UPDATE PRODUCT

      SET UnitPrice = dw\_DiscountedUnitPrice

      WHERE ProductID = dw\_ProductID;

      -- Display update information

      DBMS\_OUTPUT.PUT\_LINE('Discount applied for Product ID: ' || dw\_ProductID);

      DBMS\_OUTPUT.PUT\_LINE('Product Name: ' || dw\_ProductName);

      DBMS\_OUTPUT.PUT\_LINE('Original Unit Price: ' || dw\_OriginalUnitPrice);

      DBMS\_OUTPUT.PUT\_LINE('Discounted Unit Price: ' || dw\_DiscountedUnitPrice);

      DBMS\_OUTPUT.PUT\_LINE('----------------------');

    ELSE

      -- Display information that the special occasion is not ongoing

      DBMS\_OUTPUT.PUT\_LINE('Special Occasion has not started yet.');

    END IF;

    -- Fetch the next row

    FETCH ProductCursor INTO dw\_ProductID, dw\_ProductName, dw\_OriginalUnitPrice;

  END LOOP;

  -- Close the cursor

  CLOSE ProductCursor;

  DBMS\_OUTPUT.PUT\_LINE('Discount application completed.');

EXCEPTION

  WHEN NO\_DATA\_FOUND THEN

    DBMS\_OUTPUT.PUT\_LINE('No products found in the specified category.');

  WHEN OTHERS THEN

    DBMS\_OUTPUT.PUT\_LINE('Error occurred: ' || SQLERRM);

    RAISE;

END ApplyDiscountForSpecialOccasion;

/

**4.Create a trigger to automate review details when rating is passed**

CREATE TRIGGER updateReviewDetails

BEFORE INSERT ON REVIEWDETAILS

FOR EACH ROW

BEGIN

DECLARE reviewTitle VARCHAR(255);

DECLARE reviewText VARCHAR(255);

-- Determine ReviewTitle and ReviewText based on the Rating

CASE NEW.Rating

WHEN 1 THEN

SET reviewTitle = 'Terrible Experience';

SET reviewText = 'Unfortunately, my experience was terrible.';

WHEN 2 THEN

SET reviewTitle = 'Not Satisfied';

SET reviewText = 'I was not satisfied with the product or service.';

WHEN 3 THEN

SET reviewTitle = 'Average Experience';

SET reviewText = 'The experience was average, neither good nor bad.';

WHEN 4 THEN

SET reviewTitle = 'Good Experience';

SET reviewText = 'I had a good experience with the product or service.';

WHEN 5 THEN

SET reviewTitle = 'Excellent Experience';

SET reviewText = 'My experience was excellent, highly recommended.';

ELSE

SET reviewTitle = 'Unknown Rating';

SET reviewText = 'The customer provided an unknown rating.';

END CASE;

-- Update ReviewTitle and ReviewText in REVIEWDETAILS

SET NEW.ReviewTitle = reviewTitle;

SET NEW.ReviewText = reviewText;

END;

# Chapter 6: User Interface

**URL :** <https://ec2-35-91-201-28.us-west-2.compute.amazonaws.com/mis531/>

**Credentials:** username: datawizards

Password: mis531

**Screenshots:**

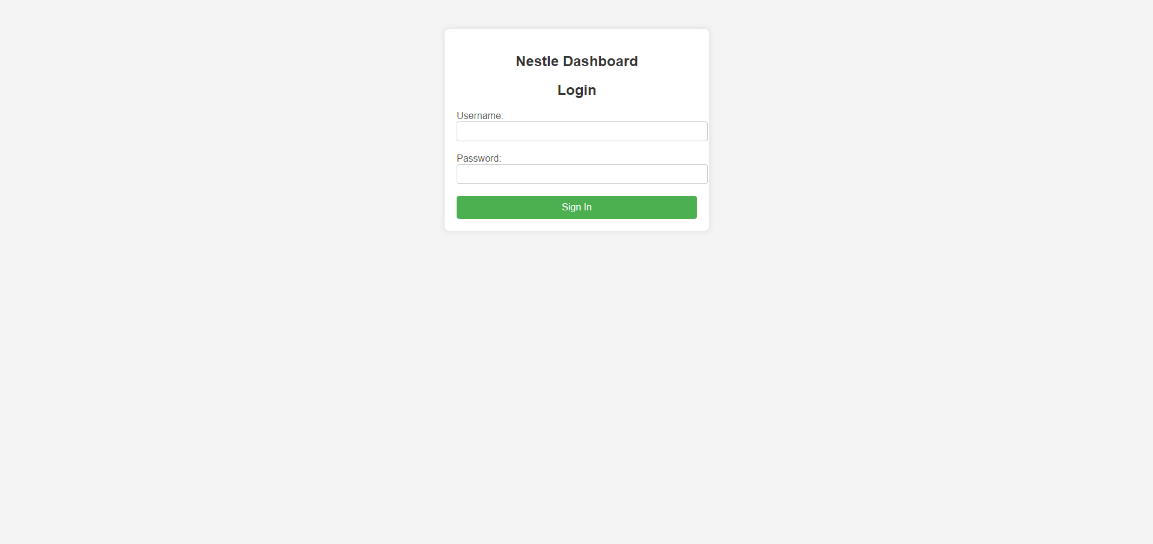
****

Figure 1: Login Page

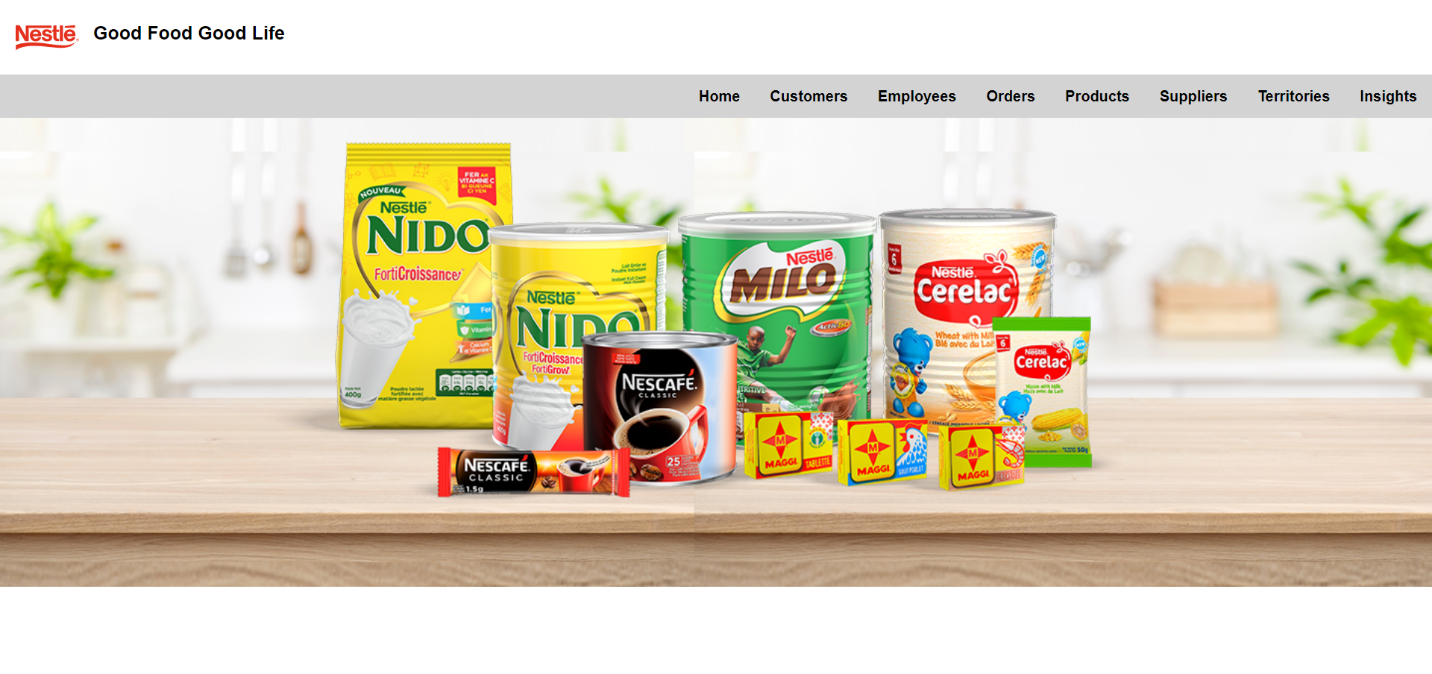


Figure 2: Homepage

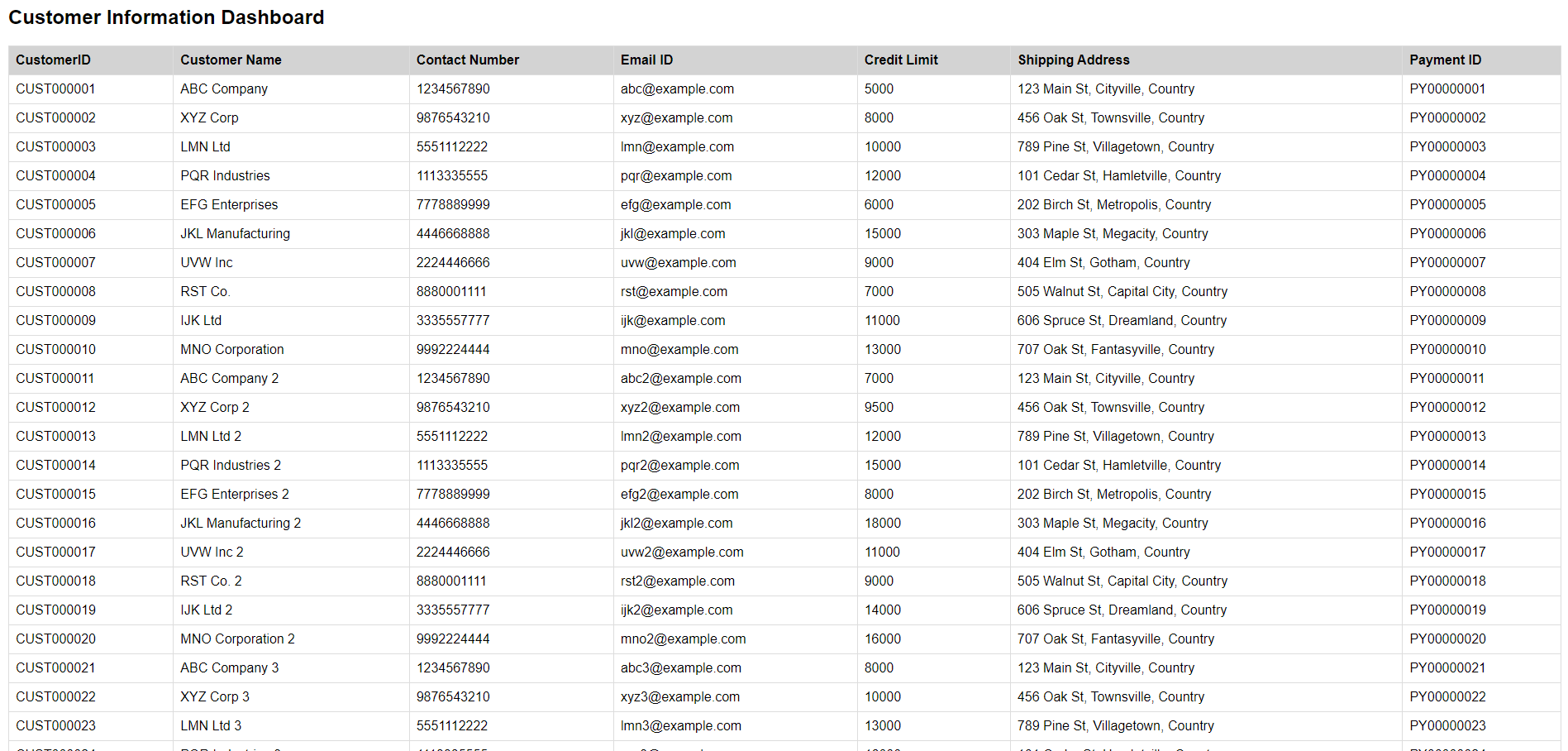


Figure 3: Customer Table(CRUD Table view)

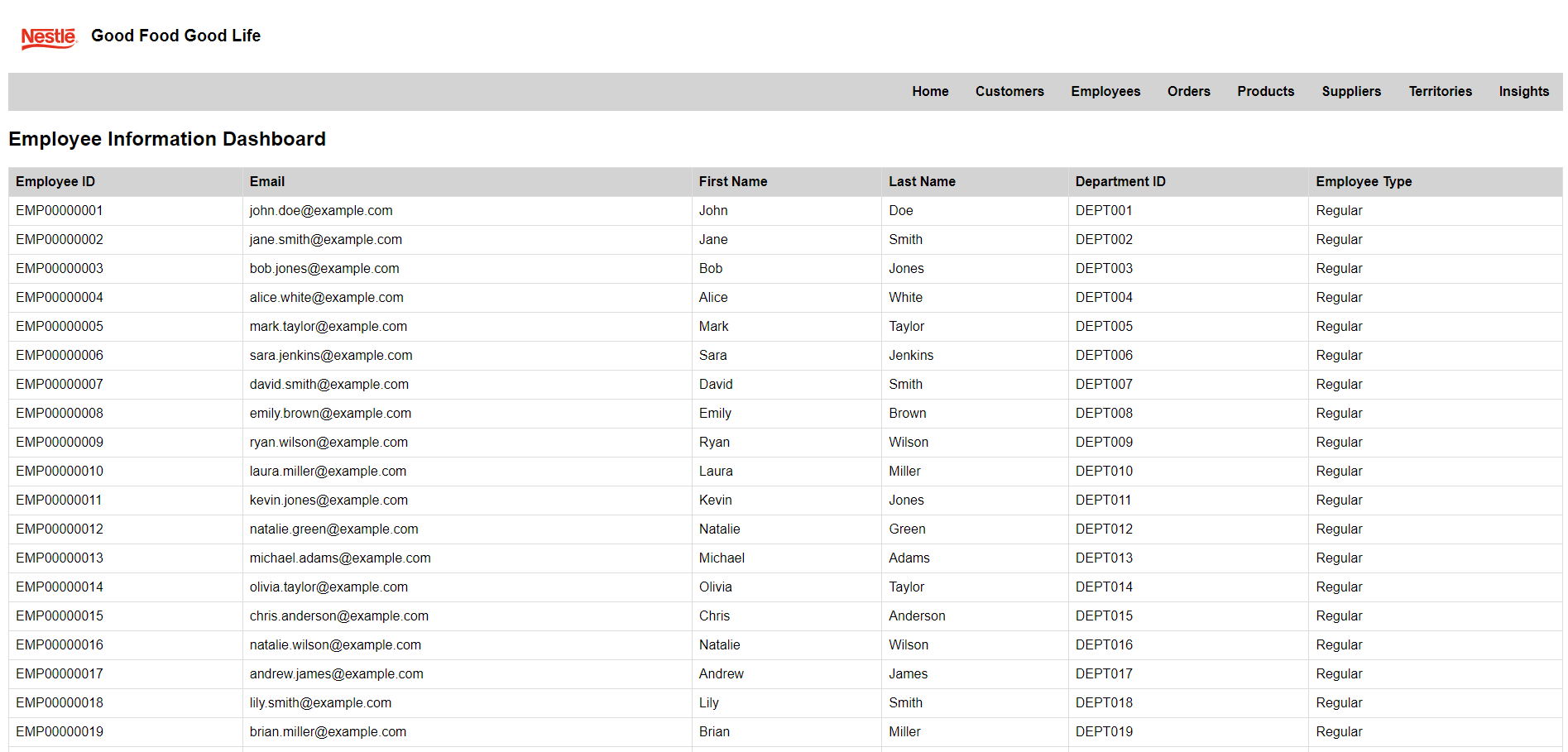


Figure 4:Employee Table

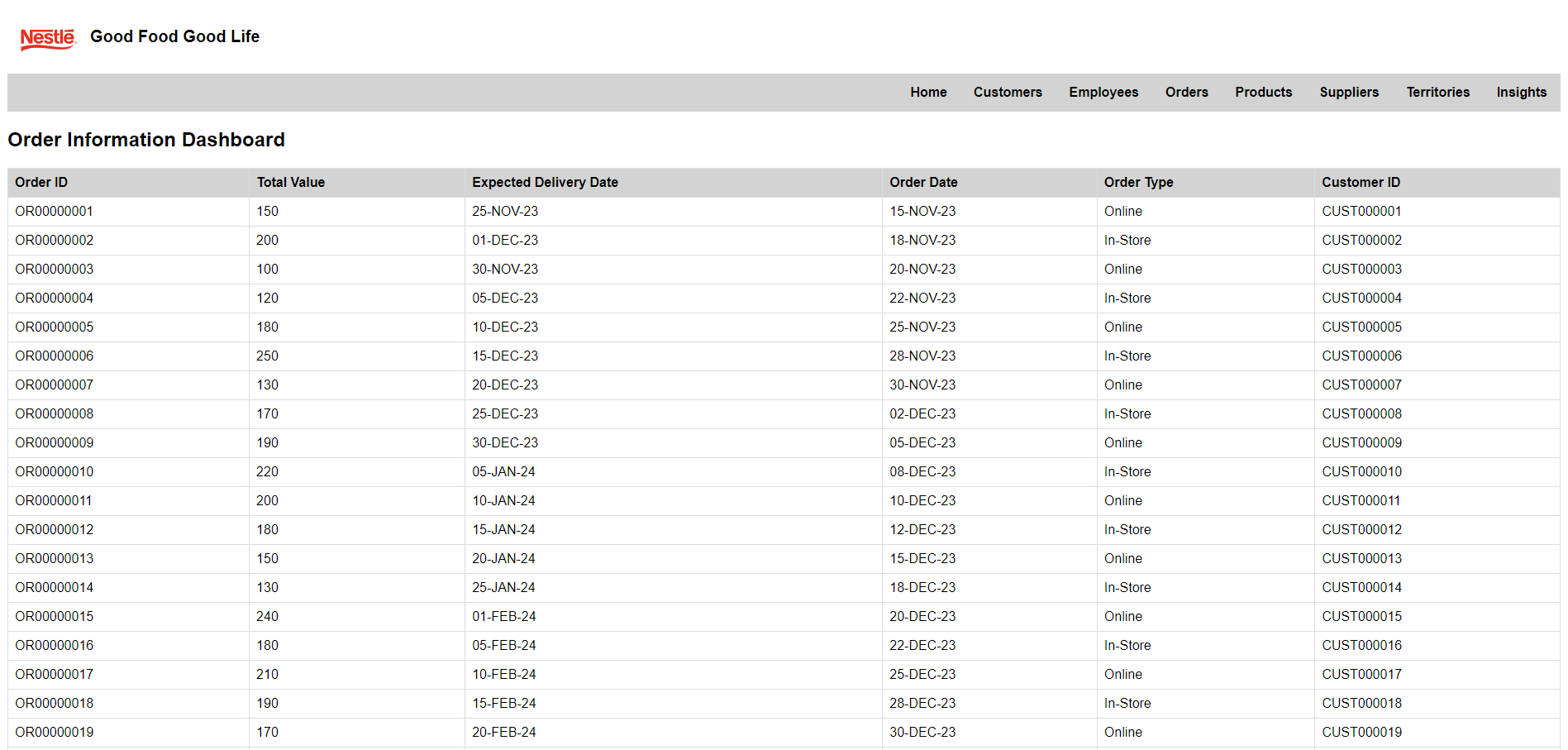


Figure 5:Order Table

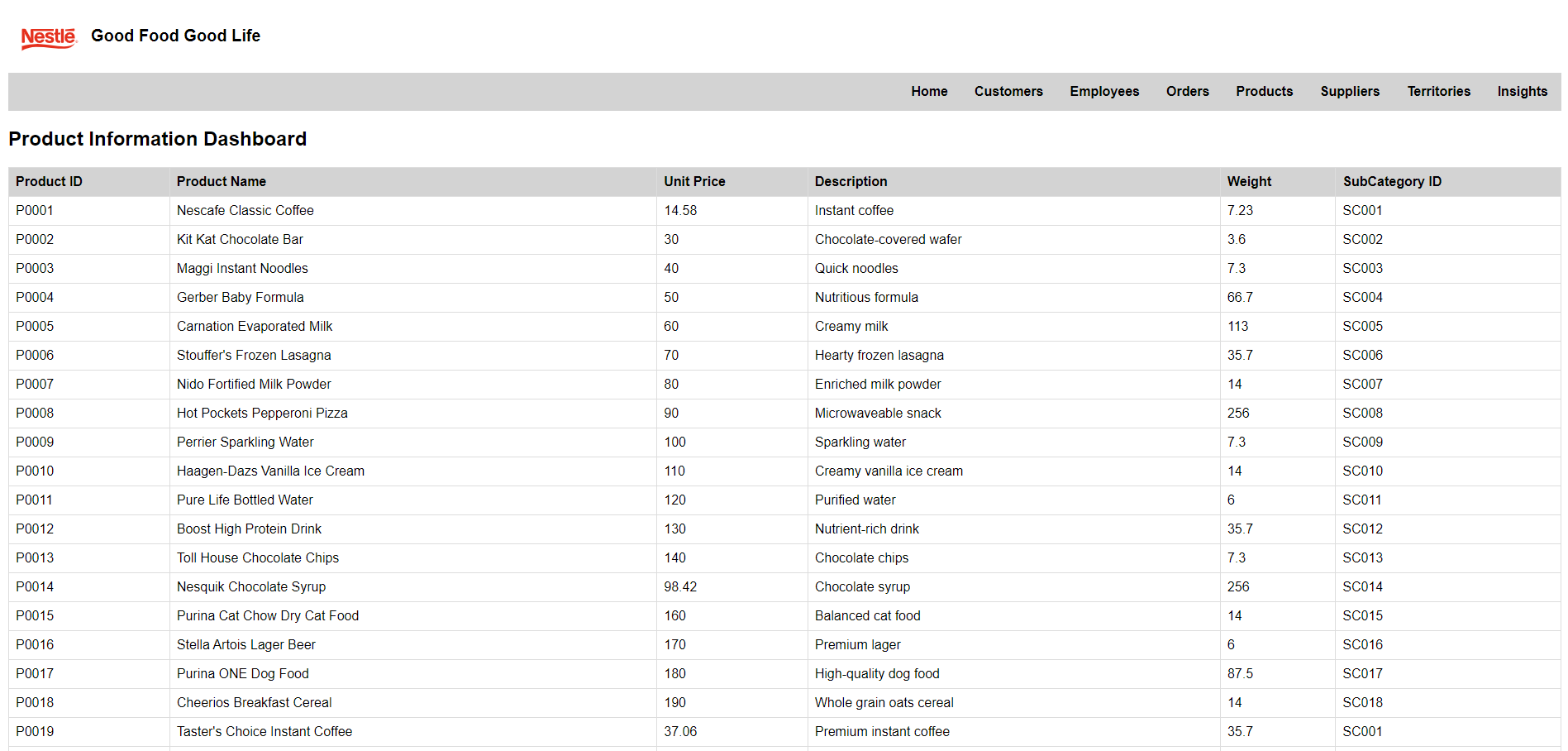
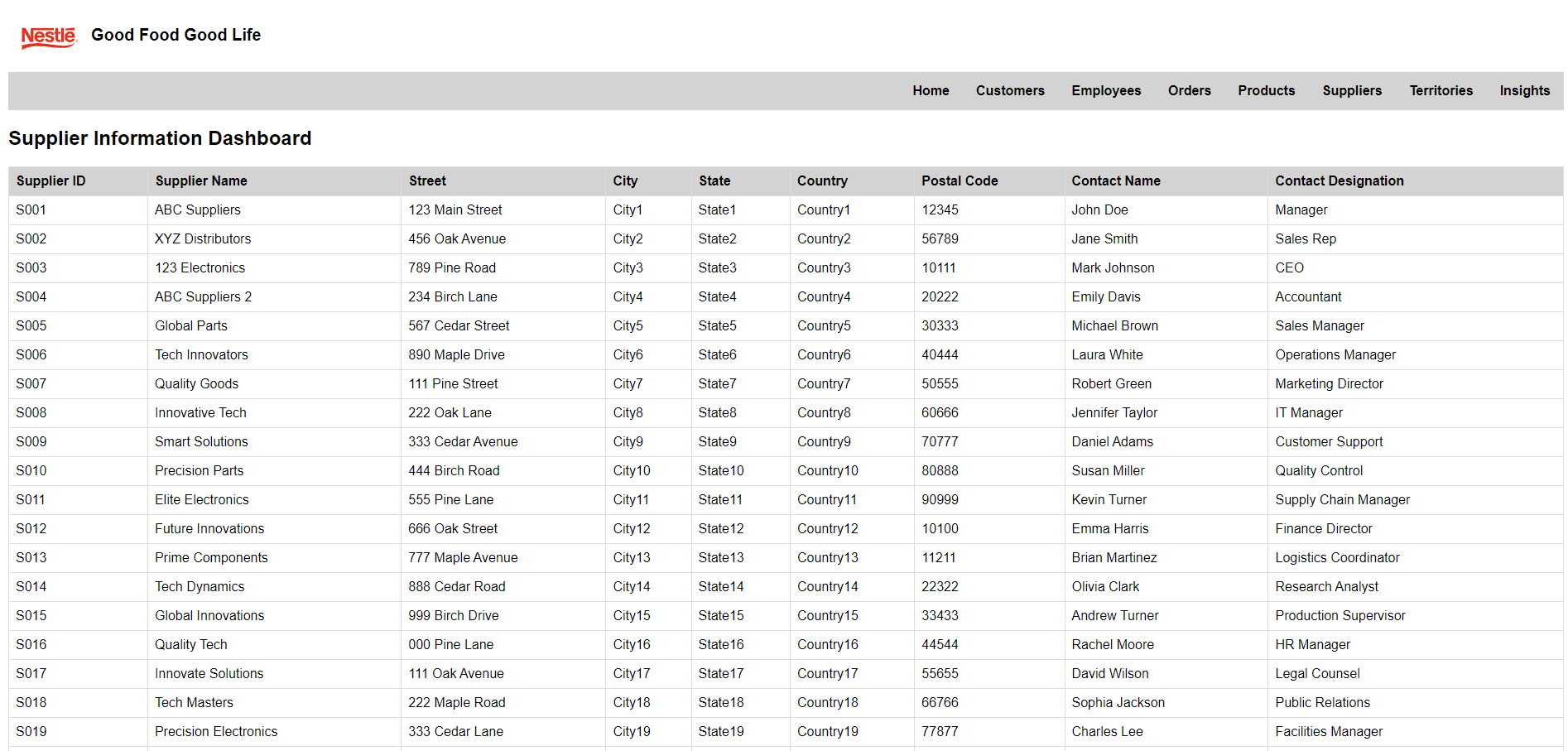


Figure 6:Product Table



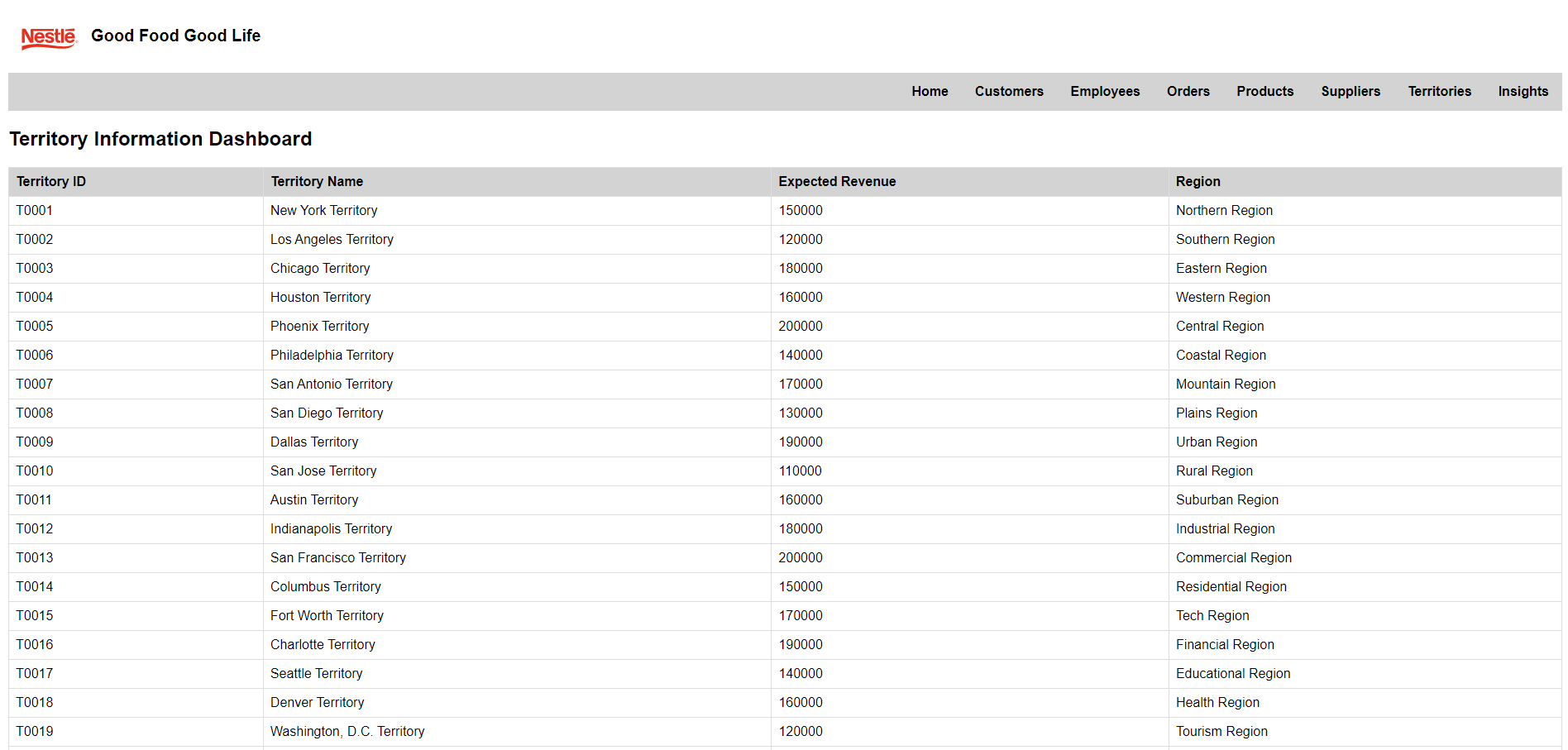
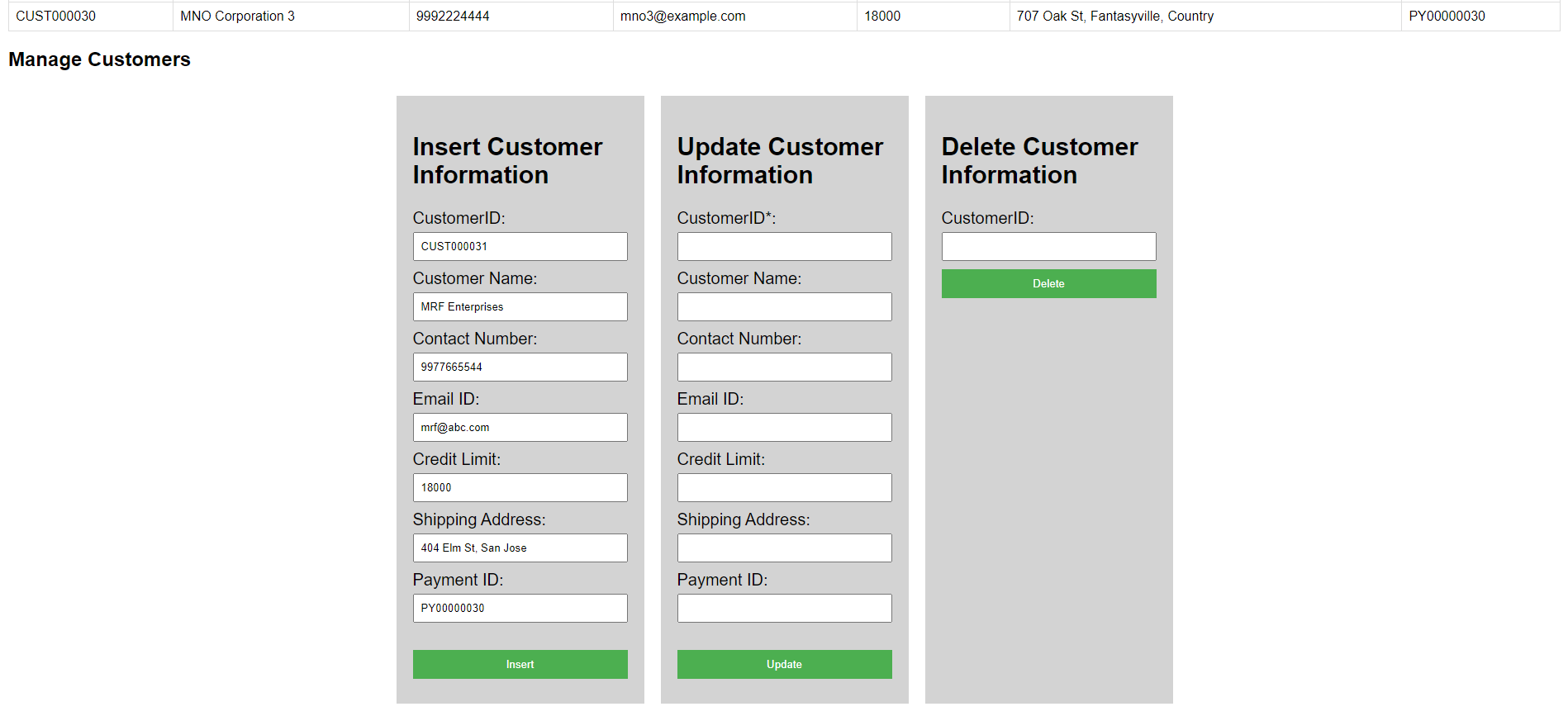
Figure 7:Supplier Table

Figure 8:Territory Table



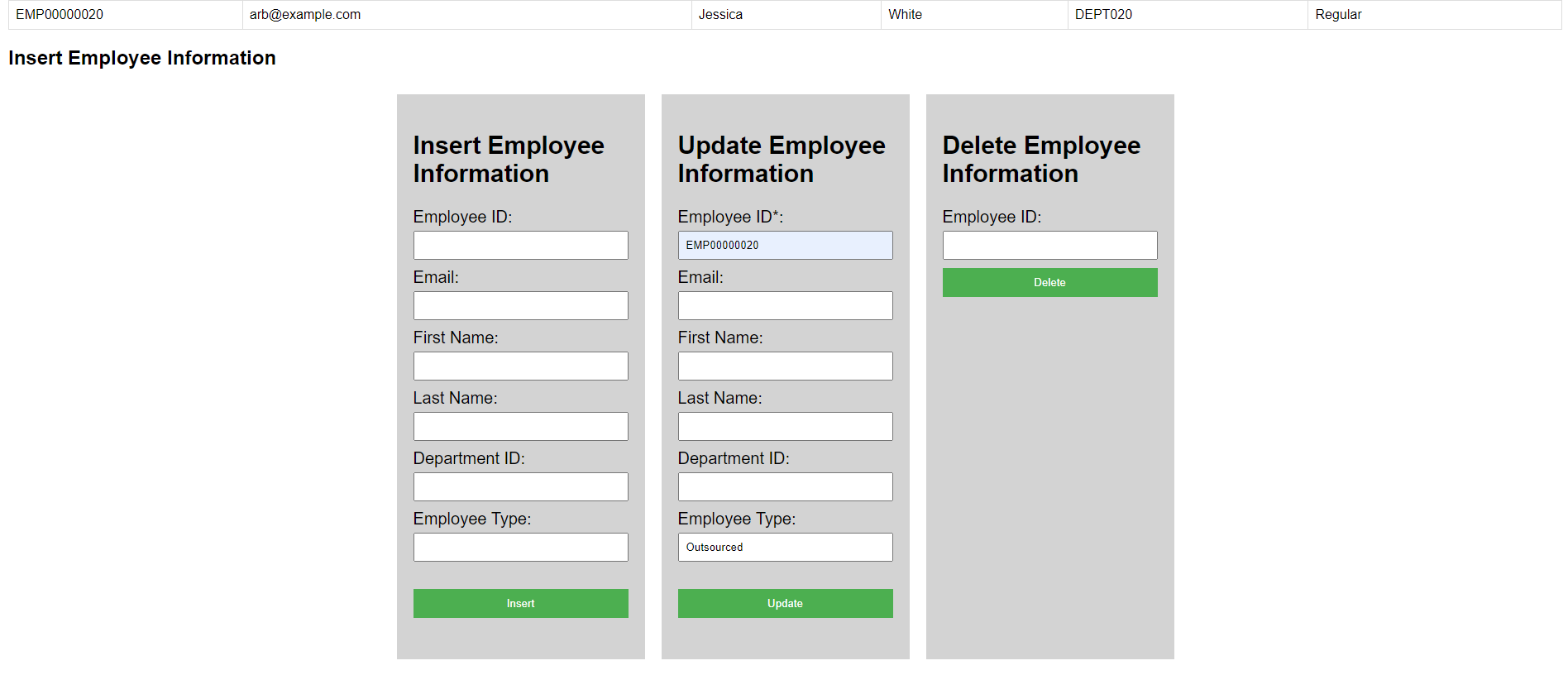
Figure 9: Inserting a new customer

Figure 10:Updating an employee's information

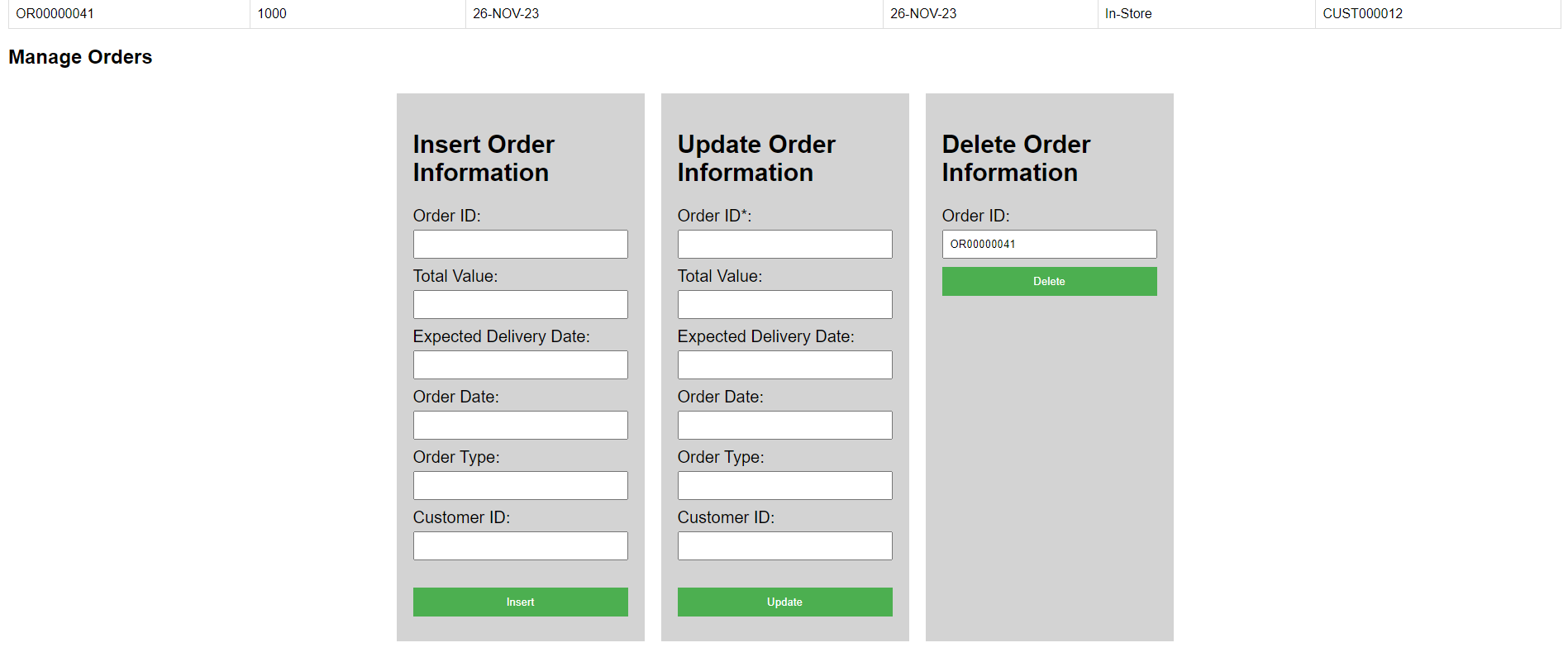


Figure 11: Deleting an order

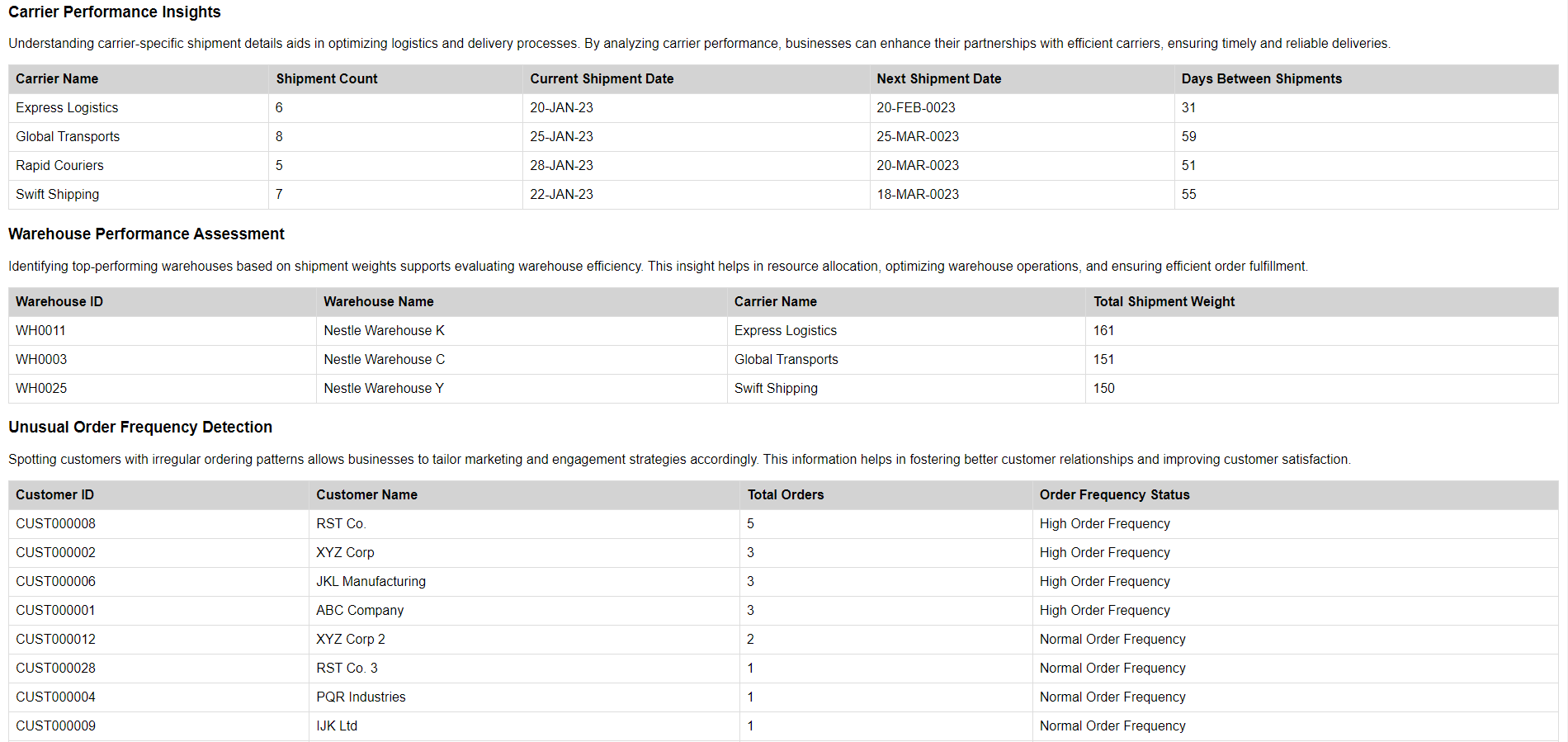
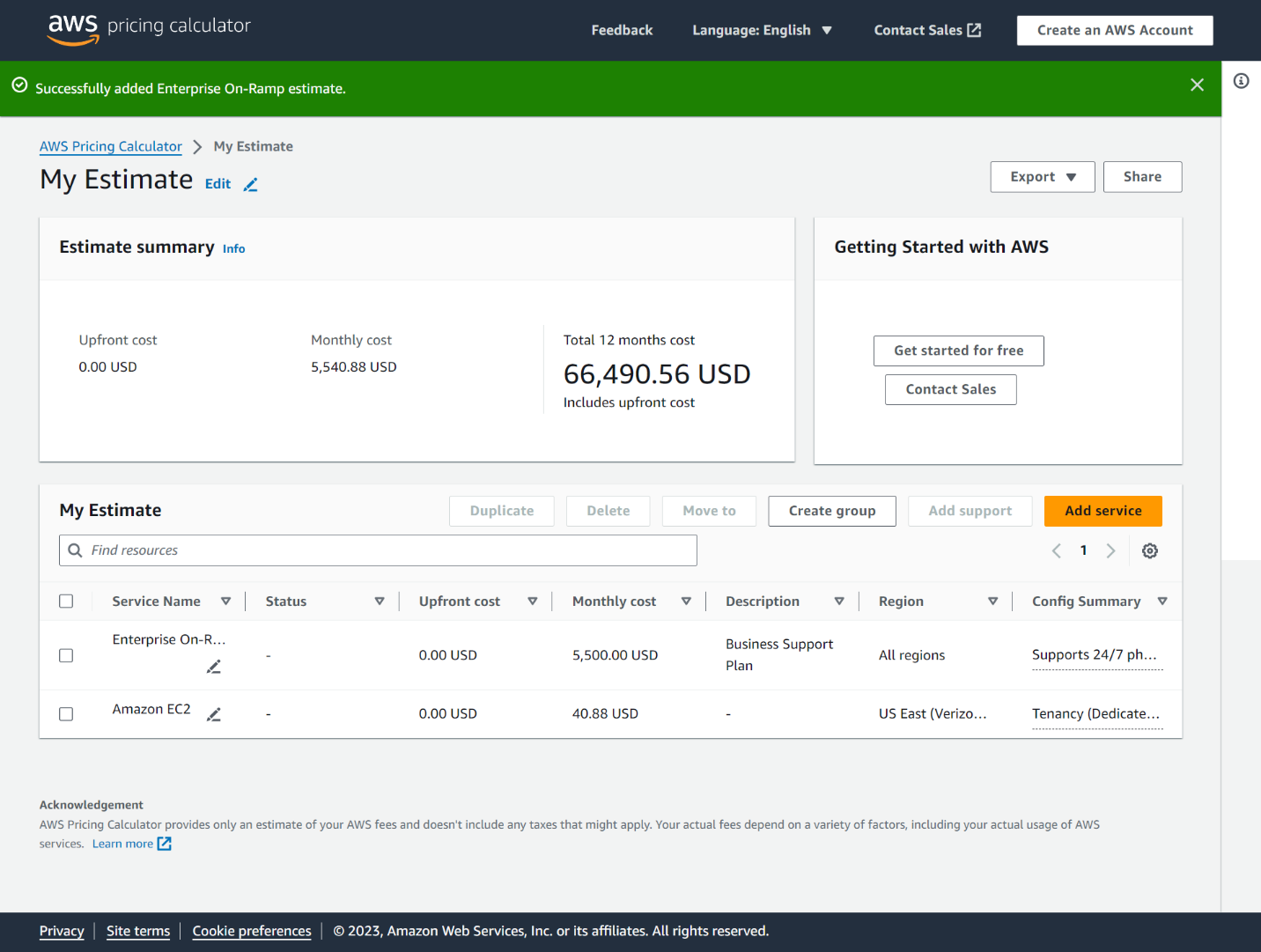


Figure 12: Insights View

# Chapter 7: Implementation Plan

We are assembling a dedicated team of seven professionals to embark on a comprehensive project involving the development and management of a database website. The team comprises key roles, including a Database Administrator, UI/UX Designer, System Developer, and Database Security Administrator.

The project is structured into four main phases: the Development Phase for initial creation, the Modification Phase for implementing changes, the Testing Phase for ensuring functionality and security, and the Maintenance Phase for ongoing support.

The estimated annual cost for this cloud data warehousing effort is $66,490.56 USD, covering essential services such as EC2 and RDS, along with hard costs including cloud instances, software licenses, load balancers, storage, and data integration tools. It's important to note that costs may be subject to fluctuations, particularly in relation to internet traffic. The entire project is projected to span six months, encompassing all the aforementioned phases. For more detailed breakdowns and specific information, please refer to the accompanying screenshots.

COCOMO (Constructive Cost Model) is a well-known technique used for estimating the cost, effort, and duration of software projects. It utilizes various parameters to calculate these estimates.

Based on the parameter assumptions and considering 8000 new lines of code, COCOMO analysis leads to:

* **Effort:** The project is estimated to require 26.2 person-months of work.
* **Schedule:** The anticipated duration for completing the project is estimated to be 10.2 months.
* **Cost:** The estimated cost for the entire project is $262,206 (This includes the cloud data warehousing cost of $66,490)

A screenshot of a computer

Description automatically generatedA graph of a column with numbers and a chart of a graph

Description automatically generated with medium confidence

# Appendix A

In reflecting on our project experience, we've gathered invaluable insights into both back-end and front-end operations, as well as the intricate process of database design and development. Throughout this project journey, we were able to practically apply the concepts and techniques we learned in our MIS 531 lectures and labs. The collective dedication and effort of our team, from selecting our client to completing the database, underscored our commitment to the project's success. Fortunately, the entire process unfolded smoothly, and we adeptly addressed challenges with a professional outlook, ensuring that setbacks didn't hinder our progress. Importantly, observing other groups during their project presentations offered us additional lessons, from structuring PowerPoint presentations effectively to communicating content clearly and managing time efficiently among presenters. Furthermore, we gleaned essential teamwork lessons—starting a project early, proactive planning, regular meetings for project tracking, and maintaining task transparency among team members. Above all, the paramount lesson learned was the importance of seeking assistance when faced with prolonged challenges or roadblocks.

# References

AWS Pricing Calculator: [Add service - AWS Pricing Calculator](https://calculator.aws/#/addService)

COCOMO: http://softwarecost.org/tools/COCOMO/

Nestle: [Nestlé: Good food, Good life | Nestlé Global (nestle.com)](https://www.nestle.com/)

Currim, F., Snodgrass, R. T., Jensen, C. S., Dyreson, C., Zhao, H., Zhang, L., Zhao, L., & Currim, S. (n.d.). Conversion from the Er to the Relational Model. Reading.

Currim, F., Snodgrass, R. T., Jensen, C. S., Dyreson, C., Zhao, H., Zhang, L., Zhao, L., & Currim, S. (n.d.). SQL: Structured Query Language. Reading.