CPSC 304 Project Cover Page

Milestone #: 2

Date: 25/09/2024

Group Number: 46

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Andrew Chen	42072777	x6b9i	Iphoneotto3@gmail.com
Helen Ma	44776268	n0j9u	helenjym@gmail.com
Stephanie Gao	38000618	y2m7b	stephmimi27@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Deliverables

All the following items must be put together into a **single** PDF file.

1. A completed cover page (template on Canvas)

2. A brief (~2-3 sentences) summary of your project. Many of your TAs are managing multiple projects, so this will help them remember the details of your project.

3. The ER diagram you are basing your item #3 (below) on. This ER diagram may be the same as your milestone 1 submission, or it might be different. If you have made changes from the version submitted in milestone 1, attach a note indicating what changes have been made and why.

4. The schema derived from your ER diagram (above). For the translation of the ER diagram to the relational model, follow the same instructions as in your lectures. The process should be reasonably straightforward. For each table:
    a. List the table definition (e.g., Table1(attr1: domain1, attr2: domain2, ...)). Make sure to include the domains for each attribute.
    b. Specify the primary key (PK), candidate key, (CK) foreign keys (FK), and other constraints (e.g., not null, unique, etc.) that the table must maintain.

5. Functional Dependencies (FDs)
    a. Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key.

6. NOTmalization
    a. NOTmalize each of your tables to be in 3NF or BCNF. Give the list of tables, their primary keys, their candidate keys, and their foreign keys after NOTmalization.

7. The SQL DDL statements required to create all the tables from item #6. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

8. INSERT statements to populate each table with at least 5 tuples. You will likely want to have more than 5 tuples so that you can have meaningful queries later.

Project Description (Brief)

This project is designed to create a database system for managing **warehouse operations** and **supply chain logistics**. The system focuses on efficiently tracking inventory levels, maintaining supplier relationships, and managing the process of restocking products when stock levels fall below a certain threshold. The goal is, therefore, to ensure that products are available without overstocking, enabling warehouse managers or business owners to make informed decisions about when and how much to restock.

Database Specifications

The database will allow users (like business owners or warehouse managers) to:

- Track inventory levels and determine when to restock.
- Manage relationships with suppliers, including placing and tracking orders.
- View product details and restock history.

The database will also store details about shipments, payments, and inventory availability in multiple warehouses, providing a comprehensive solution for supply chain management.

Description of the application platform

We will use the department-provided **Oracle** for our database and **HTML/CSS & JavaScript** to implement the application in our technology stack.

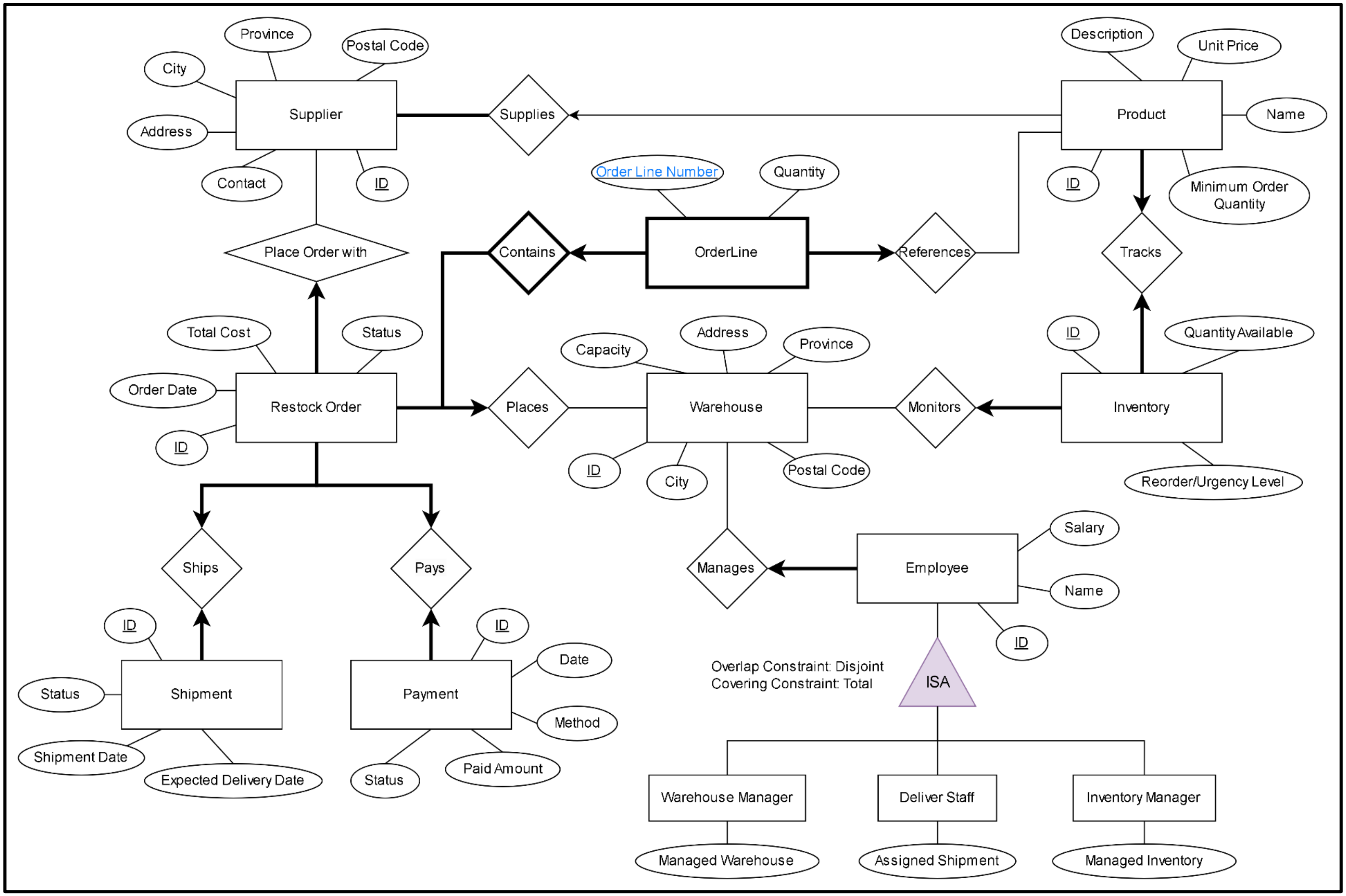
ER-Diagram

NOTE: The blue underlined is a partial key.

CHANGE:

- Additional attributes are added to entities:
    - **WarehouseManager** → Team Size
    - **Deliver Staff** → Delivery Vehicle
    - **InventoryManager** → Specialization
    - **Employee** → Salary
    - **Supplier** → City, Province, Postal Code
    - **Warehouse** → City, Province, Postal Code

- Participation & cardinality modifications:
    - **Supplier → RestockOrder**
        - Cardinality: Many **RestockOrder** can be placed by one **Supplier**.
        - Participation: Total on the **RestockOrder** side.
    - **Inventory → Product**
        - Cardinality: An **Inventory** tracks only one **Product**.
        - Participation: Total on both sides.
    - **Warehouse → Inventory**
        - Cardinality: Many **Inventory** can be monitored by one **Warehouse**.
        - Participation: Total on the **Inventory** Side.
    - **Warehouse → Employee**
        - Cardinality: Many **Employees** can be managed by one **Warehouse**.
        - Participation: Total on the **Employee** Side.


Supplier
Province
Postal Code
City
Address
Contact
ID
Supplies
Product
Description
Unit Price
Name
ID
Minimum Order Quantity
Order Line Number
Quantity
OrderLine
Contains
References
Tracks
Place Order with
Total Cost
Status
Order Date
ID
Restock Order
Places
Capacity
Address
Province
Warehouse
ID
City
Postal Code
Monitors
ID
Quantity Available
Inventory
Reorder/Urgency Level
Ships
Pays
Manages
Employee
Salary
Name
ID
ISA
Overlap Constraint: Disjoint
Covering Constraint: Total
ID
Status
Shipment
Shipment Date
Expected Delivery Date
ID
Date
Payment
Method
Status
Paid Amount
Warehouse Manager
Deliver Staff
Inventory Manager
Managed Warehouse
Assigned Shipment
Managed Inventory

Schema

Warehouse(WID: char(10), Capacity: int, Address: varchar(255), City: varchar(255), Province: varchar(255), PostalCode: varchar(10))

PK: (WID)

Employee(EID: char(10), Name: varchar(255), Salary: decimal(10, 2), **WID**: char(10))

PK: (EID)

FK: (WID references Warehouse ID)

NOT NULL: (Name, WID)

WareHouseManager(**EID**: char(10), TeamSize: int)

PK: (EID)

FK: (EID references Employee ID)

InventoryManager(**EID**: char(10), DeliveryVehicle: varchar(255))

PK: (EID)

FK: (EID references Employee ID)

DeliveryStaff(**EID**: char(10), Specialization: varchar(255))

PK: (EID)

FK: (EID references Employee ID)

Inventory (IID: char(10), QuantityAvailable: int, ReorderLevel: int, **WID**: char(10), **PID**: char(10))

PK: (IID)

FK: (WID references Warehouse ID)
(PID references Product ID)

CK: (PID)

NOT NULL: (QuantityAvailable, ReorderLevel, WID, PID)

UNIQUE: (PID)

Product(PID: char(10), Name: varchar(255), Description: varchar(255), UnitPrice: decimal(10, 2), MinimumOrderQuantity: int, **SID**: char(10))
PK: (PID)
FK: (SID references Supplier ID)

Supplier(SID: char(10), Contact: varchar(255), Address: varchar(255), City: varchar(255), Province: varchar(255), PostalCode: varchar(10))
PK: (SID)
ASSERTION & TRIGGER: (Every Supplier → Product)

RestockOrder(ROID: char(10), Status: ENUM, OrderDate: date, TotalCost: decimal(10, 2), **SID**: char(10), **WID**: char(10))
PK: (ROID)
FK: (WID references Warehouse ID)
(SID references Supplier ID)
NOT NULL: (Status, SID, WID)
ENUM('Pending', 'Confirmed', 'Processing', 'Completed', 'Returned')
ASSERTION & TRIGGER: (Every RestockOrder → OrderLine)

Shipment(ShipID: char(10), Status: ENUM, ShipmentDate: date, ExpectedDeliveryDate: date, **ROID**: char(10))
PK: (ShipID)
FK: (ROID references RestockOrder ID)
CK: (RIOD)
NOT NULL: (Status, ROID)
UNIQUE: (ROID)
ENUM('Shipped', 'In Transit', 'Delayed', 'Delivered')

Payment(<u>PayID</u>: char(10), Status: ENUM, Date: date, PaymentMethod: varchar(50), AmountPaid: decimal(10, 2), **ROID**: char(10))

PK: (PayID)
FK: (ROID references RestockOrder ID)
CK: (RIOD)
NOT NULL: (Status, ROID)
UNIQUE: (ROID)
ENUM('Pending`, 'Processing', 'Completed', 'Failed', 'Cancelled', 'Refunded')

OrderLine(<u>OID</u>: char(10), QuantityOrder: int, **PID**: char(10), **<u>ROID</u>**: char(10))

PK: (ROID, OID)
FK: (ROID references RestockOrder ID)
(PID references Product ID)
NOT NULL: (QuantityOrder, PID)

NOTE: Assertions & Triggers are needed.

CHANGE: OrderLine Number → OID.

FD

Warehouse(WID, Capacity, Address, City, Province, PostalCode)

WID → Capacity, Address, City, Province, PostalCode
PostalCode → City, Province

Employee(EID, Name, Salary, **WID**)

EID → Name, Salary, **WID**

WareHouseManager(**EID**, TeamSize)

EID → TeamSize

InventoryManager(**EID**, DeliveryVehicle)

EID → DeliveryVehicle

DeliveryStaff(**EID**, Specialization)

EID → Specialization

Inventory (IID, QuantityAvailable, ReorderLevel, **WID**, **PID**)

IID → QuantityAvailable, ReorderLevel, **WID**, **PID**
PID → QuantityAvailable, ReorderLevel, **WID**, IID

Product(PID, Name, Description, UnitPrice, MinimumOrderQuantity, **SID**)

PID → Name, Description, UnitPrice, MinimumOrderQuantity, **SID**
Description → Name

Supplier(SID, Contact, Address, City, Province, PostalCode)

SID → Contact, Address, City, Province, PostalCode
PostalCode → City, Province

RestockOrder(ROID, Status, OrderDate, TotalCost, **SID**, **WID**)

ROID → Status, OrderDate, TotalCost, **SID**, **WID**

Shipment(ShipID, Status, ShipmentDate, ExpectedDeliveryDate, **ROID**)

ShipID → Status, ShipmentDate, ExpectedDeliveryDate, **ROID**

ROID → Status, ShipmentDate, ExpectedDeliveryDate, ShipID

Payment(PayID, Status, Date, PaymentMethod, AmountPaid, **ROID**)

PayID → Status, Date, PaymentMethod, AmountPaid, **ROID**

ROID → Status, Date, PaymentMethod, AmountPaid, PayID

OrderLine(OID, QuantityOrder, **PID**, **ROID**)

OID, ROID → QuantityOrder, **PID**

Normalization

Warehouse(WID, Capacity, Address, City, Province, PostalCode)

WID → Capacity, Address, City, Province, PostalCode
PostalCode → City, Province

- First, let's find the closure of all the FDs and the keys of the relation.

  $Postal\ Code^{+} = \{PostalCode, City, Province\}$

  $WID^{+} = \{WID, Capacity, Address, PostalCode, City, Province\}$

- The only key is WID.
- Now, let's find the minimal cover of FDs.

  PostalCode → City

  PostalCode → Province

  WID → PostalCode

  WID → Capacity

  WID → Address

- Since PostalCode → City violates 3NF, we decompose R to get R_1(PostalCode, City) & R_2(**PostalCode**, WID, Capacity, Address, Province).
- PostalCode → Province violates 3NF, we decompose R_2 to get R_3(**PostalCode**, Province) & R_4(**PostalCode**, WID, Capacity, Address).
- There are no more violations, so the final answer is:

Warehouse(<u>WID</u>: char(10), Capacity: int, Address: varchar(255), **PostalCode**: varchar(10))
PK: (WID)
FK: (PostalCode references WarehouseProvince PostalCode)

WarehouseProvince(<u>**PostalCode**</u>: varchar(10), Province: varchar(255))
PK: (PostalCode)
FK: (PostalCode references WarehouseCity PostalCode)

WarehouseCity(<u>PostalCode</u>: varchar(10), City: varchar(255))
PK: (PostalCode)

Product(PID, Name, Description, UnitPrice, MinimumOrderQuantity, **SID**)

PID → Name, Description, UnitPrice, MinimumOrderQuantity, **SID**)
Description → Name

- First, let's find the closure of all the FDs and the keys of the relation.

$Description^{+} = \{Description, Name\}$

$PID^{+} = \{PID, Name, Description, UnitPrice, MinimumOrderQuantity, SID\}$

- The only key is PID.
- Now, let's find the minimal cover of FDs.

Description → Name
PID → Description
PID → UnitPrice
PID → MinimumOrderQuantity
PID → SID

- Since Description → Name violates 3NF, we decompose R to get

R_1(Description, Name) & R_2(**Description**, PID, UnitPrice, MinimumOrderQuantity, **SID**).

- There are no more violations, so the final answer is:

Product(PID: char(10), **Description**: varchar(255), UnitPrice: decimal(10, 2), MinimumOrderQuantity: int, **SID**: char(10))
PK: (PID)
FK: (SID references Supplier ID)
(**Description** references ProductName Description)

ProductName(**Description**: varchar(255), Name: varchar(255)
PK: (Description)

Supplier(SID, Contact, Address, City, Province, PostalCode)

SID → Contact, Address, City, Province, PostalCode
PostalCode → City, Province

- First, let's find the closure of all the FDs and the keys of the relation.

  $Postal\ Code^{+} = \{PostalCode, City, Province\}$

  $SID^{+} = \{SID, Contact, Address, PostalCode, City, Province\}$

- The only key is SID.
- Now, let's find the minimal cover of FDs.

  PostalCode → City

  PostalCode → Province

  SID → PostalCode

  SID → Contact

  SID → Address

- Since PostalCode → City violates 3NF, we decompose R to get
  R_1(PostalCode, City) & R_2(**PostalCode**, SID, Contact, Address, Province).
- PostalCode → Province violates 3NF, we decompose R_2 to get
  R_3(**PostalCode**, Province) & R_4(**PostalCode**, SID, Contact, Address).
- There are no more violations, so the final answer is:

Supplier(SID: char(10), Address: varchar(255), **PostalCode**: varchar(10))
PK: (SID)
FK: (PostalCode references SupplierProvince PostalCode)

SupplierProvince (**PostalCode**: varchar(10), Province: varchar(255))
PK: (PostalCode)
FK: (PostalCode references SupplierCity PostalCode)

SupplierCity(PostalCode: varchar(10), City: varchar(255))
PK: (PostalCode)

After Normalization

Warehouse(WID: char(10), Capacity: int, Address: varchar(255), **PostalCode**: varchar(10))
PK: (WID)
FK: (PostalCode references WarehouseProvince PostalCode)

WarehouseProvince(**PostalCode**: varchar(10), Province: varchar(255))
PK: (PostalCode)
FK: (PostalCode references WarehouseCity PostalCode)

WarehouseCity(PostalCode: varchar(10), City: varchar(255))
PK: (PostalCode)

Employee(EID: char(10), Name: varchar(255), Salary: decimal(10, 2), **WID**: char(10))
PK: (EID)
FK: (WID references Warehouse ID)
NOT NULL: (Name, WID)

WareHouseManager(**EID**: char(10), TeamSize: int)
PK: (EID)
FK: (EID references Employee ID)

InventoryManager(**EID**: char(10), DeliveryVehicle: varchar(255))
PK: (EID)
FK: (EID references Employee ID)

DeliveryStaff(**EID**: char(10), Specialization: varchar(255))
PK: (EID)
FK: (EID references Employee ID)

Inventory (<u>IID</u>: char(10), QuantityAvailable: int, ReorderLevel: int, **WID**: char(10), **PID**: char(10))

PK: (IID)
FK: (WID references Warehouse ID)
(PID references Product ID)
CK: (PID)
NOT NULL: (QuantityAvailable, ReorderLevel, WID, PID)
UNIQUE: (PID)

Product(<u>PID</u>: char(10), **Description**: varchar(255), UnitPrice: decimal(10, 2), MinimumOrderQuantity: int, **SID**: char(10))

PK: (PID)
FK: (SID references Supplier ID)
(**Description** references ProductName Description)

ProductName(**Description**: varchar(255), Name: varchar(255)

PK: (Description)

Supplier(<u>SID</u>: char(10), Address: varchar(255), **PostalCode**: varchar(10))

PK: (SID)
FK: (PostalCode references SupplierProvince PostalCode)
ASSERTION & TRIGGER: (Every Supplier → Product)

SupplierProvince (<u>**PostalCode**</u>: varchar(10), Province: varchar(255))

PK: (PostalCode)
FK: (PostalCode references SupplierCity PostalCode)

SupplierCity(<u>PostalCode</u>: varchar(10), City: varchar(255))

PK: (PostalCode)

RestockOrder(ROID: char(10), Status: ENUM, OrderDate: date, TotalCost: decimal(10, 2), **SID**: char(10), **WID**: char(10))

PK: (ROID)
FK: (WID references Warehouse ID)
(SID references Supplier ID)
NOT NULL: (Status, SID, WID)
ENUM('Pending', 'Confirmed', 'Processing', 'Completed', 'Returned')
ASSERTION & TRIGGER: (Every RestockOrder → OrderLine)

Shipment(ShipID: char(10), Status: ENUM, ShipmentDate: date, ExpectedDeliveryDate: date, **ROID**: char(10))

PK: (ShipID)
FK: (ROID references RestockOrder ID)
CK: (RIOD)
NOT NULL: (Status, ROID)
UNIQUE: (ROID)
ENUM('Shipped', 'In Transit', 'Delayed', 'Delivered')

Payment(PayID: char(10), Status: ENUM, Date: date, PaymentMethod: varchar(50), AmountPaid: decimal(10, 2), **ROID**: char(10))

PK: (PayID)
FK: (ROID references RestockOrder ID)
CK: (RIOD)
NOT NULL: (Status, ROID)
UNIQUE: (ROID)
ENUM('Pending`, 'Processing', 'Completed', 'Failed', 'Cancelled', 'Refunded')

OrderLine(OID: char(10), QuantityOrder: int, **PID**: char(10), **ROID**: char(10))

PK: (ROID, OID)
FK: (ROID references RestockOrder ID)
(PID references Product ID)
NOT NULL: (QuantityOrder, PID)

SQL DDL Create Statements

```
CREATE TABLE Warehouse (
        WID CHAR(10) PRIMARY KEY,
        Capacity INT,
        Address VARCHAR(255),
        PostalCode VARCHAR(10),
        FOREIGN KEY (PostalCode) REFERENCES WarehouseProvince(PostalCode)
);

CREATE TABLE WarehouseProvince (
        PostalCode VARCHAR(10) PRIMARY KEY,
        Province VARCHAR(255)
        FOREIGN KEY (PostalCode) REFERENCES WarehouseCity(PostalCode)
);

CREATE TABLE WarehouseCity (
        PostalCode VARCHAR(10) PRIMARY KEY,
        City VARCHAR(255)
);

CREATE TABLE Employee (
        EID CHAR(10) PRIMARY KEY,
        Name VARCHAR(255) NOT NULL,
        Salary DECIMAL(10, 2),
        WID CHAR(10) NOT NULL,
        FOREIGN KEY (WID) REFERENCES Warehouse(WID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);
```

```
CREATE TABLE WarehouseManager (
        EID CHAR(10) PRIMARY KEY,
        TeamSize INT,
        FOREIGN KEY (EID) REFERENCES Employee(EID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);

CREATE TABLE InventoryManager (
        EID CHAR(10) PRIMARY KEY,
        DeliveryVehicle VARCHAR(255),
        FOREIGN KEY (EID) REFERENCES Employee(EID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);

CREATE TABLE DeliveryStaff (
        EID CHAR(10) PRIMARY KEY,
        Specialization VARCHAR(255),
        FOREIGN KEY (EID) REFERENCES Employee(EID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);
```

```
CREATE TABLE Inventory (
	IID CHAR(10) PRIMARY KEY,
	QuantityAvailable INT NOT NULL,
	ReorderLevel INT NOT NULL,
	WID CHAR(10) NOT NULL,
	PID CHAR(10) NOT NULL,
	UNIQUE (PID),
	FOREIGN KEY (WID) REFERENCES Warehouse(WID),
		ON DELETE CASCADE
		ON UPDATE CASCADE
	FOREIGN KEY (PID) REFERENCES Product(PID)
		ON DELETE CASCADE
		ON UPDATE CASCADE
);

CREATE TABLE Product (
	PID CHAR(10) PRIMARY KEY,
	Description VARCHAR(255),
	UnitPrice DECIMAL(10, 2),
	MinimumOrderQuantity INT,
	SID CHAR(10),
	FOREIGN KEY (SID) REFERENCES Supplier(SID)
		ON DELETE CASCADE
		ON UPDATE CASCADE
	FOREIGN KEY (Description) REFERENCES ProductName (Description)
		ON DELETE CASCADE
		ON UPDATE CASCADE
);

CREATE TABLE ProductName (
	Description VARCHAR(255) PRIMARY KEY,
	Name VARCHAR(255)
);
```

```
CREATE TABLE Supplier (
        SID CHAR(10) PRIMARY KEY,
        Address VARCHAR(255),
        PostalCode VARCHAR(10),
        FOREIGN KEY (PostalCode) REFERENCES SupplierProvince(PostalCode)
);

CREATE TABLE SupplierProvince (
        PostalCode VARCHAR(10) PRIMARY KEY,
        Province VARCHAR(255)
        FOREIGN KEY (PostalCode) REFERENCES SupplierCity(PostalCode)
);

CREATE TABLE SupplierCity (
        PostalCode VARCHAR(10) PRIMARY KEY,
        City VARCHAR(255)
);

CREATE TABLE RestockOrder (
        ROID CHAR(10) PRIMARY KEY,
        Status ENUM('Pending', 'Confirmed',
                        'Processing', 'Completed',
                        'Returned') NOT NULL,
        OrderDate DATE,
        TotalCost DECIMAL(10, 2),
        SID CHAR(10) NOT NULL,
        WID CHAR(10) NOT NULL,
        FOREIGN KEY (SID) REFERENCES Supplier(SID),
                ON DELETE CASCADE
                ON UPDATE CASCADE
        FOREIGN KEY (WID) REFERENCES Warehouse(WID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);
```

```
CREATE TABLE Shipment (
        ShipID CHAR(10) PRIMARY KEY,
        Status ENUM('Shipped', 'In Transit', 'Delayed', 'Delivered') NOT NULL,
        ShipmentDate DATE,
        ExpectedDeliveryDate DATE,
        ROID CHAR(10) NOT NULL,
        UNIQUE(ROID),
        FOREIGN KEY (ROID) REFERENCES RestockOrder(ROID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);

CREATE TABLE Payment (
        PayID CHAR(10) PRIMARY KEY,
        Status ENUM('Pending', 'Processing',
                        'Completed', 'Failed',
                        'Cancelled', 'Refunded') NOT NULL,
        Date DATE,
        PaymentMethod VARCHAR(50),
        AmountPaid DECIMAL(10, 2),
        ROID CHAR(10) NOT NULL,
        UNIQUE(ROID)
        FOREIGN KEY (ROID) REFERENCES RestockOrder(ROID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);
```

```
CREATE TABLE OrderLine (
        OID CHAR(10),
        QuantityOrder INT NOT NULL,
        PID CHAR(10) NOT NULL,
        ROID CHAR(10),
        PRIMARY KEY (ROID, OID),
        FOREIGN KEY (ROID) REFERENCES RestockOrder(ROID),
                ON DELETE CASCADE
                ON UPDATE CASCADE
        FOREIGN KEY (PID) REFERENCES Product(PID)
                ON DELETE CASCADE
                ON UPDATE CASCADE
);
```

Insert Statements

NOTE: Assume all IDs are 5 characters long (just for this part).

```
INSERT INTO Warehouse (WID, Capacity, Address, PostalCode)
VALUES
        ('W001', 5000, '1234 Warehouse Lane', 'A1B2C3'),
        ('W002', 3000, '5678 Storage Blvd', 'D4E5F6'),
        ('W003', 10000, '910 Warehouse Rd', 'G7H8I9'),
        ('W004', 7500, '1111 Distribution Dr', 'J1K2L3'),
        ('W005', 4000, '2222 Logistics St', 'M4N5O6'),
        ('W006', 6000, '9876 Transit St', 'P1Q2R3'),
        ('W007', 4500, '8888 Industrial Rd', 'S4T5U6'),
        ('W008', 8500, '3333 Distribution Ave', 'V7W8X9'),
        ('W009', 5000, '4444 Logistics Rd', 'Y1Z2A3'),
        ('W010', 7000, '5555 Storage Lane', 'B2C3D4');
```

```
INSERT INTO WarehouseProvince (PostalCode, Province)
VALUES
        ('A1B2C3', 'Ontario'),
        ('D4E5F6', 'Quebec'),
        ('G7H8I9', 'British Columbia'),
        ('J1K2L3', 'Alberta'),
        ('M4N5O6', 'Nova Scotia'),
        ('P1Q2R3', 'Manitoba'),
        ('S4T5U6', 'Saskatchewan'),
        ('V7W8X9', 'Newfoundland'),
        ('Y1Z2A3', 'Prince Edward Island'),
        ('B2C3D4', 'New Brunswick');

INSERT INTO WarehouseCity (PostalCode, City)
VALUES
        ('A1B2C3', 'Toronto'),
        ('D4E5F6', 'Montreal'),
        ('G7H8I9', 'Vancouver'),
        ('J1K2L3', 'Calgary'),
        ('M4N5O6', 'Halifax'),
        ('P1Q2R3', 'Winnipeg'),
        ('S4T5U6', 'Regina'),
        ('V7W8X9', 'St. John\'s'),
        ('Y1Z2A3', 'Charlottetown'),
        ('B2C3D4', 'Fredericton');
```

```
INSERT INTO Employee (EID, Name, Salary, WID)
VALUES
        ('E001', 'John Doe', 60000.00, 'W001'),
        ('E002', 'Jane Smith', 55000.00, 'W002'),
        ('E003', 'Mike Johnson' , 40000.00, 'W003'),
        ('E004', 'Emily Davis', 62000.00, 'W004'),
        ('E005', 'Sarah Brown', 56000.00, 'W005'),
        ('E006', 'Chris Green', 45000.00, 'W006'),
        ('E007', 'Anna Lee', 59000.00, 'W007'),
        ('E008', 'Paul White', 57000.00, 'W008'),
        ('E009', 'Laura Black', 42000.00, 'W009'),
        ('E010', 'James Gold', 61000.00, 'W010');

INSERT INTO WarehouseManager (EID, TeamSize)
VALUES
        ('E001', 10),
        ('E004', 8),
        ('E007', 9),
        ('E010', 7)
        ('E012', 15);
```

```
INSERT INTO InventoryManager (EID, DeliveryVehicle)
VALUES
        ('E002', 'Van - ABC123'),
        ('E005', 'Truck - DEF456'),
        ('E008', 'Truck - GHI789'),
        ('E017', 'Truck - JKLM123'),
        ('E018', 'Van - QRS567');

INSERT INTO DeliveryStaff (EID, Specialization)
VALUES
        ('E003', 'Electronics Delivery'),
        ('E006', 'Furniture Delivery'),
        ('E009', 'Perishable Goods Delivery'),
        ('E024', 'Medical Equipment Delivery'),
        ('E025', 'Heavy Machinery Delivery');

INSERT INTO Inventory (IID, QuantityAvailable, ReorderLevel, WID, PID)
VALUES
        ('I001', 100, 20, 'W001', 'P001'),
        ('I002', 50, 15, 'W002', 'P002'),
        ('I003', 200, 25, 'W003', 'P003'),
        ('I004', 300, 30, 'W004', 'P004'),
        ('I005', 120, 10, 'W005', 'P005'),
        ('I006', 80, 10, 'W006', 'P006'),
        ('I007', 150, 20, 'W007', 'P007'),
        ('I008', 400, 35, 'W008', 'P008'),
        ('I009', 90, 15, 'W009', 'P009'),
        ('I010', 130, 25, 'W010', 'P010');
```

```
INSERT INTO Product (PID, Description, UnitPrice, MinimumOrderQuantity, SID)
VALUES
        ('P001', 'Resistor', 0.50, 100, 'S001'),
        ('P002', 'Capacitor', 0.30, 150, 'S002'),
        ('P003', 'Inductor', 0.70, 200, 'S003'),
        ('P004', 'Transistor', 1.00, 80, 'S004'),
        ('P005', 'Diode', 0.25, 120, 'S005'),
        ('P006', 'LED', 0.15, 90, 'S006'),
        ('P007', 'Fuse', 0.20, 100, 'S007'),
        ('P008', 'Switch', 0.60, 110, 'S008'),
        ('P009', 'Battery', 1.20, 50, 'S009'),
        ('P010', 'Wire', 0.10, 300, 'S010');

INSERT INTO ProductName (Description, Name)
VALUES
        ('Resistor', 'Electronic Component - Resistor'),
        ('Capacitor', 'Electronic Component - Capacitor'),
        ('Inductor', 'Electronic Component - Inductor'),
        ('Transistor', 'Electronic Component - Transistor'),
        ('Diode', 'Electronic Component - Diode'),
        ('LED', 'Light Emitting Diode'),
        ('Fuse', 'Electronic Component - Fuse'),
        ('Switch', 'Electrical Switch'),
        ('Battery', 'Energy Storage Battery'),
        ('Wire', 'Electrical Wire');
```

```
INSERT INTO Supplier (SID, Address, PostalCode)
VALUES
        ('S001', '123 Supply Rd', 'A1B2C3'),
        ('S002', '456 Vendor St', 'D4E5F6'),
        ('S003', '789 Wholesaler Blvd', 'G7H8I9'),
        ('S004', '101 Supplier Ln', 'J1K2L3'),
        ('S005', '202 Distributor Dr', 'M4N5O6'),
        ('S006', '303 Merchant Ave', 'P1Q2R3'),
        ('S007', '404 Trade St', 'S4T5U6'),
        ('S008', '505 Exchange Blvd', 'V7W8X9'),
        ('S009', '606 Commerce St', 'Y1Z2A3'),
        ('S010', '707 Marketplace Ln', 'B2C3D4');

INSERT INTO SupplierProvince (PostalCode, Province)
VALUES
        ('A1B2C3', 'Ontario'),
        ('D4E5F6', 'Quebec'),
        ('G7H8I9', 'British Columbia'),
        ('J1K2L3', 'Alberta'),
        ('M4N5O6', 'Nova Scotia'),
        ('P1Q2R3', 'Manitoba'),
        ('S4T5U6', 'Saskatchewan'),
        ('V7W8X9', 'Newfoundland and Labrador'),
        ('Y1Z2A3', 'Prince Edward Island'),
        ('B2C3D4', 'New Brunswick');
```

```
INSERT INTO SupplierCity (PostalCode, City)
VALUES
        ('A1B2C3', 'Toronto'),
        ('D4E5F6', 'Montreal'),
        ('G7H8I9', 'Vancouver'),
        ('J1K2L3', 'Calgary'),
        ('M4N5O6', 'Halifax'),
        ('P1Q2R3', 'Winnipeg'),
        ('S4T5U6', 'Regina'),
        ('V7W8X9', 'St. John\'s'),
        ('Y1Z2A3', 'Charlottetown'),
        ('B2C3D4', 'Fredericton');

INSERT INTO OrderLine (OID, QuantityOrder, PID, ROID)
VALUES
        ('O001', 100, 'P001', 'RO001'),
        ('O002', 150, 'P002', 'RO002'),
        ('O003', 200, 'P003', 'RO003'),
        ('O004', 80, 'P004', 'RO004'),
        ('O005', 120, 'P005', 'RO005'),
        ('O006', 250, 'P006', 'RO006'),
        ('O007', 300, 'P007', 'RO007'),
        ('O008', 350, 'P008', 'RO008'),
        ('O009', 90, 'P009', 'RO009'),
        ('O010', 400, 'P010', 'RO010');
```

```
INSERT INTO RestockOrder (ROID, Status, OrderDate, TotalCost, SID, WID)
VALUES
        ('RO001', 'Pending', '2024-01-01', 5000.00, 'S001', 'W001'),
        ('RO002', 'Confirmed', '2024-01-05', 3000.00, 'S002', 'W002'),
        ('RO003', 'Processing', '2024-01-10', 7000.00, 'S003', 'W003'),
        ('RO004', 'Completed', '2024-01-15', 6000.00, 'S004', 'W004'),
        ('RO005', 'Returned', '2024-01-20', 2000.00, 'S005', 'W005'),
        ('RO006', 'Pending', '2024-01-25', 4000.00, 'S006', 'W006'),
        ('RO007', 'Confirmed', '2024-02-01', 2500.00, 'S007', 'W007'),
        ('RO008', 'Processing', '2024-02-05', 8000.00, 'S008', 'W008'),
        ('RO009', 'Completed', '2024-02-10', 7500.00, 'S009', 'W009'),
        ('RO010', 'Returned', '2024-02-15', 3000.00, 'S010', 'W010');

INSERT INTO Shipment (ShipID, Status, ShipmentDate, ExpectedDeliveryDate, ROID)
VALUES
        ('SHP001', 'Shipped', '2024-01-05', '2024-01-10', 'RO001'),
        ('SHP002', 'In Transit', '2024-01-10', '2024-01-15', 'RO002'),
        ('SHP003', 'Delayed', '2024-01-15', '2024-01-20', 'RO003'),
        ('SHP004', 'Delivered', '2024-01-20', '2024-01-25', 'RO004'),
        ('SHP005', 'Shipped', '2024-01-25', '2024-01-30', 'RO005'),
        ('SHP006', 'In Transit', '2024-02-01', '2024-02-05', 'RO006'),
        ('SHP007', 'Shipped', '2024-02-05', '2024-02-10', 'RO007'),
        ('SHP008', 'Delivered', '2024-02-10', '2024-02-15', 'RO008'),
        ('SHP009', 'In Transit', '2024-02-12', '2024-02-18', 'RO009'),
        ('SHP010', 'Delayed', '2024-02-14', '2024-02-20', 'RO010');
```

```
INSERT INTO Payment (PayID, Status, Date, PaymentMethod, AmountPaid, ROID)
VALUES
        ('PAY001', 'Pending', '2024-01-01', 'Credit Card', 5000.00, 'RO001'),
        ('PAY002', 'Processing', '2024-01-05', 'Bank Transfer', 3000.00, 'RO002'),
        ('PAY003', 'Completed', '2024-01-10', 'Cash', 7000.00, 'RO003'),
        ('PAY004', 'Failed', '2024-01-15', 'Credit Card', 6000.00, 'RO004'),
        ('PAY005', 'Refunded', '2024-01-20', 'Bank Transfer', 2000.00, 'RO005'),
        ('PAY006', 'Completed', '2024-01-25', 'Credit Card', 4000.00, 'RO006'),
        ('PAY007', 'Processing', '2024-02-01', 'Cash', 2500.00, 'RO007'),
        ('PAY008', 'Pending', '2024-02-05', 'Bank Transfer', 8000.00, 'RO008'),
        ('PAY009', 'Completed', '2024-02-10', 'Cash', 7500.00, 'RO009'),
        ('PAY010', 'Failed', '2024-02-15', 'Credit Card', 3000.00, 'RO010');
```