CREATE TABLE Inventory (

InventoryID INT PRIMARY KEY,

Quantity INT,

SupplierName VARCHAR(255),

DateOfAddition DATE,

ExpireDate DATE,

WarehouseID INT,

PricePerPiece DECIMAL(10, 2),

FOREIGN KEY (WarehouseID) REFERENCES Warehouse(WarehouseID)

-- Add other necessary columns

);

CREATE TABLE ProductCategory (

CategoryID INT PRIMARY KEY,

Prod\_Category VARCHAR(255),

TaxPercentage DECIMAL(5, 2),

ReturnPolicyInfo TEXT,

WarrantyInfo TEXT,

-- Add other necessary columns

);

CREATE TABLE Size (

SizeId INT PRIMARY KEY,

ProductId INT,

SizeValue VARCHAR(50),

FOREIGN KEY (ProductId) REFERENCES ProductInformation(Prod\_Id)

-- Add other necessary columns

);

CREATE TABLE ProductSize (

Prod\_Id INT,

SizeId INT,

PRIMARY KEY (Prod\_Id, SizeId),

FOREIGN KEY (Prod\_Id) REFERENCES ProductInformation(Prod\_Id),

FOREIGN KEY (SizeId) REFERENCES Size(SizeId)

);

CREATE TABLE Color (

ColorId INT PRIMARY KEY,

ProductId INT,

ColorValue VARCHAR(50),

FOREIGN KEY (ProductId) REFERENCES ProductInformation(Prod\_Id)

-- Add other necessary columns

);

CREATE TABLE ProductColor (

Prod\_Id INT,

ColorId INT,

PRIMARY KEY (Prod\_Id, ColorId),

FOREIGN KEY (Prod\_Id) REFERENCES ProductInformation(Prod\_Id),

FOREIGN KEY (ColorId) REFERENCES Color(ColorId)

);

CREATE TABLE ShoppingCart (

cart\_id INT PRIMARY KEY,

session\_id INT,

quantity INT,

per\_product\_total\_price DECIMAL(10, 2),

product\_id INT,

user\_id INT,

FOREIGN KEY (session\_id) REFERENCES UserSession(session\_id),

FOREIGN KEY (product\_id) REFERENCES ProductInformation(Prod\_Id), -- Assuming ProductInformation is the product table

FOREIGN KEY (user\_id) REFERENCES UserInfo(UserID) -- Assuming UserInfo is the user information table

);

CREATE TABLE ProductInformation (

Prod\_Id INT PRIMARY KEY,

CategoryID INT,

InventoryID INT,

Prod\_Name VARCHAR(255),

Prod\_Description TEXT,

Prod\_manufacturer VARCHAR(255),

Prod\_Price DECIMAL(10, 2),

image VARCHAR(255),

isActive BOOLEAN,

NumberOfPoints INT,

FOREIGN KEY (InventoryID) REFERENCES Inventory(InventoryID),

FOREIGN KEY (CategoryID) REFERENCES ProductCategory(CategoryID))

-- Add other necessary columns

);

CREATE TABLE OrderInfo (

OrderID INT PRIMARY KEY,

UserID INT,

CartID INT,

FinalPrice DECIMAL(10, 2),

PaymentID INT,

OrderDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

OrderStatus BOOLEAN,

DiscountApplied DECIMAL(5, 2),

CityDeliveryFeeID INT,

FOREIGN KEY (UserID) REFERENCES UserProfileInfo(UserID),

FOREIGN KEY (CartID) REFERENCES ShoppingCart(CartID),

FOREIGN KEY (PaymentID) REFERENCES PaymentCentral(PaymentID),

FOREIGN KEY (CityDeliveryFeeID) REFERENCES CityDeliveryFee(CityDeliveryFeeID)

-- Add other necessary columns

);

-- Payment central Table

CREATE TABLE Payment (

PaymentID INT PRIMARY KEY,

UserID INT,

Amount DECIMAL(10, 2),

PaymentDate DATE,

PaymentStatus VARCHAR(50),

PaymentMethod VARCHAR(50),

FOREIGN KEY (UserID) REFERENCES Users(UserID)

)

CREATE TABLE CustomerOffersOrders (

CustomerOfferID INT PRIMARY KEY,

OrderID INT, -- Foreign Key

OfferID INT -- Foreign Key,

PurchaseDate DATE,

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (OfferID) REFERENCES Offers(OfferID)

);

CREATE TABLE ProductOffers (

OfferID INT PRIMARY KEY,

OfferTypeID INT, -- Foreign Key referencing OfferTypes Table

ProductID INT, -- Foreign Key referencing Product Table

SpecialOfferPrice DECIMAL,

OfferDescription VARCHAR(255),

FOREIGN KEY (OfferTypeID) REFERENCES OfferTypes(OfferTypeID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

CREATE TABLE CollectionCenter (

CenterID INT PRIMARY KEY,

OrderID INT, -- Foreign Key

OrderStatus VARCHAR(255),

ReturnID INT -- Foreign Key

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ReturnID) REFERENCES Returns(ReturnID)

);

CREATE TABLE Delivery (

RiderID INT PRIMARY KEY,

OrderID INT, -- Foreign Key

CustomerName VARCHAR(255),

ShippingAddress VARCHAR(255),

ContactInformation VARCHAR(255),

OrderStatus VARCHAR(255),

ReturnID INT, -- Foreign Key

DeliveryType varchar(70) default order,

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

FOREIGN KEY (ReturnID) REFERENCES Returns(ReturnID)

);

CREATE TABLE ReturnExchange (

ReturnExchangeID INT PRIMARY KEY,

ProductID INT, -- Foreign Key

OrderID INT, -- Foreign Key

ExchangeReturnDate DATE,

Type VARCHAR(255), -- Return or Exchange

Status VARCHAR(255), -- Pending, Completed

Description VARCHAR(255),

FOREIGN KEY (ProductID) REFERENCES ProductInformation(Prod\_ID),

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)

);

CREATE TABLE Interaction (

InteractionID INT PRIMARY KEY,

InteractionType VARCHAR(255),

InteractionTimestamp TIMESTAMP,

InteractionDetails VARCHAR(255)

);

CREATE TABLE LuckyDraw (

draw\_id INT PRIMARY KEY,

draw\_name VARCHAR(255),

draw\_start\_date DATE,

draw\_end\_date DATE,

prize\_description VARCHAR(255),

participants\_limit INT,

registration\_status BOOLEAN DEFAULT FALSE

);

CREATE TABLE Participant (

participant\_id INT PRIMARY KEY,

draw\_id INT,

order\_id INT, -- Foreign Key

participant\_name VARCHAR(255),

participant\_email VARCHAR(255),

registration\_date TIMESTAMP,

winner BOOLEAN DEFAULT FALSE,

FOREIGN KEY (draw\_id) REFERENCES LuckyDraw(draw\_id),

FOREIGN KEY (order\_id) REFERENCES OrderInfo(order\_id)

);

CREATE TABLE Bundle (

BundleID INT PRIMARY KEY,

OfferID INT,

ProductID INT,

DiscountPercentage DECIMAL(5,2),

BundlePrice DECIMAL(10,2),

FOREIGN KEY (OfferID) REFERENCES ProductOffers(OfferID),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

-- WebsiteTraffic Table

CREATE TABLE WebsiteTraffic (

VisitID INT PRIMARY KEY,

Timestamp TIMESTAMP,

PageURL VARCHAR(255),

UserID INT FOREIGN KEY REFERENCES User(UserID),

-- Other WebsiteTraffic columns...

);

CREATE TABLE HamperRecipient (

recipient\_id INT PRIMARY KEY,

order\_id INT, -- Foreign Key

hamper\_id INT, -- Foreign Key

recipient\_name VARCHAR(255),

recipient\_address VARCHAR(255),

purchase\_date DATE,

FOREIGN KEY (order\_id) REFERENCES OrderInfo(order\_id),

FOREIGN KEY (hamper\_id) REFERENCES GiftHampers(hamper\_id)

);

CREATE TABLE GiftHampers (

HamperID INT PRIMARY KEY,

HamperName VARCHAR(255),

HamperDescription VARCHAR(255),

HamperLimit DECIMAL(10,2)

);

CREATE TABLE SeasonalTrends (

SeasonalTrendID INT PRIMARY KEY,

Timestamp TIMESTAMP,

TrendDescription VARCHAR(255)

);

CREATE TABLE SalesTrends (

SalesTrendID INT PRIMARY KEY,

ProductID INT,

Timestamp TIMESTAMP,

TrendDescription VARCHAR(255),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

CREATE TABLE UserProfiles (

UserID INT PRIMARY KEY,

Full\_Name VARCHAR(255),

User\_Name VARCHAR(255),

User\_Password VARCHAR(255),

Mobile\_Number VARCHAR(20),

Email VARCHAR(255),

Date\_of\_Birth DATE,

Verification\_Code VARCHAR(255),

Address VARCHAR(255),

Gender VARCHAR(10),

NumberOfPoints INT,

profileCompletionPercent INT,

Created\_Account\_Date TIMESTAMP,

Update\_Account\_Date TIMESTAMP

);

CREATE TABLE Referrals (

ReferralID INT PRIMARY KEY,

ReferrerUserID INT,

ReferralType VARCHAR(255),

FOREIGN KEY (ReferrerUserID) REFERENCES UserProfiles(UserID)

);

CREATE TABLE SeasonalProducts (

SeasonalProductID INT PRIMARY KEY,

ProductID INT,

SeasonStart DATE,

SeasonEnd DATE,

Description VARCHAR(255),

IsActive BOOLEAN,

CreatedAt TIMESTAMP,

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

CREATE TABLE FAQs (

FAQID INT PRIMARY KEY,

ProductID INT,

Question VARCHAR(255),

FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

);

CREATE TABLE FAQAnswers (

AnswerID INT PRIMARY KEY,

FAQID INT,

AnswerText TEXT,

FOREIGN KEY (FAQID) REFERENCES FAQs(FAQID)

);

CREATE TABLE GuestUsers (

GuestUserID INT PRIMARY KEY,

SessionID INT,

Timestamps TIMESTAMP,

FOREIGN KEY (SessionID) REFERENCES UserSessions(session\_id)

);

CREATE TABLE UserSessions (

session\_id INT PRIMARY KEY,

user\_id INT,

session\_token VARCHAR(255),

last\_activity TIMESTAMP,

FOREIGN KEY (user\_id) REFERENCES UserProfiles(UserID)

);

-- CityDeliveryFee Table

CREATE TABLE CityDeliveryFee (

CityDeliveryFeeID INT PRIMARY KEY,

Fee DECIMAL(10, 2) NOT NULL,

CityName VARCHAR(255) UNIQUE

);

—--------------------#98

42. Tables 3, JUNCTION TABLE , Multiple Inventories can be associated with many warehouse (many to many) (Inventory Total - warehouse Total)

-- Create Warehouse table

CREATE TABLE Warehouse (

WarehouseID INT PRIMARY KEY,

OrderID INT, -- Foreign Key referencing some other table

OrderStatus VARCHAR(50),

Location VARCHAR(255),

ReturnID INT, -- Foreign Key referencing some other table

);

-- Additional tables and relationships should be defined as needed.

43. Tables 2, FOREIGN KEY any side , Each admin can be associated with a unique user (1 to 1) (admin Total - User Partial)

-- Create User Profile Info table

-- Create Administration table

CREATE TABLE Administration (

AdminID INT PRIMARY KEY AUTO\_INCREMENT,

UserID INT UNIQUE, -- Foreign Key referencing UserProfileInfo

FOREIGN KEY (UserID) REFERENCES UserProfileInfo(UserID),

Role ENUM('SuperAdmin', 'Moderator', 'Support') NOT NULL,

IsActive BOOLEAN

);

44. Tables 3, Junction table , Many expiring product is associated with multiple inventories (many to many) (Expiring Total - Inventory Partial)

-- Create Expiring Discounts table

CREATE TABLE ExpiringDiscounts (

ExpireID INT PRIMARY KEY AUTO\_INCREMENT,

InventoryID INT, -- Foreign Key

ProductOfferID INT, -- Foreign Key

OrderID INT, -- Foreign Key

SellingDate DATE,

Discounts DECIMAL(10, 2),

DemandIncrease INT,

NewSellingDate DATE,

FOREIGN KEY (InventoryID) REFERENCES ProductInventory(InventoryID),

FOREIGN KEY (ProductOfferID) REFERENCES ProductInventory(ProductOfferID),

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID) -- Adjust based on your actual Orders table

);

45. Tables 2, FOREIGN KEY any side , One Expiring product can be associated with One product offer (1 to 1) (Expiring Product Total - Product Offer Partial)

46. Tables 3, JUNCTION TABLE , Multiple Expiring Products can be associated with multiple Orders (many to many) (Expiring Product Partial Order Total)

47. Tables 3, JUNCTION TABLE , Many banned product is associated with multiple inventories (many to many) (Banned Total - Inventory Partial)

-- Create Banned Products table

CREATE TABLE BannedProducts (

BannedProductID INT PRIMARY KEY AUTO\_INCREMENT,

InventoryID INT, -- Foreign Key

OfferID INT, -- Foreign Key

OrderID INT, -- Foreign Key

Demand INT,

Discounted BOOLEAN,

FOREIGN KEY (InventoryID) REFERENCES ProductInventory(InventoryID), -- Assuming ProductInventory table

FOREIGN KEY (OfferID) REFERENCES ProductOffers(OfferID), -- Assuming ProductOffers table

FOREIGN KEY (OrderID) REFERENCES OrderInfo(OrderID) -- Assuming OrderInfo table

);

48. Tables 2, FOREIGN KEY any side , One Banned Product can be associated with One product offer (1 to 1) ( Banned Product Total - Product Offer Partial)

49. Tables 3, JUNCTION TABLE , Multiple Banned Products can be associated with multiple Orders (many to many) (Banned Product Partial Order Total)

50. Tables 3, JUNCTION TABLE , Many unbanned product is associated with multiple inventories (many to many) (Unbanned Total - Inventory Partial)

-- Create Unbanned Products table

CREATE TABLE UnbannedProducts (

UnbannedProductID INT PRIMARY KEY AUTO\_INCREMENT,

InventoryID INT, -- Foreign Key

OfferID INT, -- Foreign Key

OrderID INT, -- Foreign Key

Demand INT,

HikedPrice DECIMAL(10, 2),

FOREIGN KEY (InventoryID) REFERENCES ProductInventory(InventoryID), -- Assuming ProductInventory table

FOREIGN KEY (OfferID) REFERENCES ProductOffers(OfferID), -- Assuming ProductOffers table

FOREIGN KEY (OrderID) REFERENCES OrderInfo(OrderID) -- Assuming OrderInfo table

);

51. Tables 2, FOREIGN KEY any side , One Unbanned Product can be associated with One product offer (1 to 1) ( Unbanned Product Total - Product Offer Partial)

52. Tables 3, JUNCTION TABLE , Multiple Unbanned Products can be associated with multiple Orders (many to many) (Unbanned Product Partial Order Total)

53. Tables 3, JUNCTION TABLE , Many Seasonal hike product is associated with multiple inventories (many to many) (Seasonal Hike Total - Inventory Partial)

-- Create Seasonal Hikes table

CREATE TABLE SeasonalHikes (

SeasonID INT PRIMARY KEY AUTO\_INCREMENT,

InventoryID INT, -- Foreign Key

OfferID INT, -- Foreign Key

OrderID INT, -- Foreign Key

Demand INT,

HikedPrice DECIMAL(10, 2),

FOREIGN KEY (InventoryID) REFERENCES ProductInventory(InventoryID), -- Assuming ProductInventory table

FOREIGN KEY (OfferID) REFERENCES ProductOffers(OfferID), -- Assuming ProductOffers table

FOREIGN KEY (OrderID) REFERENCES OrderInfo(OrderID) -- Assuming OrderInfo table

);

54. Tables 2, FOREIGN KEY any side , One Seasonal Hike Product can be associated with One product offer (1 to 1) (Seasonal Hike Product Total - Product Offer Partial)

55. Tables 3, JUNCTION TABLE , Multiple Seasonal Hike Products can be associated with multiple Orders (many to many) (Seasonal Hike Product Partial Order Total)

56. Tables 3, JUNCTION TABLE , Many Sale Product is associated with multiple inventories (many to many) (Sale product Total - Inventory Partial)

-- Create Sale Optimization table

CREATE TABLE SaleOptimization (

SaleID INT PRIMARY KEY AUTO\_INCREMENT,

OfferID INT, -- Foreign Key

InventoryID INT, -- Foreign Key

SuccessRate DECIMAL(5, 2),

FOREIGN KEY (OfferID) REFERENCES ProductOffers(OfferID), -- Assuming ProductOffers table

FOREIGN KEY (InventoryID) REFERENCES ProductInventory(InventoryID) -- Assuming ProductInventory table

);

57. Tables 2, Foreign key any side , One Sale Product can be associated with One product offer (1 to 1) (Sale Hike Product Total - Product Offer Total)

);

58. Tables 3, JUNCTION TABLE , Many Charity Purchase will have multiple partners (many to many) (Charity Purchase Total - partner Partial )

-- Create Charitable Purchase Donations table

CREATE TABLE CharitablePurchaseDonations (

DonatedID INT PRIMARY KEY AUTO\_INCREMENT,

ProductID INT, -- Foreign Key

PartnerID INT, -- Foreign Key

CartID INT, -- Foreign Key

FOREIGN KEY (ProductID) REFERENCES Products(ProductID), -- Assuming Products table

FOREIGN KEY (PartnerID) REFERENCES CharityPartners(PartnerID),

FOREIGN KEY (CartID) REFERENCES ShoppingCart(CartID) -- Assuming ShoppingCart table

);

-- Create Charity Partners table

CREATE TABLE CharityPartners (

PartnerID INT PRIMARY KEY AUTO\_INCREMENT,

PartnerName VARCHAR(255),

OrganizationLocation VARCHAR(255),

LinkDate DATE,

TotalAmountPaid DECIMAL(10, 2),

TotalAmountDue DECIMAL(10, 2)

-- Add other necessary fields

);

59. Tables 3, JUNCTION TABLE , Many product will have multiple charities (many to many) (Product Partial - Charity Total)

60. Tables 2, FOREIGN KEY cart , Each charity will have multiple carts ( 1 to many) (charity total - Cart Partial)

61. Tables 3, JUNCTION TABLE , Many Charity on purchase will have multiple partners (many to many) (Charity on Purchase Total - Partner Partial)

-- Create Charity On Purchase table

CREATE TABLE CharityOnPurchase (

CharPerProID INT PRIMARY KEY AUTO\_INCREMENT,

ProductID INT, -- Foreign Key

PartnerID INT, -- Foreign Key

CartID INT, -- Foreign Key

ProfitPerPiece DECIMAL(10, 2),

DonationPerPiece DECIMAL(10, 2),

DemandIncreased INT,

FOREIGN KEY (ProductID) REFERENCES Products(ProductID), -- Assuming Products table

FOREIGN KEY (PartnerID) REFERENCES CharityPartners(PartnerID),

FOREIGN KEY (CartID) REFERENCES ShoppingCart(CartID) -- Assuming ShoppingCart table

);

62. Tables 3, JUNCTION TABLE , Many Product will have multiple charities on purchase (many to many) (Charity on Purchase Total - Product Partial)

63. Tables 2, FOREIGN KEY carts , Each charity purchase will have multiple carts ( 1 to many) (Charity on purchase Total - cart Partial)

64. Tables 3, JUNCTION TABLE , Many customer interest will have multiple orders ( many to many) Customer interest Total - Order Partial)

-- Create Customer Interests table

CREATE TABLE CustomerInterests (

CustIntID INT PRIMARY KEY AUTO\_INCREMENT,

OrderID INT, -- Foreign Key

ProductCategory VARCHAR(255),

ProductPrice DECIMAL(10, 2),

FOREIGN KEY (OrderID) REFERENCES OrderInfo(OrderID) -- Assuming OrderInfo table

);

—------------------------#142

**Point 12**

**Point 13**

CREATE TABLE PaypalPayment (

PayPalPaymentID INT PRIMARY KEY,

PaymentID INT REFERENCES Payment(PaymentID),

PayPalEmail VARCHAR(255),

PayPalAmount DECIMAL(10, 2),

PayPalDate DATE,

PayPalStatus VARCHAR(50)

);

**Point 14**

CREATE TABLE GatewayPayment (

GatewayPaymentID INT PRIMARY KEY,

PaymentID INT REFERENCES Payment(PaymentID),

PaymentGateway VARCHAR(50),

GatewayAmount DECIMAL(10, 2),

GatewayAccountNumber varchar(50),

GatewayDate DATE,

GatewayStatus VARCHAR(50)

);

**Point 16**

CREATE TABLE CardPayment (

CardPaymentID INT PRIMARY KEY,

PaymentID INT REFERENCES Payment(PaymentID),

CardType VARCHAR(20),

CardNumber VARCHAR(16),

ExpiryDate DATE,

cnicNumber VARCHAR(20),

CVV VARCHAR(4),

CardHolderName VARCHAR(100),

BillingAddress VARCHAR(255),

CardPaymentAmount DECIMAL(10, 2),

CardPaymentDate DATE,

CardPaymentStatus VARCHAR(50)

);

**Point 17**

CREATE TABLE BankTransferPayment (

BankTransferID INT PRIMARY KEY,

PaymentID INT REFERENCES Payment(PaymentID),

BankName VARCHAR(50),

AccountHolderName VARCHAR(100),

AccountNumber VARCHAR(20),

IBAN VARCHAR(34),

TransferAmount DECIMAL(10, 2),

TransferDate DATE,

TransferStatus Bit

);

**Point 18**

CREATE TABLE CODPayment (

CODPaymentID INT PRIMARY KEY,

PaymentID INT REFERENCES Payment(PaymentID),

COD\_Received BOOLEAN,

COD\_PaidAmount DECIMAL(10, 2),

COD\_PaymentStatus VARCHAR(50),

COD\_PaidDate DATE,

COD\_ReceiverName VARCHAR(100)

);

**Point 19**

CREATE TABLE ProductIssues (

IssueID INT PRIMARY KEY,

ProductID INT REFERENCES Product(Product\_Id),

OrderID INT REFERENCES Orders(OrderID),

IssueType VARCHAR(50),

Description TEXT,

Status VARCHAR(50)

);

**Point 20**

**Point 21**

CREATE TABLE UserSearchHistory (

SearchID INT PRIMARY KEY,

UserID INT REFERENCES UserProfile(UserID),

SearchText VARCHAR(255) NOT NULL

);

**Point 22**

**Point 23**

**Point** 24

**Point 25**

CREATE TABLE CustomerSupportRequests (

RequestID INT PRIMARY KEY,

UserID INT REFERENCES UserProfile(UserID),

RequestType VARCHAR(50),

RequestDate DATE,

Status VARCHAR(50),

Description TEXT

);

CREATE TABLE SupportResponses (

ResponseID INT PRIMARY KEY,

RequestID INT REFERENCES CustomerSupportRequests(RequestID),

ResponderName VARCHAR(255),

ResponseType VARCHAR(50),

ResponseDate DATE,

ResponseText TEXT

);

**Point 27**

CREATE TABLE Wishlists (

WishlistID INT PRIMARY KEY,

UserID INT REFERENCES UserProfile(UserID),

ProductID INT REFERENCES Product(Product\_Id),

DateAdded DATE

);

**Point 28**

**Point 29**

CREATE TABLE ReviewsAndRating (

ReviewID INT PRIMARY KEY,

UserID INT REFERENCES UserProfile(UserID),

ProductID INT REFERENCES Product(Prod\_Id),

Rating INT, -- Assuming a rating is an integer (e.g., 1 to 5)

ReviewText TEXT,

ReviewDate DATE

);

**Point 30**

**Point 31**

CREATE TABLE Feedback (

FeedbackID INT PRIMARY KEY,

UserID INT REFERENCES UserProfile(UserID),

FeedbackText TEXT,

FeedbackDate DATE,

Rating INT

);

**Point 32**

CREATE TABLE OfferTypes (

OfferTypeID INT PRIMARY KEY,

OfferTypeName VARCHAR(50),

OfferTypeDescription TEXT,

StartDate DATE,

EndDate DATE,

DefaultDiscountPercentage DECIMAL(5, 2),

OtherOfferTypeDetails TEXT

);

**Point 33**