THE DATA DYNAMOS GROUP

Prepared for

Enterprise Data Management

University of Arizona

Adwait Minde, Ojas Pawar, Aswath Nambiar, Nimish Basu

TABLE OF CONTENTS

TABLE OF CONTENTS 2
Chapter 1: Requirements Analysis 3
Chapter: Conceptual Schema 5
    Data Dictionary(Conceptual / for ER Modeling) 6
Chapter 3: Relational Schema 15
    Relational Schema 15
    Data Dictionary (Relational) 18
    DDL Appendix 40
Chapter 4: SQL Queries 49
Chapter 6: User Interface 58
Chapter 7: Implementation Plan 63
Appendix A 64
References 64

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.

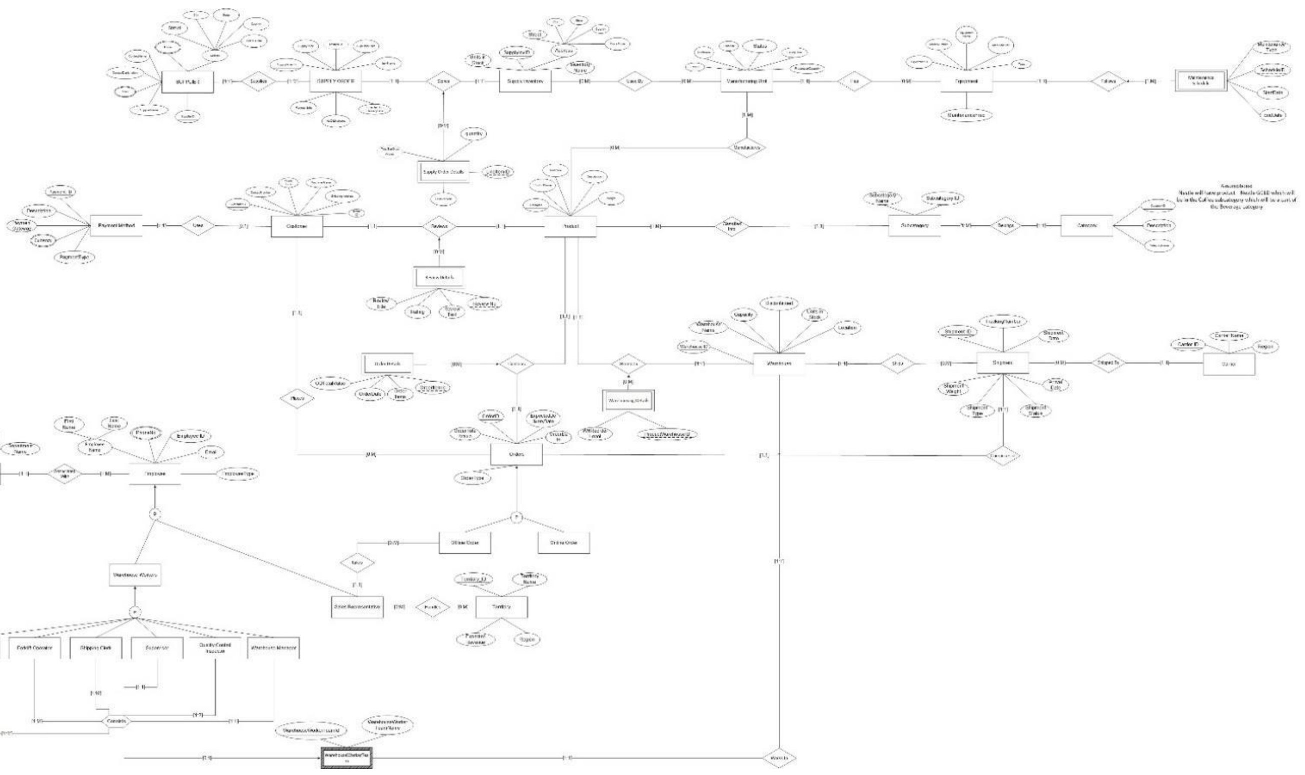
Sales representatives, like SalesRepId 501 associated with "North Region," are responsible for managing customer relationships, as seen in CustomerId 601 representing "ABC Retailers." Customer details, including CreditLimit and ShippingAddress, are stored alongside attributes like CustomerName and ContactNumber. 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	

2. 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	

3. 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	

4. 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	

5. Category

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

6. SubCategory

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

7. 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	

8. 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	

9. 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	

10. Carrier

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

11. 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	

12. 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

13. WarehouseWorkerTeam

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

14. 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	

15. 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.

16. 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	

17. 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	

18. 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	

19. 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

20. 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	

21. 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	

22. 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	

23. 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)

   a. **SUPPLIER_PHONE** (SupplierID, PhoneNo)

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

   b. **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)

- **a. OFFLINEORDER** (OrderID, EmployeeID)

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

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

- **b. 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)

- **a. SALESREPRESENTATIVE** (EmployeeID)

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

- **b. 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)

1. 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		

2. 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		

3. CONSISTS_PACKER

Schema Construct	Data Type	Constraint
EmployeeID	VARCHAR 2	Primary Key Foreign Key References EMPLOYEE(EmployeeID)

WarehouseWorkerTeamID	VARCHAR 2	Primary Key Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

4. CONSISTS_FORKLIFTOPERATOR

Schema Construct	Data Type	Constraint
EmployeeID	VARCHAR 2	Primary Key Foreign Key References EMPLOYEE(EmployeeID)
WarehouseWorkerTeamID	VARCHAR 2	Primary Key Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

5. CONSISTS_SHIPPINGCLERK

Schema Construct	Data Type	Constraint
EmployeeID	VARCHAR 2	Primary Key Foreign Key References EMPLOYEE(EmployeeID)
WarehouseWorkerTeamID	VARCHAR 2	Primary Key Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

6. CONSISTS_SUPERVISOR

Schema Construct	Data Type	Constraint
EmployeeID	VARCHAR 2	Primary Key Foreign Key References EMPLOYEE(EmployeeID)
WarehouseWorkerTeamID	VARCHAR 2	Primary Key Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

7. CONSISTS_QUALITYCONTROLINSPECTOR

Schema Construct	Data Type	Constraint
EmployeeID	VARCHAR 2	Primary Key Foreign Key References EMPLOYEE(EmployeeID)
WarehouseWorkerTeamID	VARCHAR 2	Primary Key Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

8. CONSISTS_WAREHOUSEMANAGER

Schema Construct	Data Type	Constraint
EmployeeID	VARCHAR 2	Primary Key Foreign Key References EMPLOYEE(EmployeeID)

WarehouseWorkerTeamID	VARCHAR2	Primary Key Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamID)

9. 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		

10. 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		

11. 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

12. EMPLOYEEPHONE

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

13. SALESREPRESENTATIVE

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

14. WAREHOUSEWORKERS

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

15. 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		

16. HANDLES

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

17. 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		

18. 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		

19. 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		

20. 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		

21. 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		

22. ONLINEORDER

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

23. 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		

24. 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		

25. PAYMENTCURRENCY

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

26. 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		

27. 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		

28. 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		

29. 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		

30. 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		

31. 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		

32. 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		

33. 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		

34. 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		

35. 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		

36. 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		

37. 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		

38. WAREHOUSE

Schema Construct	Data Type	Constraint

WarehouseId	VARCHAR 2	Primary Key Format begins with WH followed by 4 digits
UnitsInOrder	NUMBER	Not Null
WarehouseName	VARCHAR 2	
Capacity	NUMBER	
Discontinued	CHAR	
Location	VARCHAR 2	
WarehouseWorkerTeamI D	VARCHAR 2	Foreign Key References WAREHOUSEWORKERTEAM(WarehouseWorkerTeamI D)
FD:WarehouseId->UnitsInOrder, WarehouseName, Capacity, Discontinued, Location, WarehouseWorkerTeamID		

39. 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		

40. 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;
```

2. **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;
```

3. **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;
```

4. **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;
```

5. **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;
```

6. **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;
```

7. **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;
```

8. **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;
```

9. 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;
```

10. 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;
```

11. 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;
```

12. 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:

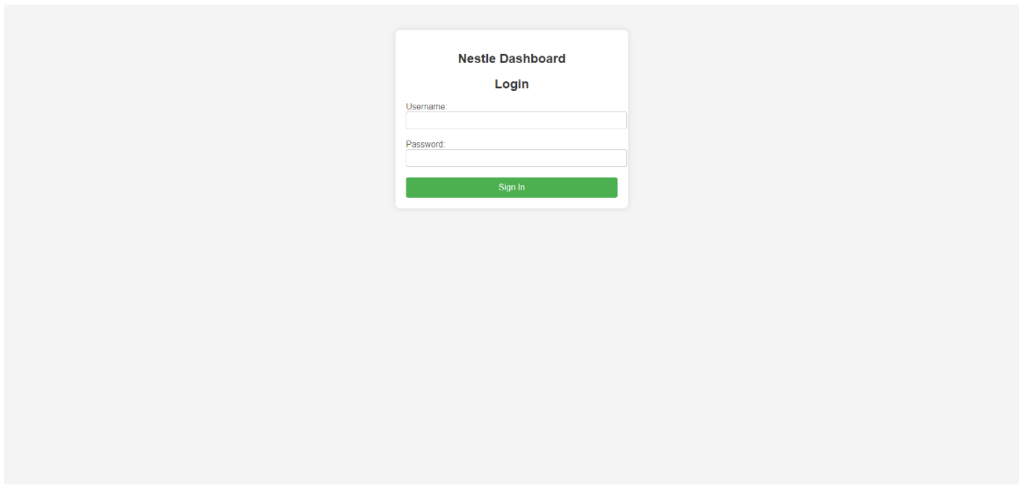


Figure 1: Login Page

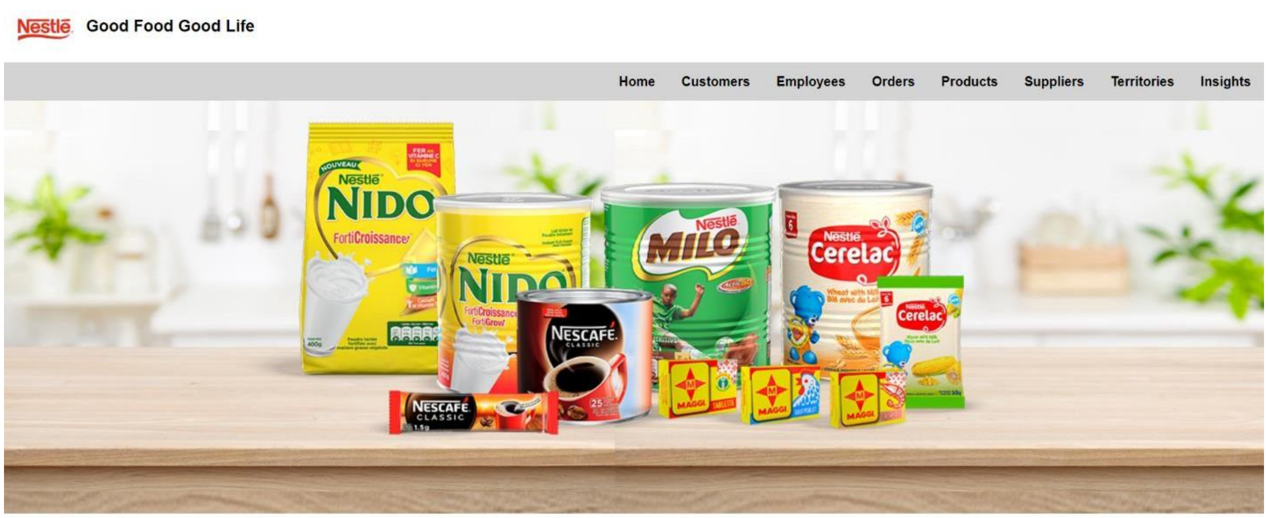
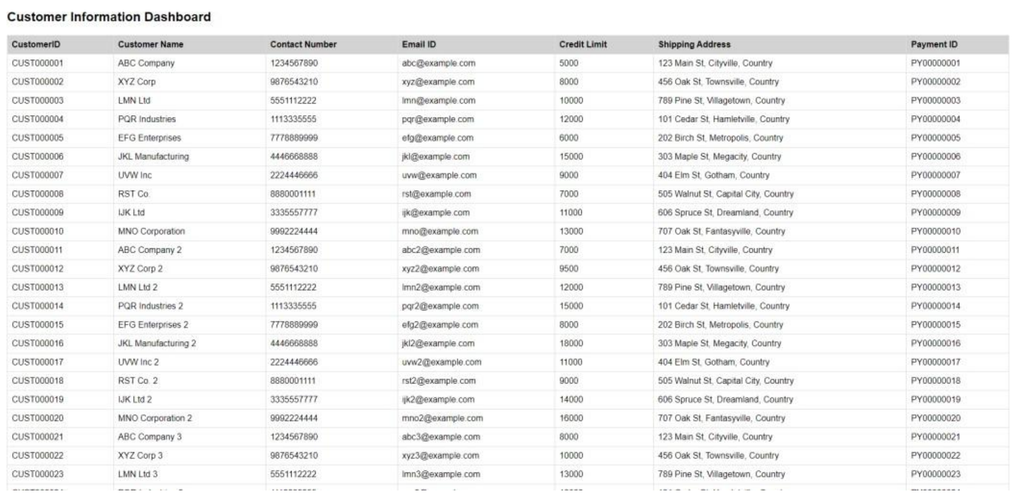


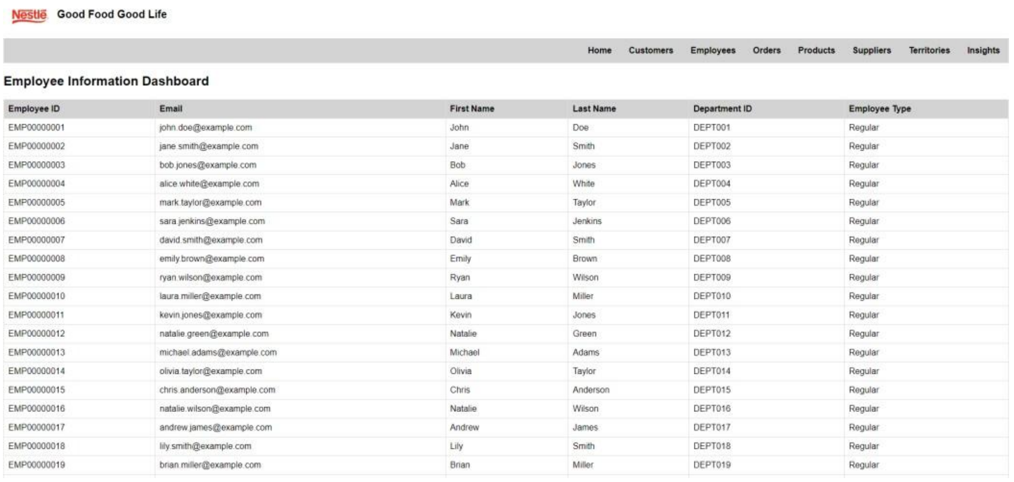
Figure 2: Homepage



Customer Information Dashboard

CustomerID	Customer Name	Contact Number	Email ID	Credit Limit	Shipping Address	Payment ID
CUST000001	ABC Company	1234567890	abc@example.com	5000	123 Main St, Cityville, Country	PY00000001
CUST000002	XYZ Corp	9876543210	xyz@example.com	8000	456 Oak St, Townsville, Country	PY00000002
CUST000003	LMN Ltd	5551112222	lmn@example.com	10000	789 Pine St, Villagetown, Country	PY00000003
CUST000004	PQR Industries	1113335555	pqr@example.com	12000	101 Cedar St, Hamletville, Country	PY00000004
CUST000005	EFG Enterprises	7778889999	efg@example.com	6000	202 Birch St, Metropolis, Country	PY00000005
CUST000006	JKL Manufacturing	4446668888	jkl@example.com	15000	303 Maple St, Megacity, Country	PY00000006
CUST000007	UVW Inc	2224446666	uvw@example.com	9000	404 Elm St, Gotham, Country	PY00000007
CUST000008	RST Co.	8880001111	rst@example.com	7000	505 Walnut St, Capital City, Country	PY00000008
CUST000009	IJK Ltd	3335557777	ijk@example.com	11000	606 Spruce St, Dreamland, Country	PY00000009
CUST000010	MNO Corporation	9992224444	mno@example.com	13000	707 Oak St, Fantasyville, Country	PY00000010
CUST000011	ABC Company 2	1234567890	abc2@example.com	7000	123 Main St, Cityville, Country	PY00000011
CUST000012	XYZ Corp 2	9876543210	xyz2@example.com	9500	456 Oak St, Townsville, Country	PY00000012
CUST000013	LMN Ltd 2	5551112222	lmn2@example.com	12000	789 Pine St, Villagetown, Country	PY00000013
CUST000014	PQR Industries 2	1113335555	pqr2@example.com	15000	101 Cedar St, Hamletville, Country	PY00000014
CUST000015	EFG Enterprises 2	7778889999	efg2@example.com	8000	202 Birch St, Metropolis, Country	PY00000015
CUST000016	JKL Manufacturing 2	4446668888	jkl2@example.com	18000	303 Maple St, Megacity, Country	PY00000016
CUST000017	UVW Inc 2	2224446666	uvw2@example.com	11000	404 Elm St, Gotham, Country	PY00000017
CUST000018	RST Co. 2	8880001111	rst2@example.com	9000	505 Walnut St, Capital City, Country	PY00000018
CUST000019	IJK Ltd 2	3335557777	ijk2@example.com	14000	606 Spruce St, Dreamland, Country	PY00000019
CUST000020	MNO Corporation 2	9992224444	mno2@example.com	16000	707 Oak St, Fantasyville, Country	PY00000020
CUST000021	ABC Company 3	1234567890	abc3@example.com	8000	123 Main St, Cityville, Country	PY00000021
CUST000022	XYZ Corp 3	9876543210	xyz3@example.com	10000	456 Oak St, Townsville, Country	PY00000022
CUST000023	LMN Ltd 3	5551112222	lmn3@example.com	13000	789 Pine St, Villagetown, Country	PY00000023

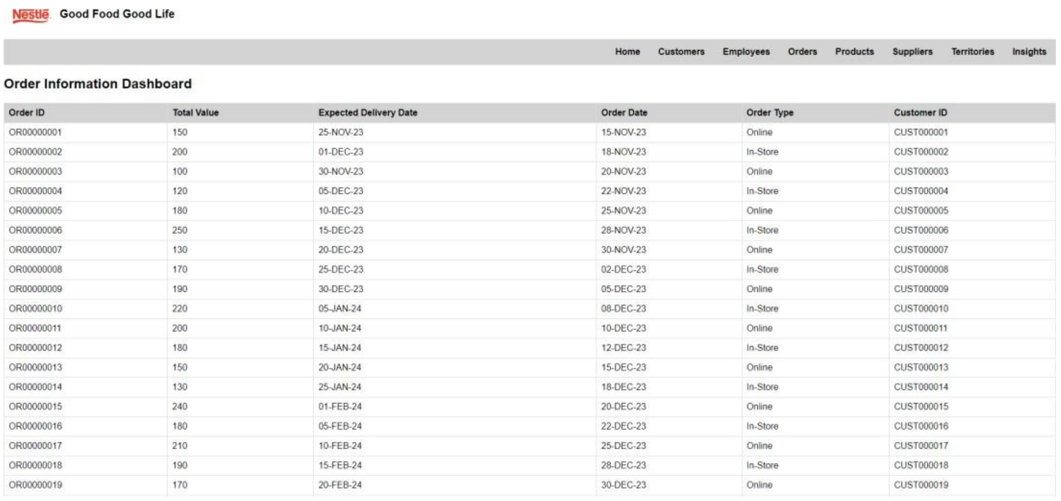
Figure 3: Customer Table(CRUD Table view)



Employee Information Dashboard

Employee ID	Email	First Name	Last Name	Department ID	Employee Type
EMP00000001	john.doe@example.com	John	Doe	DEPT001	Regular
EMP00000002	jane.smith@example.com	Jane	Smith	DEPT002	Regular
EMP00000003	bob.jones@example.com	Bob	Jones	DEPT003	Regular
EMP00000004	alice.white@example.com	Alice	White	DEPT004	Regular
EMP00000005	mark.taylor@example.com	Mark	Taylor	DEPT005	Regular
EMP00000006	sara.jenkins@example.com	Sara	Jenkins	DEPT006	Regular
EMP00000007	david.smith@example.com	David	Smith	DEPT007	Regular
EMP00000008	emily.brown@example.com	Emily	Brown	DEPT008	Regular
EMP00000009	ryan.wilson@example.com	Ryan	Wilson	DEPT009	Regular
EMP00000010	laura.miller@example.com	Laura	Miller	DEPT010	Regular
EMP00000011	kevin.jones@example.com	Kevin	Jones	DEPT011	Regular
EMP00000012	natalie.green@example.com	Natalie	Green	DEPT012	Regular
EMP00000013	michael.adams@example.com	Michael	Adams	DEPT013	Regular
EMP00000014	olivia.taylor@example.com	Olivia	Taylor	DEPT014	Regular
EMP00000015	chris.anderson@example.com	Chris	Anderson	DEPT015	Regular
EMP00000016	natalie.wilson@example.com	Natalie	Wilson	DEPT016	Regular
EMP00000017	andrew.james@example.com	Andrew	James	DEPT017	Regular
EMP00000018	lily.smith@example.com	Lily	Smith	DEPT018	Regular
EMP00000019	brian.miller@example.com	Brian	Miller	DEPT019	Regular

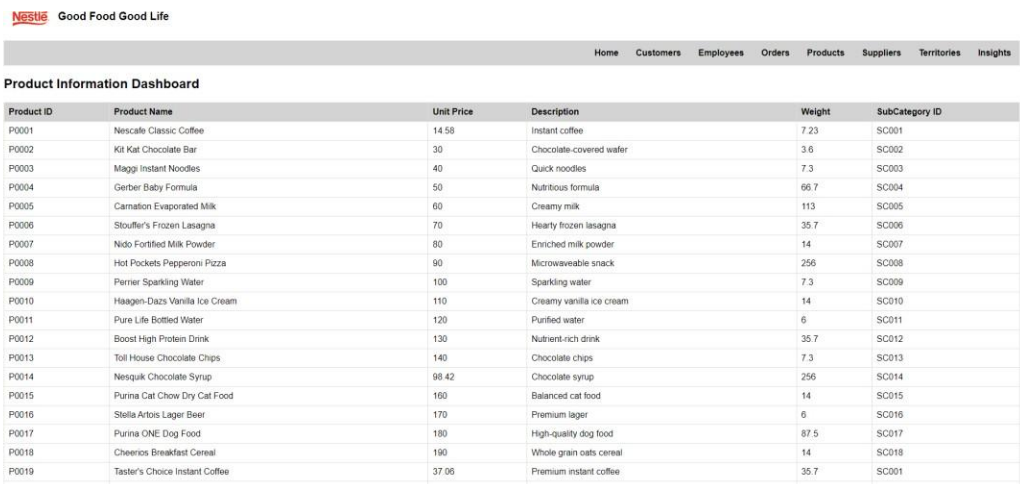
Figure 4:Employee Table



Order Information Dashboard

Order ID	Total Value	Expected Delivery Date	Order Date	Order Type	Customer ID
OR00000001	150	25-NOV-23	15-NOV-23	Online	CUST000001
OR00000002	200	01-DEC-23	18-NOV-23	In-Store	CUST000002
OR00000003	100	30-NOV-23	20-NOV-23	Online	CUST000003
OR00000004	120	05-DEC-23	22-NOV-23	In-Store	CUST000004
OR00000005	180	10-DEC-23	25-NOV-23	Online	CUST000005
OR00000006	250	15-DEC-23	28-NOV-23	In-Store	CUST000006
OR00000007	130	20-DEC-23	30-NOV-23	Online	CUST000007
OR00000008	170	25-DEC-23	02-DEC-23	In-Store	CUST000008
OR00000009	190	30-DEC-23	05-DEC-23	Online	CUST000009
OR00000010	220	05-JAN-24	08-DEC-23	In-Store	CUST000010
OR00000011	200	10-JAN-24	10-DEC-23	Online	CUST000011
OR00000012	180	15-JAN-24	12-DEC-23	In-Store	CUST000012
OR00000013	150	20-JAN-24	15-DEC-23	Online	CUST000013
OR00000014	130	25-JAN-24	18-DEC-23	In-Store	CUST000014
OR00000015	240	01-FEB-24	20-DEC-23	Online	CUST000015
OR00000016	180	05-FEB-24	22-DEC-23	In-Store	CUST000016
OR00000017	210	10-FEB-24	25-DEC-23	Online	CUST000017
OR00000018	190	15-FEB-24	28-DEC-23	In-Store	CUST000018
OR00000019	170	20-FEB-24	30-DEC-23	Online	CUST000019

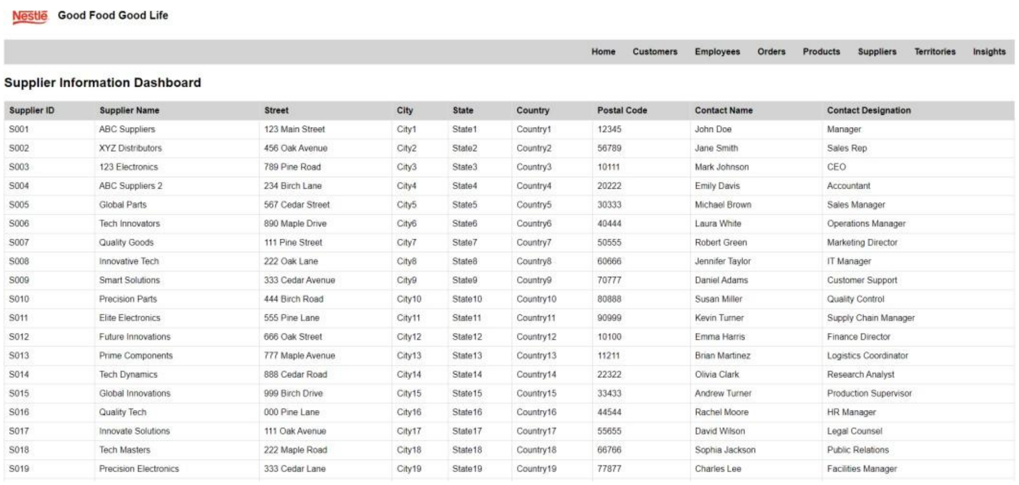
Figure 5:Order Table



Product Information Dashboard

Product ID	Product Name	Unit Price	Description	Weight	SubCategory ID
P0001	Nescafe Classic Coffee	14.58	Instant coffee	7.23	SC001
P0002	Kit Kat Chocolate Bar	30	Chocolate-covered wafer	3.6	SC002
P0003	Maggi Instant Noodles	40	Quick noodles	7.3	SC003
P0004	Gerber Baby Formula	50	Nutritious formula	66.7	SC004
P0005	Carnation Evaporated Milk	60	Creamy milk	113	SC005
P0006	Stouffer's Frozen Lasagna	70	Hearty frozen lasagna	35.7	SC006
P0007	Nido Fortified Milk Powder	80	Enriched milk powder	14	SC007
P0008	Hot Pockets Pepperoni Pizza	90	Microwaveable snack	256	SC008
P0009	Perrier Sparkling Water	100	Sparkling water	7.3	SC009
P0010	Haagen-Dazs Vanilla Ice Cream	110	Creamy vanilla ice cream	14	SC010
P0011	Pure Life Bottled Water	120	Purified water	6	SC011
P0012	Boost High Protein Drink	130	Nutrient-rich drink	35.7	SC012
P0013	Toll House Chocolate Chips	140	Chocolate chips	7.3	SC013
P0014	Nesquik Chocolate Syrup	98.42	Chocolate syrup	256	SC014
P0015	Purina Cat Chow Dry Cat Food	160	Balanced cat food	14	SC015
P0016	Stella Artois Lager Beer	170	Premium lager	6	SC016
P0017	Purina ONE Dog Food	180	High-quality dog food	87.5	SC017
P0018	Cheerios Breakfast Cereal	190	Whole grain oats cereal	14	SC018
P0019	Taster's Choice Instant Coffee	37.06	Premium instant coffee	35.7	SC001

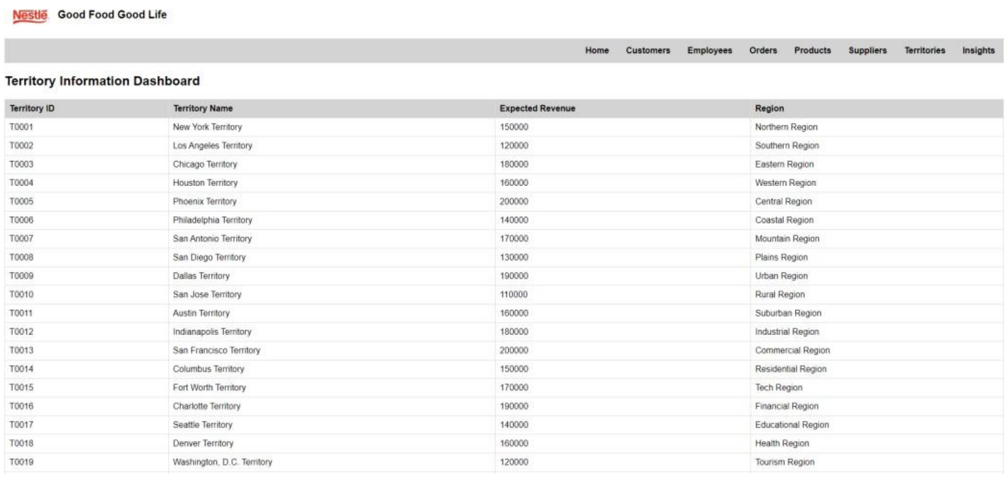
Figure 6:Product Table



Supplier Information Dashboard

Supplier ID	Supplier Name	Street	City	State	Country	Postal Code	Contact Name	Contact Designation
S001	ABC Suppliers	123 Main Street	City1	State1	Country1	12345	John Doe	Manager
S002	XYZ Distributors	456 Oak Avenue	City2	State2	Country2	56789	Jane Smith	Sales Rep
S003	123 Electronics	789 Pine Road	City3	State3	Country3	10111	Mark Johnson	CEO
S004	ABC Suppliers 2	234 Birch Lane	City4	State4	Country4	20222	Emily Davis	Accountant
S005	Global Parts	567 Cedar Street	City5	State5	Country5	30333	Michael Brown	Sales Manager
S006	Tech Innovators	890 Maple Drive	City6	State6	Country6	40444	Laura White	Operations Manager
S007	Quality Goods	111 Pine Street	City7	State7	Country7	50555	Robert Green	Marketing Director
S008	Innovative Tech	222 Oak Lane	City8	State8	Country8	60666	Jennifer Taylor	IT Manager
S009	Smart Solutions	333 Cedar Avenue	City9	State9	Country9	70777	Daniel Adams	Customer Support
S010	Precision Parts	444 Birch Road	City10	State10	Country10	80888	Susan Miller	Quality Control
S011	Elite Electronics	555 Pine Lane	City11	State11	Country11	90999	Kevin Turner	Supply Chain Manager
S012	Future Innovations	666 Oak Street	City12	State12	Country12	10100	Emma Harris	Finance Director
S013	Prime Components	777 Maple Avenue	City13	State13	Country13	11211	Brian Martinez	Logistics Coordinator
S014	Tech Dynamics	888 Cedar Road	City14	State14	Country14	22322	Olivia Clark	Research Analyst
S015	Global Innovations	999 Birch Drive	City15	State15	Country15	33433	Andrew Turner	Production Supervisor
S016	Quality Tech	000 Pine Lane	City16	State16	Country16	44544	Rachel Moore	HR Manager
S017	Innovate Solutions	111 Oak Avenue	City17	State17	Country17	55655	David Wilson	Legal Counsel
S018	Tech Masters	222 Maple Road	City18	State18	Country18	66766	Sophia Jackson	Public Relations
S019	Precision Electronics	333 Cedar Lane	City19	State19	Country19	77877	Charles Lee	Facilities Manager

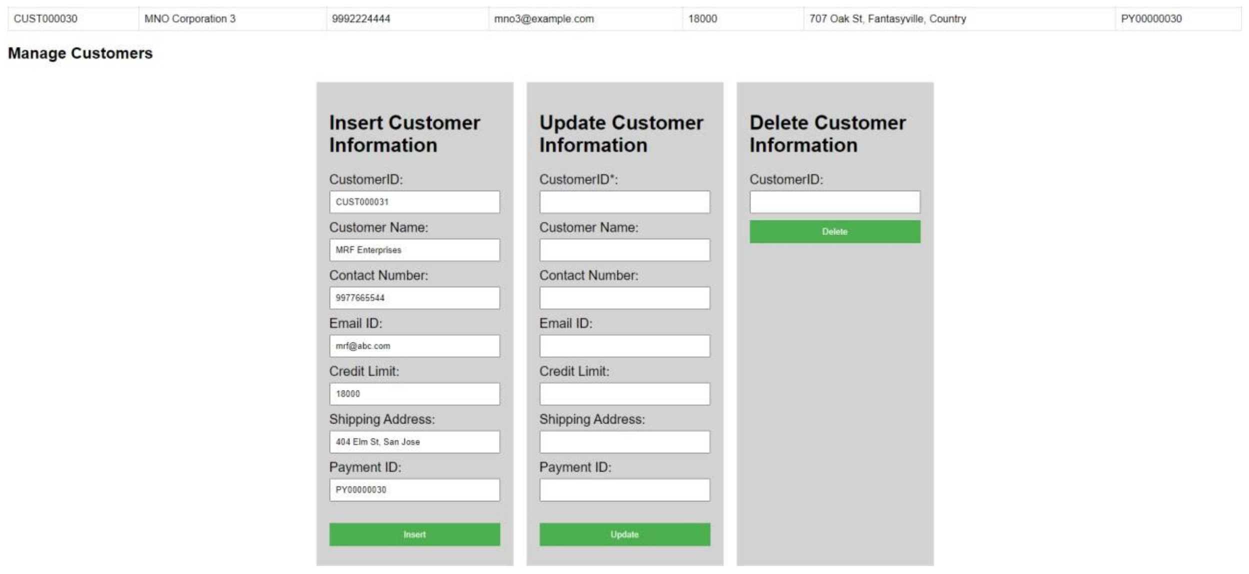
Figure 7:Supplier Table



Territory Information Dashboard

Territory ID	Territory Name	Expected Revenue	Region
T0001	New York Territory	150000	Northern Region
T0002	Los Angeles Territory	120000	Southern Region
T0003	Chicago Territory	180000	Eastern Region
T0004	Houston Territory	160000	Western Region
T0005	Phoenix Territory	200000	Central Region
T0006	Philadelphia Territory	140000	Coastal Region
T0007	San Antonio Territory	170000	Mountain Region
T0008	San Diego Territory	130000	Plains Region
T0009	Dallas Territory	190000	Urban Region
T0010	San Jose Territory	110000	Rural Region
T0011	Austin Territory	160000	Suburban Region
T0012	Indianapolis Territory	180000	Industrial Region
T0013	San Francisco Territory	200000	Commercial Region
T0014	Columbus Territory	150000	Residential Region
T0015	Fort Worth Territory	170000	Tech Region
T0016	Charlotte Territory	190000	Financial Region
T0017	Seattle Territory	140000	Educational Region
T0018	Denver Territory	160000	Health Region
T0019	Washington, D.C. Territory	120000	Tourism Region

Figure 8:Territory Table



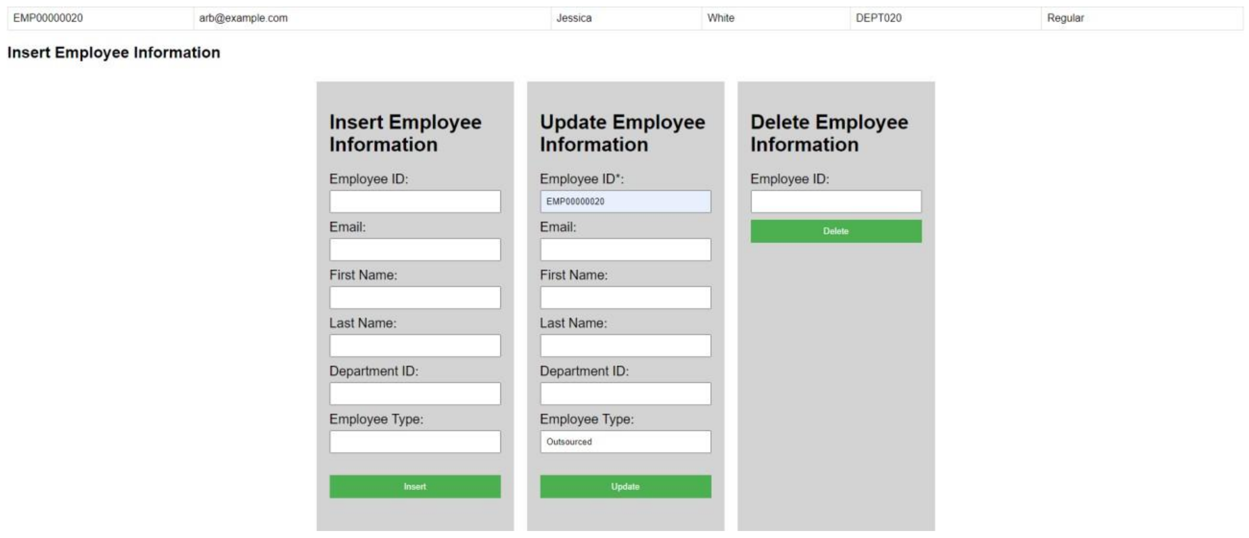
CUST000030	MNO Corporation 3	9992224444	mno3@example.com	18000	707 Oak St, Fantasyville, Country	PY00000030

Manage Customers

Insert Customer Information

CustomerID:	CUST000031
Customer Name:	MRF Enterprises
Contact Number:	9977665544
Email ID:	mrf@abc.com
Credit Limit:	18000
Shipping Address:	404 Elm St, San Jose
Payment ID:	PY00000030

Figure 9: Inserting a new customer



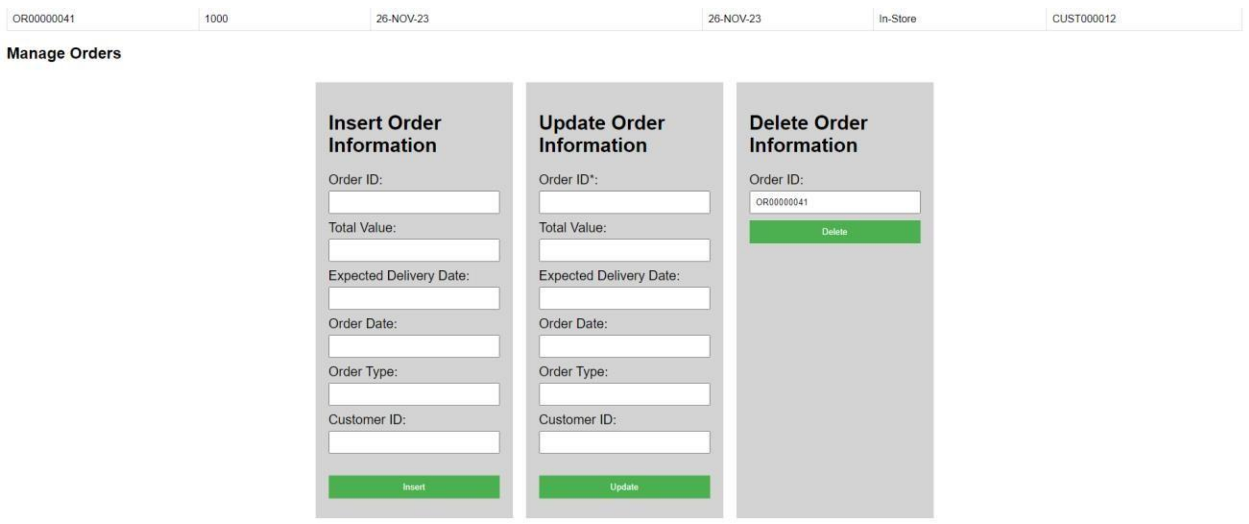
EMP00000020	arb@example.com	Jessica	White	DEPT020	Regular

Insert Employee Information

Update Employee Information

Employee ID*:	EMP00000020
Email:	
First Name:	
Last Name:	
Department ID:	
Employee Type:	Outsourced

Figure 10:Updating an employee's information



OR00000041	1000	26-NOV-23	26-NOV-23	In-Store	CUST000012

Manage Orders

Insert Order Information

Order ID:

Total Value:

Expected Delivery Date:

Order Date:

Order Type:

Customer ID:

Update Order Information

Order ID*:

Total Value:

Expected Delivery Date:

Order Date:

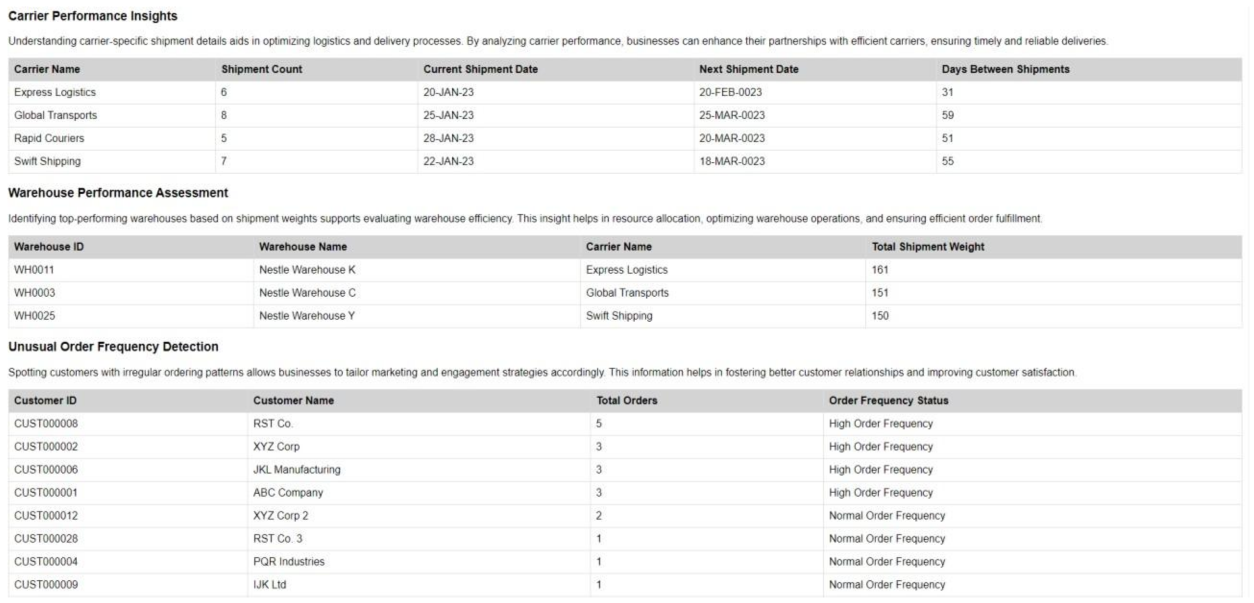
Order Type:

Customer ID:

Delete Order Information

Order ID: OR00000041

Figure 11: Deleting an order



Carrier Performance Insights

Understanding carrier-specific shipment details aids in optimizing logistics and delivery processes. By analyzing carrier performance, businesses can enhance their partnerships with efficient carriers, ensuring timely and reliable deliveries.

Carrier Name	Shipment Count	Current Shipment Date	Next Shipment Date	Days Between Shipments
Express Logistics	6	20-JAN-23	20-FEB-0023	31
Global Transports	8	25-JAN-23	25-MAR-0023	59
Rapid Couriers	5	28-JAN-23	20-MAR-0023	51
Swift Shipping	7	22-JAN-23	18-MAR-0023	55

Warehouse Performance Assessment

Identifying top-performing warehouses based on shipment weights supports evaluating warehouse efficiency. This insight helps in resource allocation, optimizing warehouse operations, and ensuring efficient order fulfillment.

Warehouse ID	Warehouse Name	Carrier Name	Total Shipment Weight
WH0011	Nestle Warehouse K	Express Logistics	161
WH0003	Nestle Warehouse C	Global Transports	151
WH0025	Nestle Warehouse Y	Swift Shipping	150

Unusual Order Frequency Detection

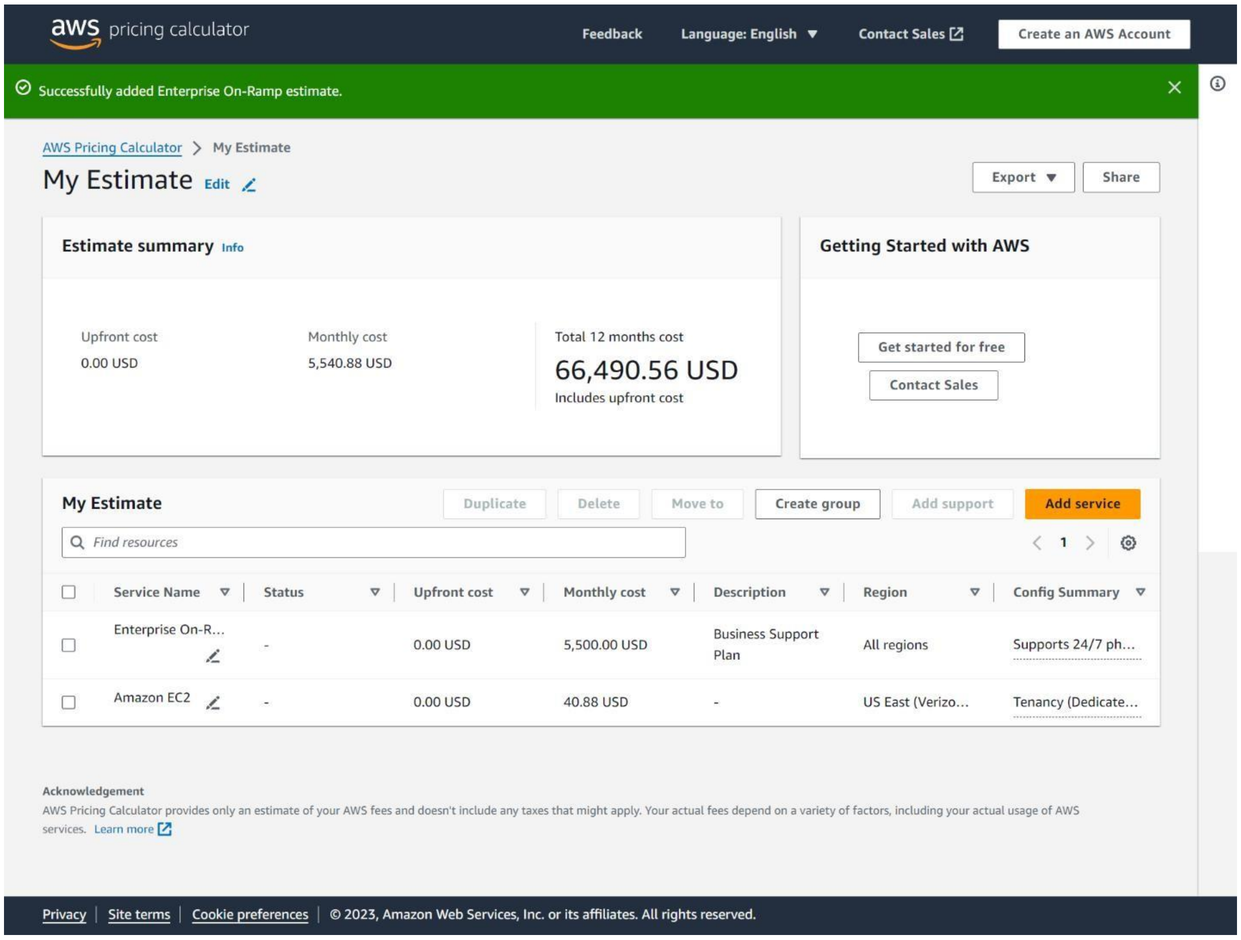
Spotting customers with irregular ordering patterns allows businesses to tailor marketing and engagement strategies accordingly. This information helps in fostering better customer relationships and improving customer satisfaction.

Customer ID	Customer Name	Total Orders	Order Frequency Status
CUST000008	RST Co.	5	High Order Frequency
CUST000002	XYZ Corp	3	High Order Frequency
CUST000006	JKL Manufacturing	3	High Order Frequency
CUST000001	ABC Company	3	High Order Frequency
CUST000012	XYZ Corp 2	2	Normal Order Frequency
CUST000028	RST Co. 3	1	Normal Order Frequency
CUST000004	PQR Industries	1	Normal Order Frequency
CUST000009	IJK Ltd	1	Normal Order Frequency

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 endeavor 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.



Successfully added Enterprise On-Ramp estimate.

AWS Pricing Calculator > My Estimate

My Estimate

Estimate summary Info

Upfront cost	Monthly cost	Total 12 months cost
0.00 USD	5,540.88 USD	66,490.56 USD Includes upfront cost

Getting Started with AWS

My Estimate

Service Name	Status	Upfront cost	Monthly cost	Description	Region	Config Summary
Enterprise On-R...	-	0.00 USD	5,500.00 USD	Business Support Plan	All regions	Supports 24/7 ph...
Amazon EC2	-	0.00 USD	40.88 USD	-	US East (Verizo...	Tenancy (Dedicate...

Acknowledgement
AWS Pricing Calculator provides only an estimate of your AWS fees and doesn't include any taxes that might apply. Your actual fees depend on a variety of factors, including your actual usage of AWS services. Learn more

Privacy | Site terms | Cookie preferences | © 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

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

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.