Generic Company Information

Name: Generic Flatpack Furniture Co. (GFFC)
Catchphrase: We sell you furniture, you give us money.

Company Specification:

Our company, Generic Flatpack Furniture Co. (GFFC) is a Swedish company which manufactures, distributes and sells affordable furniture. We are a vertically integrated company that handles our whole supply chain.

We have many outside suppliers that provide raw materials for our products, along with providing some finished goods we may sell. We have an online presence where customers may browse our catalogue of items, including the availability of items in selected stores.

In the store, our associates will be able to help customers find the products they want. Customers can go to a service counter and directly order items from that store's inventory or directly from the central warehouse. Items from the central warehouse can be picked up from our store or they can be delivered by third-party couriers.

Our supplies are catalogued along with the components they supply and the stock level. We like keeping catalogues of the available materials and products GFFC's suppliers can provide. If a new product is created, then we would like to ensure we can create enough stock using the raw materials we have.

The type of furniture we provide falls under a range of categories. Namely, those associated with the traditional rooms of the home, e.g. lounges, dining rooms, kitchens, bathrooms and bedrooms. We have recently expanded our product line to include an outdoor furniture section that incorporates jacuzzis.

Our online staff comes to a total of 20, including 4 Virtual Assistants. Each in-person branch has about 30 employees. We operate across 4 different regions, with our biggest two branches located in Slough and Swindon. There are 4 regional managers and 4 assistants to each of those regional managers. Our employees are trained to be flexible and fill in for each other, so each employee is trained to work in multiple roles such as customer assistants, warehouse workers, managers, etc.

GFFC has recently taken on a new CEO who has some very exacting standards. We want this system to take account of our vertically integrated model. Staff in branches should be able to book shifts and we should be able to track when a shift needs more staff. We also want the system to handle orders and products from both the central warehouse and an individual branch. Keeping track of how many build materials we have is needed and how many products we can produce.

Design Decisions

Tools we used

We decided to use draw.io for our logical and conceptual designs since it is free and easy to use and meets our requirements. Even though draw.io allows multiple users to interact on a project at the same time, we had some significant problems with this, as people would make edits that would contradict with each other, which caused data loss. We ended up having one person do all the diagrams and others would give them instructions or advice: multiple people did this and we'd switch roles regularly.

Once we normalised our design, we used SQL workbench to do our final design, since that can easily be converted into a database template and exported to raw SQL statements. Lastly, we used Figma to make our prototypes for the user interface. Again, Figma allows multiple users to edit a project at once, offering the team greater interaction and input on the final product. It allowed us to combine creative talents to produce a satisfying and user-friendly result.

General Design

We started by compartmentalising the system into three distinct packages: the Central Warehouse, the Branch and the Supply Chain. The Central Warehouse was the simplest as it only needed to process orders and store products to be ordered. The Supply Chain needs to store suppliers and their catalogues so we can buy build materials from them. It also needs to store a list of products we make and the build materials they use. The Branch is the most complex; it needs to list all our branches and keep track of their inventory, purchases and deliveries. We also need branches to keep track of their staff and the shifts.

Tables

Excluding joining tables, we have a total of 11 tables:

- Suppliers - stores information on suppliers (name, contact info, location)
- Staff - keeps track of company staff information
- MaterialsCatalogue - a catalogue of materials offered by our suppliers
- Shifts - for managing shifts at a particular branch
- Storage - keeps track of where our materials are stored
- Branches - information on the different branches of the company
- Products - a full table of all the products that the company sells
- Inventory - available products from the warehouse
- Orders - orders that branches have placed
- Warehouses - information about all of the company's warehouses
- Purchases - purchases customers have made

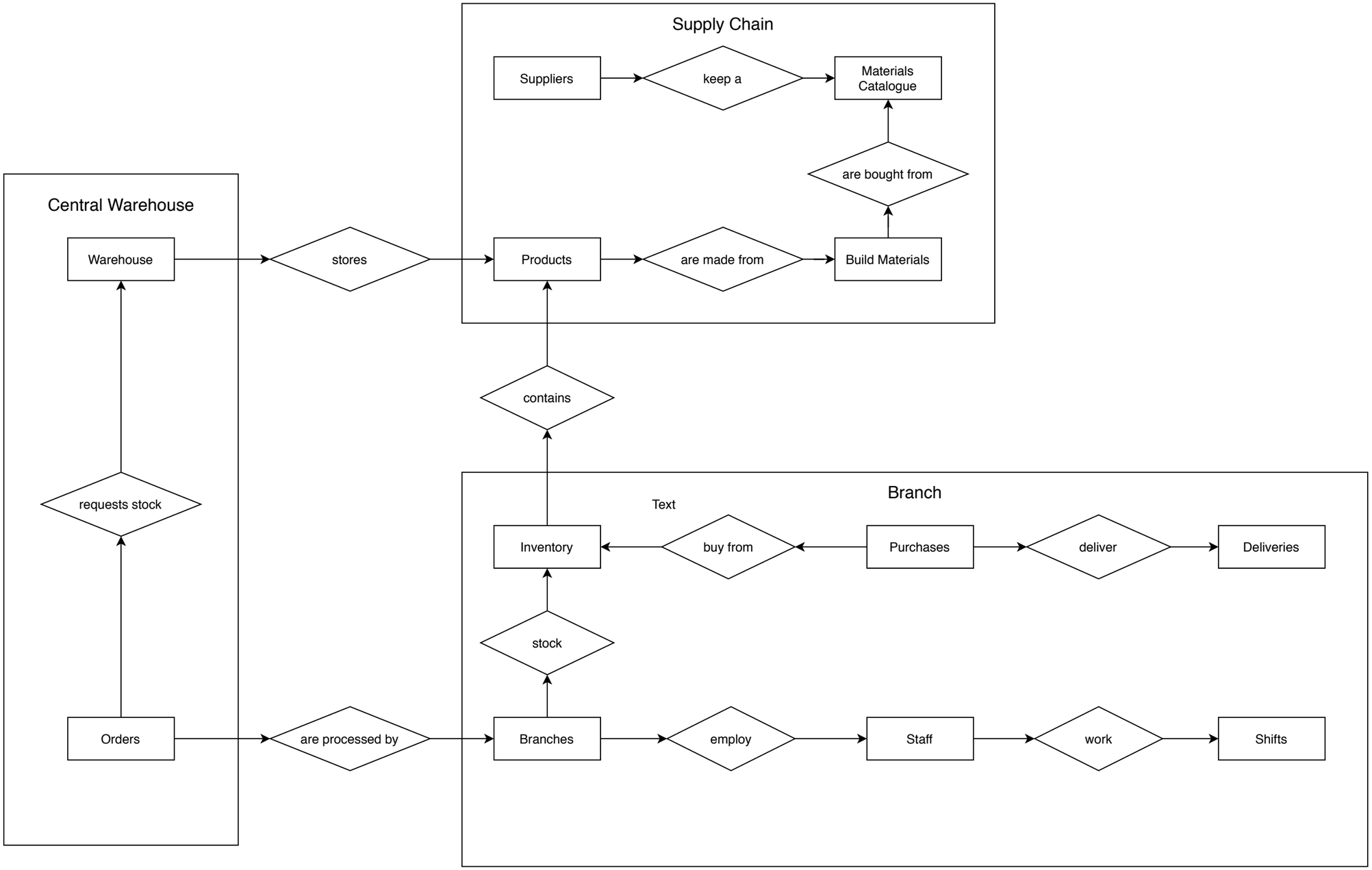
Since the company is mostly vertically integrated, there is a lot more information to keep track of since we manage a large proportion of the supply chain.

Example Product Lifecycle

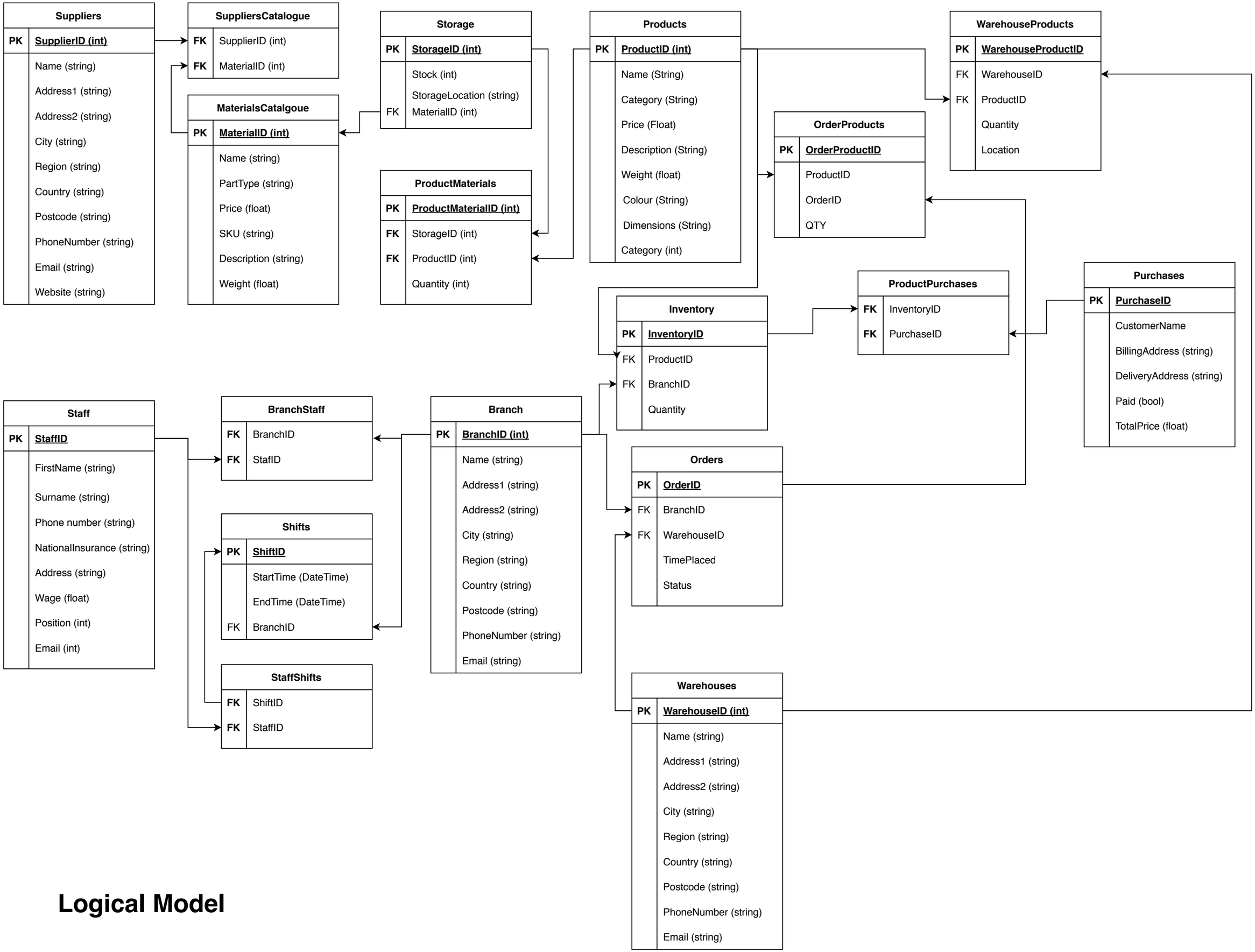
We will use a plank of medium-density fibreboard (MDF) as a demonstration of how our product supply chain works. The MDF plank begins as an item stored in MaterialsCatalogue, which can also be found in storage. The plank is then chosen to be turned into a shelf, which is listed in Products and stored in the warehouse. The Slough branch places an order (tracked in Orders) for 50 of these shelves to be kept in stock. The shelf made from our original plank is included in this order and is stored in Inventory. A customer then makes a purchase at the Slough branch for three of these shelves - the purchase is logged in Purchases, the quantity of shelves in that branches' inventory decreases by 3, and the customer walks away with three shelves.

Image Acknowledgements:

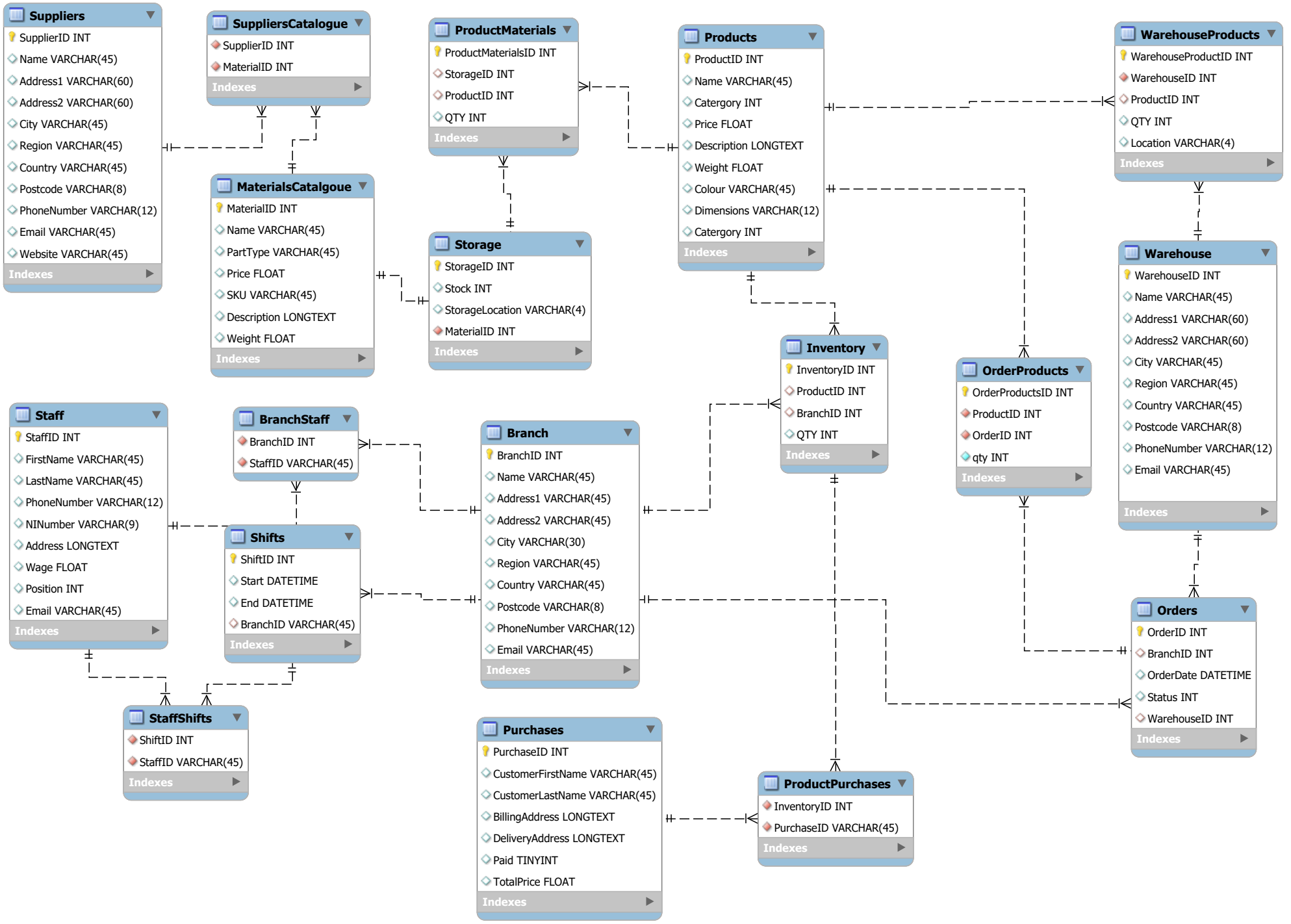
- Product Page:
    - Rugs: Isabelle Taylor on Pexels - https://www.pexels.com/photo/close-up-of-gray-cable-knit-cloth-1421253/
    - Shelves: https://www.ikea.com/gb/en/



Conceptual Model



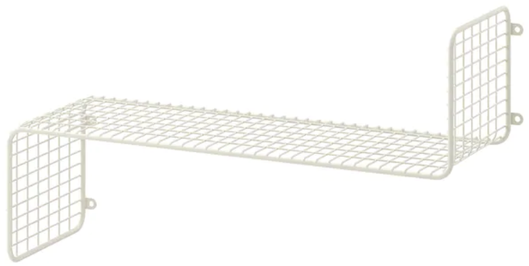
Logical Model


Suppliers
SupplierID INT
Name VARCHAR(45)
Address1 VARCHAR(60)
Address2 VARCHAR(60)
City VARCHAR(45)
Region VARCHAR(45)
Country VARCHAR(45)
Postcode VARCHAR(8)
PhoneNumber VARCHAR(12)
Email VARCHAR(45)
Website VARCHAR(45)
Indexes
SuppliersCatalogue
SupplierID INT
MaterialID INT
Indexes
ProductMaterials
ProductMaterialsID INT
StorageID INT
ProductID INT
QTY INT
Indexes
Products
ProductID INT
Name VARCHAR(45)
Catergory INT
Price FLOAT
Description LONGTEXT
Weight FLOAT
Colour VARCHAR(45)
Dimensions VARCHAR(12)
Catergory INT
Indexes
WarehouseProducts
WarehouseProductID INT
WarehouseID INT
ProductID INT
QTY INT
Location VARCHAR(4)
Indexes
MaterialsCatalgoue
MaterialID INT
Name VARCHAR(45)
PartType VARCHAR(45)
Price FLOAT
SKU VARCHAR(45)
Description LONGTEXT
Weight FLOAT
Indexes
Storage
StorageID INT
Stock INT
StorageLocation VARCHAR(4)
MaterialID INT
Indexes
Warehouse
WarehouseID INT
Name VARCHAR(45)
Address1 VARCHAR(60)
Address2 VARCHAR(60)
City VARCHAR(45)
Region VARCHAR(45)
Country VARCHAR(45)
Postcode VARCHAR(8)
PhoneNumber VARCHAR(12)
Email VARCHAR(45)
Indexes
Inventory
InventoryID INT
ProductID INT
BranchID INT
QTY INT
Indexes
OrderProducts
OrderProductsID INT
ProductID INT
OrderID INT
qty INT
Indexes
Staff
StaffID INT
FirstName VARCHAR(45)
LastName VARCHAR(45)
PhoneNumber VARCHAR(12)
NINumber VARCHAR(9)
Address LONGTEXT
Wage FLOAT
Position INT
Email VARCHAR(45)
Indexes
BranchStaff
BranchID INT
StaffID VARCHAR(45)
Branch
BranchID INT
Name VARCHAR(45)
Address1 VARCHAR(45)
Address2 VARCHAR(45)
City VARCHAR(30)
Region VARCHAR(45)
Country VARCHAR(45)
Postcode VARCHAR(8)
PhoneNumber VARCHAR(12)
Email VARCHAR(45)
Indexes
Shifts
ShiftID INT
Start DATETIME
End DATETIME
BranchID VARCHAR(45)
Indexes
Orders
OrderID INT
BranchID INT
OrderDate DATETIME
Status INT
WarehouseID INT
Indexes
StaffShifts
ShiftID INT
StaffID VARCHAR(45)
Indexes
Purchases
PurchaseID INT
CustomerFirstName VARCHAR(45)
CustomerLastName VARCHAR(45)
BillingAddress LONGTEXT
DeliveryAddress LONGTEXT
Paid TINYINT
TotalPrice FLOAT
Indexes
ProductPurchases
InventoryID INT
PurchaseID VARCHAR(45)
Indexes

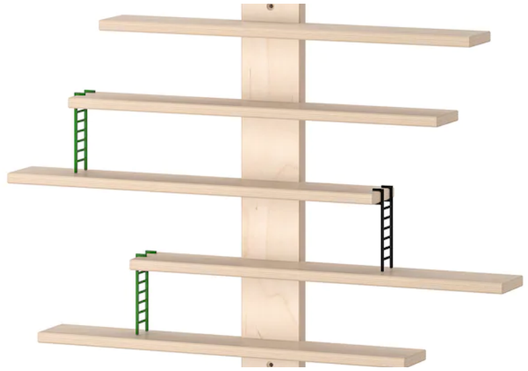
Search Catalogue

Sort by

Shelf
£10
200x30x3cm



Crappy metal shelf
£8
60x20x3



Kids' shelf
£15
37x37x3

Practical shelf
£18
101x20x3

Hipster shelf
£25
120x25x3

Hanging shelf
£12
80x20x3

Basket

×	1		Shelf 200x30X3cm	£10.00
×	1		Crappy metal shelf 60x20x3	£8.00
×	1		Kids' shelf 37x37x3	£15.00

Total £33.00

Advance to Checkout

Checkout

Name*
Enter your first and last name.

Hugh Mungous

Email address
Enter an email address that we can use to contact you.

example@example.com

Phone number
Enter your phone number so we can call you if needed.

01234 567890

Delivery address
Enter your delivery address.

Flat 1

1 Example Street

Exampleton

EX1 1EX

☐ My billing address is the same as my delivery address

☐ I agree to the Terms and Conditions*

☐ I agree to the Privacy Policy*

Credit/Debit card Checkout | G Pay | PayPal | Pay

Search Catalogue

Special Offers

Bathroom

Outdoors

Accessories

Decoration

Beds and Mattresses

Kitchen

Furniture

Textiles

Lighting

Rugs

Shift Management

Date	Time ▼	Slots left	Staff on Shift	Choose Shift	Cancel Shift
2069/4/20	04:20-10:00	5/7	7	Select Shift	Cancel
2069/4/20	10:00-16:20	4/20	20	Select Shift	Cancel
2069/4/20	16:20-22:00	6/9	9	Select Shift	Cancel
2069/4/21	04:20-10:00	5/7	7	Select Shift	Cancel
2069/4/21	10:00-16:20	4/20	20	Select Shift	Cancel
2069/4/21	16:20-22:00	6/9	9	Select Shift	Cancel
2069/4/22	04:20-10:00	5/7	7	Shift selected	Cancel
2069/4/22	10:00-16:20	4/20	20	Shift selected	Cancel
2069/4/22	16:20-22:00	6/9	9	Shift selected	Cancel

Search Inventory

Item Name ▲	Location	Price	Weight	Dimensions (WxDxH)	SKU No	Quantity	View Item Page
Crappy metal shelf	A1	£8	1.00kg	60x20x3cm	1001	69	View Item
Hanging shelf	A1	£12	1.00kg	80x20x3cm	1002	21	View Item
Hipster shelf	A1	£25	1.00kg	120x25x3cm	1003	0	View Item
Kids shelf	A1	£15	1.00kg	37x37x3cm	1004	1	View Item
Practical shelf	A1	£18	1.00kg	101x20x3cm	1005	99	View Item
Shelf	A1	£10	1.00kg	200x30x3cm	1006	20	View Item

SQL Create Commands

Suppliers

```
CREATE TABLE IF NOT EXISTS `mydb`.`Suppliers` (
  `SupplierID` INT NOT NULL,
  `Name` VARCHAR(45) NULL,
  `Address1` VARCHAR(60) NULL,
  `Address2` VARCHAR(60) NULL,
  `City` VARCHAR(45) NULL,
  `Region` VARCHAR(45) NULL,
  `Country` VARCHAR(45) NULL,
  `Postcode` VARCHAR(8) NULL,
  `PhoneNumber` VARCHAR(12) NULL,
  `Email` VARCHAR(45) NULL,
  `Website` VARCHAR(45) NULL,
  PRIMARY KEY (`SupplierID`),
  UNIQUE INDEX `SupplierID_UNIQUE` (`SupplierID` ASC) VISIBLE)
```

MaterialsCatalgoue

```
CREATE TABLE IF NOT EXISTS `mydb`.`MaterialsCatalgoue` (
  `MaterialID` INT NOT NULL,
  `Name` VARCHAR(45) NULL,
  `PartType` VARCHAR(45) NULL,
  `Price` FLOAT NULL,
  `SKU` VARCHAR(45) NULL,
  `Description` LONGTEXT NULL,
  `Weight` FLOAT NULL,
  PRIMARY KEY (`MaterialID`),
  UNIQUE INDEX `MaterialID_UNIQUE` (`MaterialID` ASC) VISIBLE)
```

SuppliersCatalogue

```
CREATE TABLE IF NOT EXISTS `mydb`.`SuppliersCatalogue` (
  `SupplierID` INT NOT NULL,
  `MaterialID` INT NOT NULL,
  INDEX `fk_SuppliersCatalogue_MaterialsCatalgoue1_idx` (`MaterialID` ASC) VISIBLE,
  CONSTRAINT `fk_SuppliersCatalogue_Suppliers1`
    FOREIGN KEY (`SupplierID`)
    REFERENCES `mydb`.`Suppliers` (`SupplierID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_SuppliersCatalogue_MaterialsCatalgoue1`
    FOREIGN KEY (`MaterialID`)
    REFERENCES `mydb`.`MaterialsCatalgoue` (`MaterialID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Storage

```
CREATE TABLE IF NOT EXISTS `mydb`.`Storage` (
  `StorageID` INT NOT NULL,
  `Stock` INT NULL,
  `StorageLocation` VARCHAR(4) NULL,
  `MaterialID` INT NOT NULL,
  UNIQUE INDEX `StorageID_UNIQUE` (`StorageID` ASC) VISIBLE,
  PRIMARY KEY (`StorageID`),
  INDEX `fk_Storage_MaterialsCatalgoue1_idx` (`MaterialID` ASC) VISIBLE,
  CONSTRAINT `fk_Storage_MaterialsCatalgoue1`
    FOREIGN KEY (`MaterialID`)
    REFERENCES `mydb`.`MaterialsCatalgoue` (`MaterialID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Products

```
CREATE TABLE IF NOT EXISTS `mydb`.`Products` (
  `ProductID` INT NOT NULL,
  `Name` VARCHAR(45) NULL,
  `Catergory` INT NULL,
  `Price` FLOAT NULL,
  `Description` LONGTEXT NULL,
  `Weight` FLOAT NULL,
  `Colour` VARCHAR(45) NULL,
  `Dimensions` VARCHAR(12) NULL,
  `Catergory` INT NULL,
  PRIMARY KEY (`ProductID`))
```

ProductMaterials

```
CREATE TABLE IF NOT EXISTS `mydb`.`ProductMaterials` (
  `ProductMaterialsID` INT NOT NULL,
  `StorageID` INT NULL,
  `ProductID` INT NULL,
  `QTY` INT NULL,
  PRIMARY KEY (`ProductMaterialsID`),
  INDEX `fk_ProductMaterials_Storage1_idx` (`StorageID` ASC) VISIBLE,
  INDEX `fk_ProductMaterials_Products1_idx` (`ProductID` ASC) VISIBLE,
  CONSTRAINT `fk_ProductMaterials_Storage1`
    FOREIGN KEY (`StorageID`)
    REFERENCES `mydb`.`Storage` (`StorageID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ProductMaterials_Products1`
    FOREIGN KEY (`ProductID`)
    REFERENCES `mydb`.`Products` (`ProductID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Staff

```
CREATE TABLE IF NOT EXISTS `mydb`.`Staff` (
  `StaffID` INT NOT NULL,
  `FirstName` VARCHAR(45) NULL,
  `LastName` VARCHAR(45) NULL,
  `PhoneNumber` VARCHAR(12) NULL,
  `NINumber` VARCHAR(9) NULL,
  `Address` LONGTEXT NULL,
  `Wage` FLOAT NULL,
  `Position` INT NULL,
  `Email` VARCHAR(45) NULL,
  PRIMARY KEY (`StaffID`))
```

Branch

```
CREATE TABLE IF NOT EXISTS `mydb`.`Branch` (
  `BranchID` INT NOT NULL,
  `Name` VARCHAR(45) NULL,
  `Address1` VARCHAR(45) NULL,
  `Address2` VARCHAR(45) NULL,
  `City` VARCHAR(30) NULL,
  `Region` VARCHAR(45) NULL,
  `Country` VARCHAR(45) NULL,
  `Postcode` VARCHAR(8) NULL,
  `PhoneNumber` VARCHAR(12) NULL,
  `Email` VARCHAR(45) NULL,
  PRIMARY KEY (`BranchID`))
```

BranchStaff

```
CREATE TABLE IF NOT EXISTS `mydb`.`BranchStaff` (
  `BranchID` INT NOT NULL,
  `StaffID` VARCHAR(45) NOT NULL,
  CONSTRAINT `fk_BranchStaff_Branch1`
    FOREIGN KEY (`BranchID`)
    REFERENCES `mydb`.`Branch` (`BranchID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_BranchStaff_Staff1`
    FOREIGN KEY (`StaffID`)
    REFERENCES `mydb`.`Staff` (`StaffID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Shifts

```
CREATE TABLE IF NOT EXISTS `mydb`.`Shifts` (
  `ShiftID` INT NOT NULL,
  `Start` DATETIME NULL,
  `End` DATETIME NULL,
  `BranchID` VARCHAR(45) NULL,
  PRIMARY KEY (`ShiftID`),
  INDEX `fk_Shifts_Branch1_idx` (`BranchID` ASC) VISIBLE,
  CONSTRAINT `fk_Shifts_Branch1`
    FOREIGN KEY (`BranchID`)
    REFERENCES `mydb`.`Branch` (`BranchID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

StaffShifts

```
CREATE TABLE IF NOT EXISTS `mydb`.`StaffShifts` (
  `ShiftID` INT NOT NULL,
  `StaffID` VARCHAR(45) NOT NULL,
  INDEX `fk_StaffShifts_Shifts1_idx` (`ShiftID` ASC) VISIBLE,
  INDEX `fk_StaffShifts_Staff1_idx` (`StaffID` ASC) VISIBLE,
  CONSTRAINT `fk_StaffShifts_Shifts1`
    FOREIGN KEY (`ShiftID`)
    REFERENCES `mydb`.`Shifts` (`ShiftID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_StaffShifts_Staff1`
    FOREIGN KEY (`StaffID`)
    REFERENCES `mydb`.`Staff` (`StaffID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Inventory

```
CREATE TABLE IF NOT EXISTS `mydb`.`Inventory` (
  `InventoryID` INT NOT NULL,
  `ProductID` INT NULL,
  `BranchID` INT NULL,
  `QTY` INT NULL,
  PRIMARY KEY (`InventoryID`),
  INDEX `fk_Inventory_Products1_idx` (`ProductID` ASC) VISIBLE,
  INDEX `fk_Inventory_Branch1_idx` (`BranchID` ASC) VISIBLE,
  CONSTRAINT `fk_Inventory_Products1`
    FOREIGN KEY (`ProductID`)
    REFERENCES `mydb`.`Products` (`ProductID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Inventory_Branch1`
    FOREIGN KEY (`BranchID`)
    REFERENCES `mydb`.`Branch` (`BranchID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Purchases

```
CREATE TABLE IF NOT EXISTS `mydb`.`Purchases` (
  `PurchaseID` INT NOT NULL,
  `CustomerFirstName` VARCHAR(45) NULL,
  `CustomerLastName` VARCHAR(45) NULL,
  `BillingAddress` LONGTEXT NULL,
  `DeliveryAddress` LONGTEXT NULL,
  `Paid` TINYINT NULL,
  `TotalPrice` FLOAT NULL,
  PRIMARY KEY (`PurchaseID`))
```

ProductPurchases

```
CREATE TABLE IF NOT EXISTS `mydb`.`ProductPurchases` (
  `InventoryID` INT NOT NULL,
  `PurchaseID` VARCHAR(45) NOT NULL,
  INDEX `fk_ProductPurchases_Purchases1_idx` (`PurchaseID` ASC) VISIBLE,
  INDEX `fk_ProductPurchases_Inventory1_idx` (`InventoryID` ASC) VISIBLE,
  CONSTRAINT `fk_ProductPurchases_Purchases1`
    FOREIGN KEY (`PurchaseID`)
    REFERENCES `mydb`.`Purchases` (`PurchaseID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ProductPurchases_Inventory1`
    FOREIGN KEY (`InventoryID`)
    REFERENCES `mydb`.`Inventory` (`InventoryID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Warehouse

```
CREATE TABLE IF NOT EXISTS `mydb`.`Warehouse` (
  `WarehouseID` INT NOT NULL,
  `Name` VARCHAR(45) NULL,
  `Address1` VARCHAR(60) NULL,
  `Address2` VARCHAR(60) NULL,
  `City` VARCHAR(45) NULL,
  `Region` VARCHAR(45) NULL,
  `Country` VARCHAR(45) NULL,
  `Postcode` VARCHAR(8) NULL,
  `PhoneNumber` VARCHAR(12) NULL,
  `Email` VARCHAR(45) NULL,
  PRIMARY KEY (`WarehouseID`),
  UNIQUE INDEX `SupplierID_UNIQUE` (`WarehouseID` ASC) VISIBLE)
```

WarehouseProducts

```
CREATE TABLE IF NOT EXISTS `mydb`.`WarehouseProducts` (
  `WarehouseProductID` INT NOT NULL,
  `WarehouseID` INT NOT NULL,
  `ProductID` INT NULL,
  `QTY` INT NULL,
  `Location` VARCHAR(4) NULL,
  PRIMARY KEY (`WarehouseProductID`),
  INDEX `fk_WarehouseProducts_Warehouse1_idx` (`WarehouseID` ASC) VISIBLE,
  INDEX `fk_WarehouseProducts_Products1_idx` (`ProductID` ASC) VISIBLE,
  CONSTRAINT `fk_WarehouseProducts_Warehouse1`
    FOREIGN KEY (`WarehouseID`)
    REFERENCES `mydb`.`Warehouse` (`WarehouseID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_WarehouseProducts_Products1`
    FOREIGN KEY (`ProductID`)
    REFERENCES `mydb`.`Products` (`ProductID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

Orders

```
CREATE TABLE IF NOT EXISTS `mydb`.`Orders` (
  `OrderID` INT NOT NULL,
  `BranchID` INT NULL,
  `OrderDate` DATETIME NULL,
  `Status` INT NULL,
  `WarehouseID` INT NULL,
  PRIMARY KEY (`OrderID`),
  INDEX `fk_Orders_Branch1_idx` (`BranchID` ASC) VISIBLE,
  INDEX `fk_Orders_Warehouse1_idx` (`WarehouseID` ASC) VISIBLE,
  CONSTRAINT `fk_Orders_Branch1`
    FOREIGN KEY (`BranchID`)
    REFERENCES `mydb`.`Branch` (`BranchID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Orders_Warehouse1`
    FOREIGN KEY (`WarehouseID`)
    REFERENCES `mydb`.`Warehouse` (`WarehouseID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

OrderProducts

```
CREATE TABLE IF NOT EXISTS `mydb`.`OrderProducts` (
  `OrderProductsID` INT NOT NULL,
  `ProductID` INT NOT NULL,
  `OrderID` INT NOT NULL,
  `qty` INT NOT NULL,
  PRIMARY KEY (`OrderProductsID`),
  INDEX `fk_OrderProducts_Products1_idx` (`ProductID` ASC) VISIBLE,
  INDEX `fk_OrderProducts_Orders1_idx` (`OrderID` ASC) VISIBLE,
  CONSTRAINT `fk_OrderProducts_Products1`
    FOREIGN KEY (`ProductID`)
    REFERENCES `mydb`.`Products` (`ProductID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_OrderProducts_Orders1`
    FOREIGN KEY (`OrderID`)
    REFERENCES `mydb`.`Orders` (`OrderID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```