ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO HỌC PHẦN CƠ SỞ DỮ LIỆU

ĐỀ TÀI: QUẢN LÝ KHO CẢNG

Tên nhóm: nhóm 5

Họ tên thành viên: Trần Trung Hậu

MSSV: 23020061

Họ tên thành viên: Mai Minh Tùng

MSSV: 23020432

Lớp học phần: INT2211_37

Giảng viên hướng dẫn: TS.Trần Hồng Việt

Hà Nội, 12/2024

Choose topic: Port Management Database System

Objective:
To develop a database system for managing operations at a port, ensuring efficient tracking and scheduling of ship arrivals, dock assignments, and departures.

Key Requirements:

1. **Data Management:**
    - Maintain records of ships, including unique identifiers (ShipID), name, and capacity.
    - Manage docks with information on DockID, dock type, and availability.
2. **Scheduling:**
    - Track the arrival and departure dates for ships at specific docks.
    - Prevent scheduling conflicts where two ships are assigned to the same dock simultaneously.
3. **Operations Tracking:**
    - Record and monitor cargo handling activities, including goods loaded or unloaded at the port.
    - Provide a history of ship dockings and operations for future reference.
4. **Efficiency and Optimization:**
    - Ensure optimal utilization of docks based on ship requirements and dock capacity.
    - Minimize delays by forecasting and resolving potential scheduling conflicts.
5. **Reporting:**

    - Generate daily, weekly, and monthly reports summarizing port activities.
    - Track key performance indicators (e.g., average docking time, cargo volume handled).

End Users:

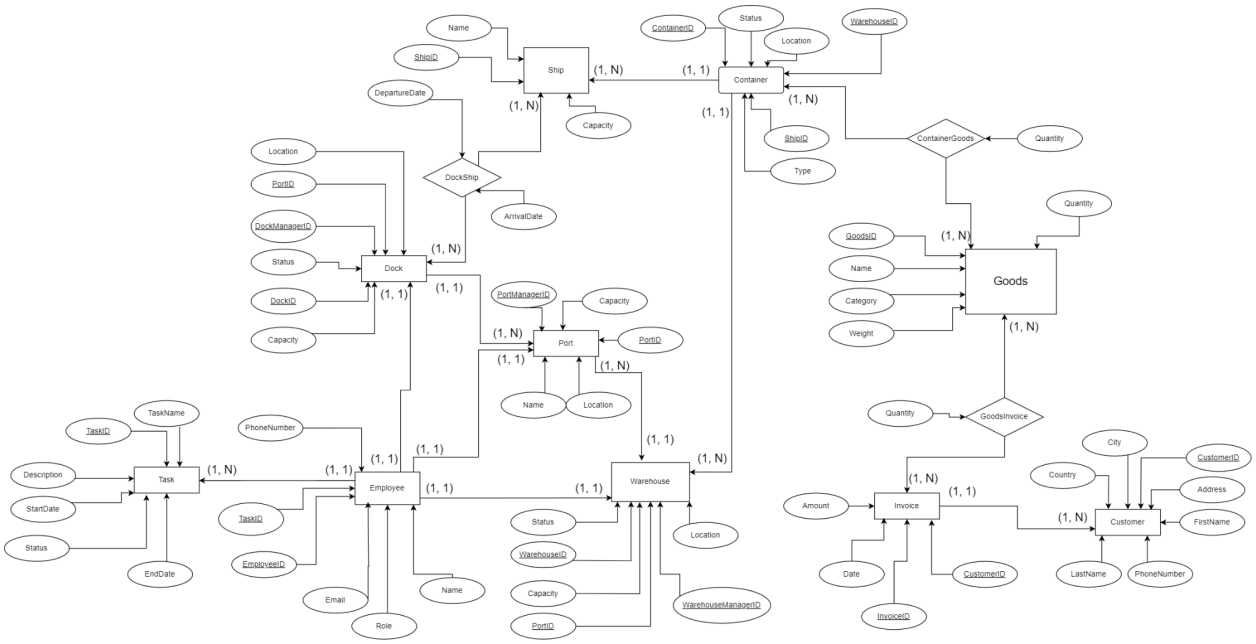
- Port administrators for scheduling and resource allocation.
- Cargo managers for tracking shipments.
- Shipping companies for monitoring docking schedules.

System Scope: The system will provide a centralized platform to manage port operations effectively, reducing manual errors, and improving overall efficiency.

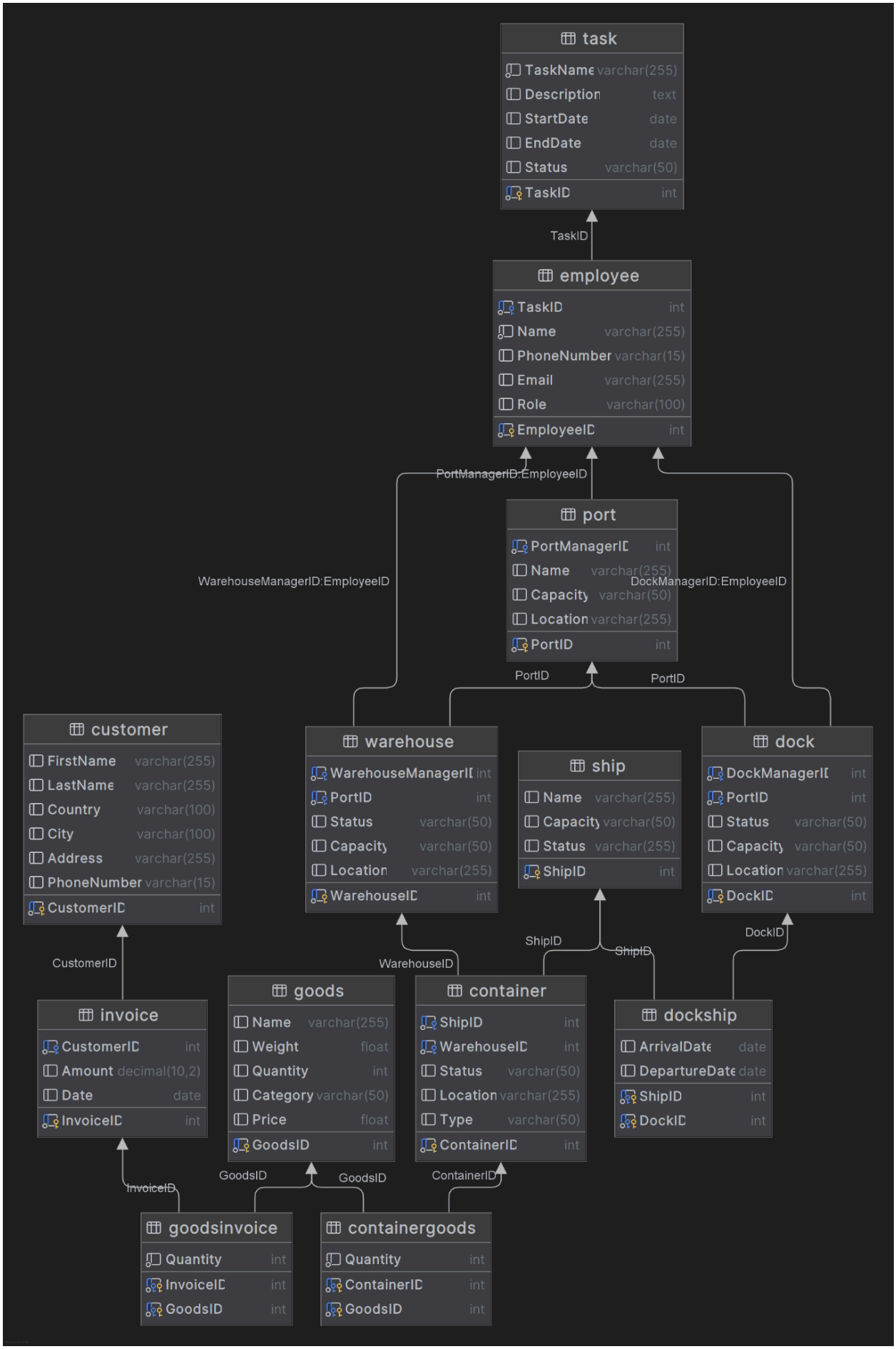
Contain 10 entities:

- Task(TaskID, TaskName, Description, StartDate, EndDate, Status)
- Employee(EmployeeID, TaskID, Name, PhoneNumber, Email, Role)
- Dock(DockID, DockManagerID, PortID, Status, Capacity, Location)
- Port(PortID, PortManagerID, Name, Capacity, Location)
- Warehouse(WarehouseID, WarehouseManagerID, PortID, Status, Capacity, Location)
- Ship(ShipID, Name, Capacity, Status)
- Container(ContainerID, ShipID, WarehouseID, Status, Location, Type)
- Goods(GoodsID, Name, Weight, Quantity, Category, Price)
- Invoice(InvoiceID, CustomerID, Amount, Date)
- Customer(CustomerID, FirstName, LastName, Country, City, Address, PhoneNumber).

ER Model For System:



Relationship Model (Convert ER To Relationship Model):


task
TaskName varchar(255)
Description text
StartDate date
EndDate date
Status varchar(50)
TaskID int
TaskID
employee
TaskID int
Name varchar(255)
PhoneNumber varchar(15)
Email varchar(255)
Role varchar(100)
EmployeeID int
PortManagerID:EmployeeID
WarehouseManagerID:EmployeeID
DockManagerID:EmployeeID
port
PortManagerID int
Name varchar(255)
Capacity varchar(50)
Location varchar(255)
PortID int
PortID
PortID
customer
FirstName varchar(255)
LastName varchar(255)
Country varchar(100)
City varchar(100)
Address varchar(255)
PhoneNumber varchar(15)
CustomerID int
warehouse
WarehouseManagerID int
PortID int
Status varchar(50)
Capacity varchar(50)
Location varchar(255)
WarehouseID int
ship
Name varchar(255)
Capacity varchar(50)
Status varchar(255)
ShipID int
dock
DockManagerID int
PortID int
Status varchar(50)
Capacity varchar(50)
Location varchar(255)
DockID int
CustomerID
WarehouseID
ShipID
ShipID
DockID
invoice
CustomerID int
Amount decimal(10,2)
Date date
InvoiceID int
goods
Name varchar(255)
Weight float
Quantity int
Category varchar(50)
Price float
GoodsID int
container
ShipID int
WarehouseID int
Status varchar(50)
Location varchar(255)
Type varchar(50)
ContainerID int
dockship
ArrivalDate date
DepartureDate date
ShipID int
DockID int
InvoiceID
GoodsID
GoodsID
ContainerID
goodsinvoice
Quantity int
InvoiceID int
GoodsID int
containergoods
Quantity int
ContainerID int
GoodsID int

- Task(TaskID, TaskName, Description, StartDate, EndDate, Status)
- Employee(EmployeeID, TaskID, Name, PhoneNumber, Email, Role)
- Dock(DockID, DockManagerID, PortID, Status, Capacity, Location)
- Port(PortID, PortManagerID, Name, Capacity, Location)
- Warehouse(WarehouseID, WarehouseManagerID, PortID, Status, Capacity, Location)
- Ship(ShipID, Name, Capacity)
- Container(ContainerID, ShipID, WarehouseID, Status, Location, Type)
- Goods(GoodsID, ContainerID, Name, Weight, Quantity, Category)
- Invoice(InvoiceID, CustomerID, Amount, Date)
- Customer(CustomerID, FirstName, LastName, Country, City, Address, PhoneNumber).
- GoodsInvoice(GoodsID, InvoiceID, Quantity)
- DockShip(DockID, ShipID, ArrivalDate, DepartureDate)
- ContainerGoods(ContainerID, GoodsID, Quantity)

Specification Data:

- Task

Data Element	Description	Composition Or Data Type	Values
TaskID	Unique identifier for a Task	Int	1, 2, 3, 4, 5,...
TaskName	Task Name	String	Mange Port,Mange Dock,...

Description	Detailed description of the task	String	Some things,...
StartDate	Start date	Date (YY-MM-DD)	2022-02-22,..
EndDate	End date	Date (YY-MM-DD)	2024-04-06,...
Status	Status of task	String	Completed, In Progress,....

- Container

Data Element	**Description**	**Composition Or Data Type**	**Values**
ContainerID	Unique identifier for a Container	Int	1, 2, 3, 4,...
ShipID	Each container belongs to a certain ship if it is being transported	Int	1, 4, 6,...
WarehouseID	Each container belongs to a terminal if it has not yet been shipped or has already been shipped	Int	1, 6, 7,...

Status	Status of container	String	On Hold, At Port,...
Location	Container storage place	String	ShipID 4941, ShipID 13,...
Type	Container category	String	Platform Container, Insulated Container,...

- Customer

Data Element	**Description**	**Composition Or Data Type**	**Values**
CustomerID	Unique identifier for a Customer	Int	1, 2, 3,...
FirstName	Customer's first name	String	Joseph, Anthony,...
LastName	Customer's last name	String	James, Green,...
Country	Country where customer lives	String	Canada, Brazil,...
City	City where customer lives	String	New George, West Joy,...
Address	Customer's address	String	Unit 5993 Box 9124 DPO AP 10883,...

PhoneNumber	Customer's phone number	String	(706)489-4272,...

- Dock

Data Element	Description	Composition Or Data Type	Values
DockID	Unique identifier for a Dock	Int	1, 2, 3,...
DockManagerID	Each dock will have a manager	Int	4, 6, 8,...
PortID	Each dock will belong to a port	Int	5, 7, 9,...
Status	Status of Dock	String	Inspection, Overcrowded,...
Capacity	Dock capacity	String	3232 million, 3113 million,...
Location	Location of the dock	String	Los Angeles, CA, New York and New Jersey, NY/NJ,...

- Employee

Data Element	Description	Composition	Values

		Or Data Type	
EmployeeID	Unique identifier for a Employee	Int	1, 2, 3, 4,...
TaskID	Each employee will have a task	Int	1, 3, 5,...
Name	Employee's name	String	James Lane, Julie Ayers,...
PhoneNumber	Employee's phone number	String	987-415-7718
Email	Employee's email	String	abs@gmail.com ,...
Role	Employee's role	String	Port Manager, Warehouse Manager, …

- Goods

Data Element	Description	Composition Or Data Type	Values
GoodsID	Unique identifier for a Goods	Int	1, 2, 3, 4,...
Name	Goods's name	String	Lamp,Water Filter,...
Weight	Goods's weight	Int	1234, 163,...

Quantity	Quantity of Goods	Int	102, 270,...
Category	Category of Goods	String	Food, Stationery,...
Price	Value of each Goods	Int	152, 546,...

- Invoice

Data Element	**Description**	**Composition Or Data Type**	**Values**
InvoiceID	Unique identifier for a Invoice	Int	1, 2, 3, 4,...
CustomerID	Each invoice will belong to a customer	Int	1, 2, 3,...
Amount	Total amount purchased by the customer	Float	432456.00, 345266.00,...
Date	Invoice date	Date (YY-MM-DD)	2023-09-21

- Port

Data Element	Description	Composition Or Data Type	Values
PortID	Unique identifier for a Port	Int	1, 2, 3,...
PortManagerID	Each port will have a manager	Int	1, 2, 3,...
Name	Port name	String	Port 1, Port 2,...
Capacity	Port capacity	String	31 million, 25 million,...
Location	Port location	String	Port of Los Angeles, CA,...

- Ship

Data Element	Description	Composition Or Data Type	Values
ShipID	Unique identifier for a Ship	Int	1, 2, 3,...
Name	Ship name	String	Ship 1, ship 2,...
Capacity	Ship capacity	String	13682 TEU, 11250 TEU,...
Status	Ship status	String	Full, Not Full

- Warehouse

Data Element	Description	Composition Or Data Type	Values
WarehouseID	Unique identifier for a Warehouse	Int	1, 2, 3, 4,...
WarehouseManagerID	Each warehouse will have a manager	Int	1, 2, 3,...
PortID	Each warehouse will belong to a port	Int	1, 2, 3,...
Status	Warehouse status	String	Closed, Expanding ,...
Capacity	Warehouse capacity	String	86726 million, 91522 million
Location	Warehouse location	String	Los Angeles, CA, New York and New Jersey, NY/NJ

- DockShip

Data Element	Description	Composition Or Data Type	Values
ShipID	Unique identifier for a	Int	1, 2, 3, 4,...

	Ship		
DockID	Unique identifier for a Dock	Int	Empty, Reserved ,...
ArrivalDate	Ship arrival date	Date (YY-MM-DD)	2022-11-02,...
DepartureDate	Ship departure date	Date (YY-MM-DD)	2024-06-03,...

- GoodsInvoice

Data Element	Description	Composition Or Data Type	Values
InvoiceID	Unique identifier for a Invoice	Int	1, 2, 3,...
GoodsID	Unique identifier for a GoodsID	Int	1, 2, 3,...
Quantity	Quantity of Goods for each Invoice	Int	234, 573,...

- ContainerGoods

Data Element	Description	Composition Or Data Type	Values

ContainerID	Unique identifier for a Container	Int	1, 2, 3,...
GoodsID	Unique identifier for a Goods	Int	1, 2, 3,...
Quantity	Quantity of Goods in each Container	Int	123, 4356,...

Relationship Model For System:

- Task - Employee:
  - Relationship type: (1 - N)
  - Relationship: One Task may be done by many Employees and many Employees can do one Task

- Employee - Dock:
  - Relationship type: (1 - 1)
  - Relationship: One Dock has one Manager and one Manager Manages one Dock

- Employee - Port:
  - Relationship type: (1 - 1)
  - Relationship: One Port has one Manger and one Manager manages one Port

- Employee - Warehouse:
  - Relationship type: (1 - 1)
  - Relationship: One Warehouse has one Manager and one Manager Manages one Warehouse

- Dock - Ship:
    - Relationship type: (N - N)
    - Relationship: One Dock contains many Ships and one Ship anchors in many Docks

- Ship - Container:
    - Relationship type: (1 - N)
    - Relationship type: One Ship contains many Containers and one Container may belong to one Ship

- Container - Goods:
    - Relationship: (N - N)
    - Relationship type: One Containers contains many Goods and one Item belongs to many Containers

- Goods - Invoice:
    - Relationship: (N - N)
    - Relationship type: One Item may appear in many Invoices and one Invoice can save the data of many Goods

- Customer - Invoice:
    - Relationship: (1 - N)
    - Relationship type: One customer has many Invoices and one invoice may belong to one Customer

- Port - Dock:
    - Relationship: (1 - N)
    - Relationship type: One Port contains many Dock and one Dock belong to one Port

- Port - Warehouse:
    - Relationship: (1 - N)
    - Relationship type: One Port contains many Warehouse and one Warehouse belong to one Port

CONVERT functional dependencies -> 3NF:

- Because all functional dependencies are converted to 3NF. So we don't need to do it.

CREATE DATABASE:

- Create entities table:
    + Create Task table:

```
-- Table Task
CREATE TABLE IF NOT EXISTS Task (
                        TaskID INT PRIMARY KEY NOT NULL ,
                        TaskName VARCHAR(255) NOT NULL,
                        Description TEXT,
                        StartDate DATE,
                        EndDate DATE,
                        Status VARCHAR(50)
);
```

    + Create Employee Table:

```
-- Table Employee
CREATE TABLE IF NOT EXISTS Employee (
                            EmployeeID INT PRIMARY KEY NOT NULL,
                            TaskID INT NOT NULL ,
                            Name VARCHAR(255) NOT NULL,
                            PhoneNumber VARCHAR(15),
                            Email VARCHAR(255),
                            Role VARCHAR(100)
);
```

+ Create Dock table:

```
-- Table Dock
CREATE TABLE IF NOT EXISTS  Dock (
                        DockID INT PRIMARY KEY NOT NULL,
                        DockManagerID INT NOT NULL,
                        PortID INT NOT NULL ,
                        Status VARCHAR(50),
                        Capacity INT,
                        Location VARCHAR(255),
                        UNIQUE (DockManagerID)
);
```

+ Create Port table:

```
-- Table Port
CREATE TABLE IF NOT EXISTS  Port (
                        PortID INT PRIMARY KEY NOT NULL,
                        PortManagerID INT NOT NULL,
                        Name VARCHAR(255),
                        Capacity INT,
                        Location VARCHAR(255),
                        UNIQUE (PortManagerID)
);
```

+ Create Warehouse table:

```
-- Table Warehouse
CREATE TABLE IF NOT EXISTS  Warehouse (
                            WarehouseID INT PRIMARY KEY NOT NULL,
                            WarehouseManagerID INT NOT NULL,
                            PortID INT NOT NULL,
                            Status VARCHAR(50),
                            Capacity INT,
                            Location VARCHAR(255),
                            UNIQUE (WarehouseManagerID)
);
```

+ Create Ship table:

```
CREATE TABLE IF NOT EXISTS  Ship (
                            ShipID INT PRIMARY KEY NOT NULL,
                            Name VARCHAR(255),
                            Capacity INT,
                            Status VARCHAR(255)
);
```

+ Create Container table:

```
-- Table Container
CREATE TABLE IF NOT EXISTS  Container (
                            ContainerID INT PRIMARY KEY NOT NULL,
                            ShipID INT NOT NULL,
                            WarehouseID INT NOT NULL,
                            Status VARCHAR(50),
                            Location VARCHAR(255),
                            Type VARCHAR(50)
);
```

+ Create Goods table:

```
CREATE TABLE IF NOT EXISTS  Container (
                            ContainerID INT PRIMARY KEY NOT NULL,
                            ShipID INT NOT NULL,
                            WarehouseID INT NOT NULL,
                            Status VARCHAR(50),
                            Location VARCHAR(255),
                            Type VARCHAR(50)
);
```

+ Create Invoice table:

```
-- Table Invoice
CREATE TABLE IF NOT EXISTS  Invoice (
                           InvoiceID INT PRIMARY KEY NOT NULL,
                           CustomerID INT NOT NULL,
                           Amount DECIMAL(10, 2),
                           Date DATE
);
```

+ Create Customer table:

```
-- Table Customer
CREATE TABLE IF NOT EXISTS  Customer (
                            CustomerID INT PRIMARY KEY NOT NULL,
                            FirstName VARCHAR(255),
                            LastName VARCHAR(255),
                            Country VARCHAR(100),
                            City VARCHAR(100),
                            Address VARCHAR(255),
                            PhoneNumber VARCHAR(15)
);
```

- Create relationship tables:

+ Relationship of Dock and Ship (Many - to - Many):

```
-- Table DockShip
CREATE TABLE IF NOT EXISTS  DockShip (
                              DockID INT NOT NULL,
                              ShipID INT NOT NULL,
                              ArrivalDate DATE,
                              DepartureDate DATE,
                              PRIMARY KEY (DockID, ShipID)
);
```

+ Relationship of Container and Goods (Many - to - Many):

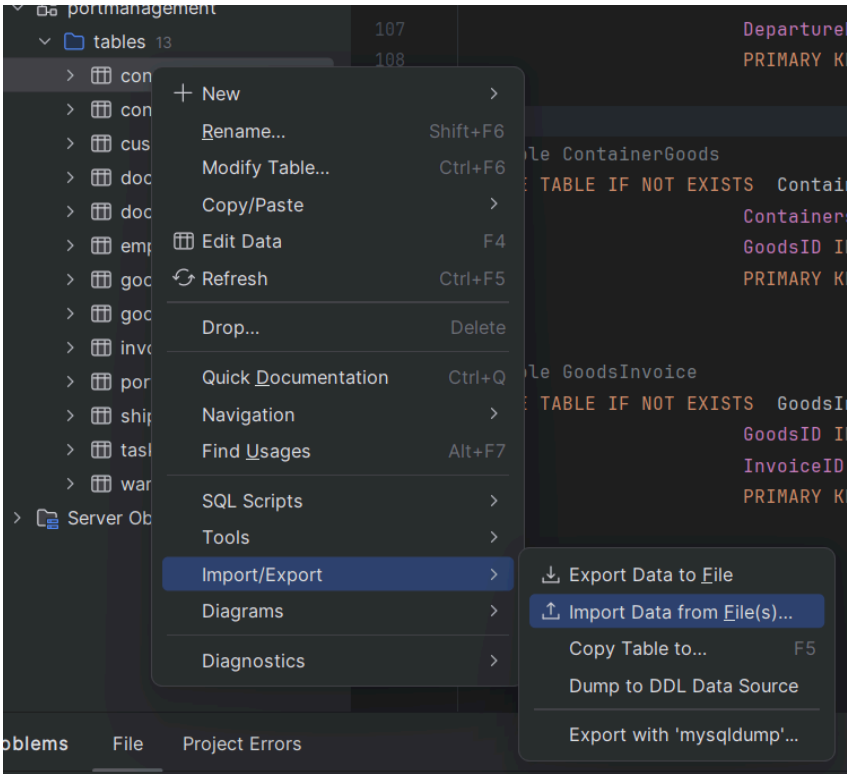
```
-- Table ContainerGoods
CREATE TABLE IF NOT EXISTS  ContainerGoods (
                            ContainersID INT NOT NULL,
                            GoodsID INT NOT NULL,
                            PRIMARY KEY (ContainersID, GoodsID)
);
```

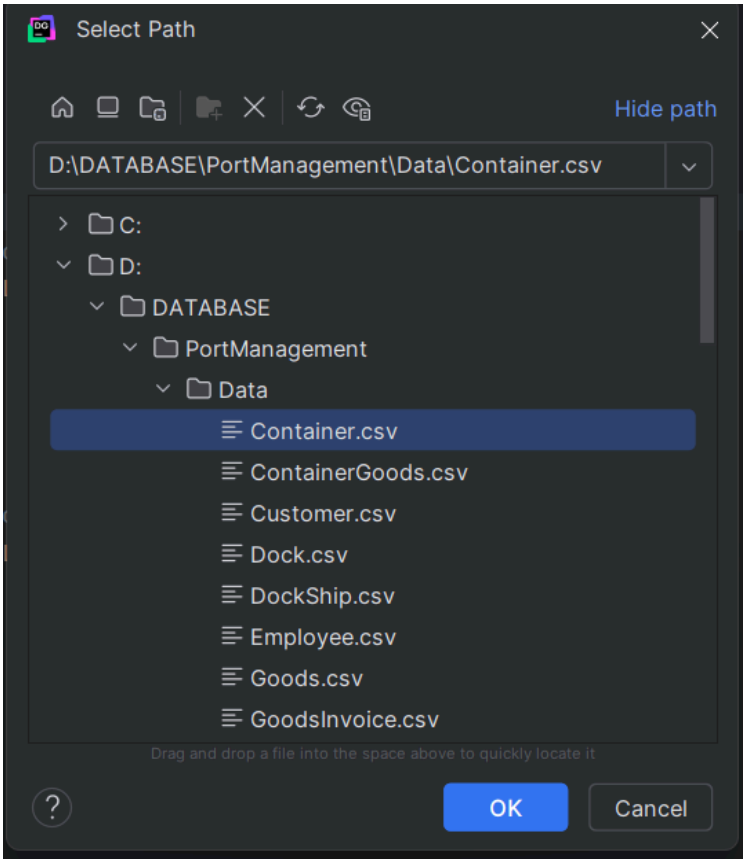
+ Relationship of Goods and Invoice (Many - to - Many):

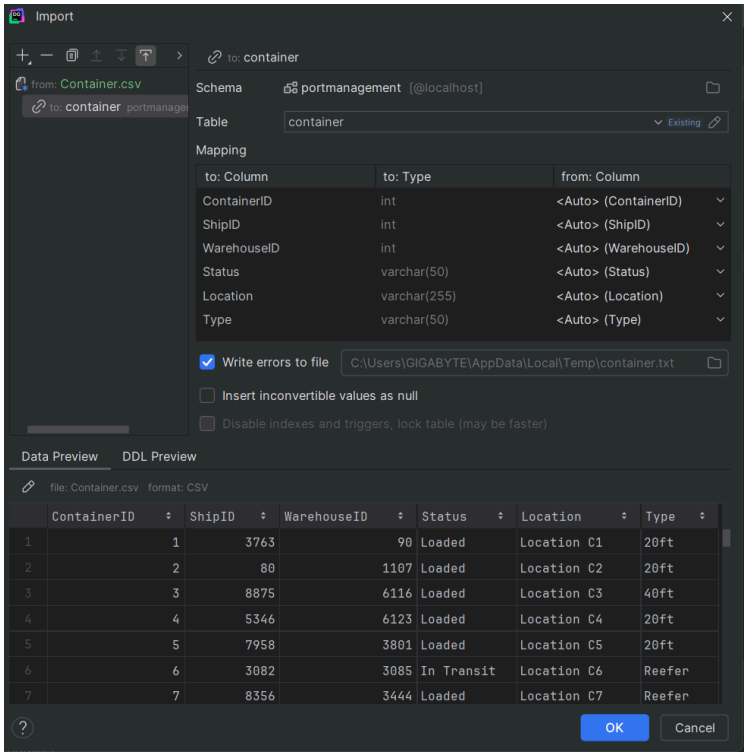
```
-- Table GoodsInvoice
CREATE TABLE IF NOT EXISTS  GoodsInvoice (
                            GoodsID INT NOT NULL,
                            InvoiceID INT NOT NULL,
                            PRIMARY KEY (GoodsID, InvoiceID)
);
```

IMPORT DATA:

Generate data in .CSV file, then import data:




Select Path
Hide path
D:\DATABASE\PortManagement\Data\Container.csv
C:
D:
DATABASE
PortManagement
Data
Container.csv
ContainerGoods.csv
Customer.csv
Dock.csv
DockShip.csv
Employee.csv
Goods.csv
GoodsInvoice.csv
Drag and drop a file into the space above to quickly locate it
OK
Cancel



QUERY:

- Query using inner join:
    - Find each port belonging to each dock.

```
SELECT port.Name, port.Capacity as PortCapacity,
       port.PortManagerID, dock.DockID,
       dock.Capacity as DockCapacity
FROM port
INNER JOIN dock
    ->1..n: ON port.PortID = dock.PortID;
```

- Find the weight of the current container.

```
SELECT con.ContainerID,
       SUM(Quantity * Weight) AS WeightContainer
FROM container con
INNER JOIN goods g
ON g.ContainerID = con.ContainerID
GROUP BY g.ContainerID;
```

- Query using outer join:
  - Display each task for each employee.

```
SELECT employee.Name, task.TaskName,
       task.Description, employee.Role
FROM task
LEFT JOIN employee
 1<->0..n: ON task.TaskID = employee.TaskID
UNION
SELECT  Name employee.Name,  TaskName task.TaskName,
        Description task.Description,  Role employee.Role
FROM task
RIGHT JOIN employee
 1<->1..n: ON task.TaskID = employee.TaskID;
```

- Display the number of containers in each ship.

```
SELECT si.ShipID, si.Name,
       COUNT(con.ContainerID) AS NumberContainer
FROM ship si
LEFT JOIN container con
 1<->0..n: ON si.ShipID = con.ShipID
GROUP BY si.ShipID;
```

- Using subquery in where:
  - Display total amount spent by each customer.

```
SELECT CONCAT(cus.FirstName, ' ', cus.LastName) AS FullName,
       (SELECT SUM(Amount)
        FROM invoice iv
        WHERE iv.CustomerID = cus.CustomerID) as Total
FROM customer cus
WHERE CustomerID IN (SELECT CustomerID FROM invoice)
GROUP BY CustomerID;
```

  - Display the total amount for each container.

```
SELECT g.ContainerID,
       (Quantity * Price) AS TotalAmount
FROM Goods g
WHERE g.ContainerID IN (SELECT ContainerID
                        FROM container)
```

- Using subquery in from:
  - Display container which has an amount over 5000$.

```
SELECT congo.TotalContainer
FROM (SELECT SUM(Price * Quantity) AS TotalContainer
      FROM Goods g
      INNER JOIN Container con
      ON g.ContainerID = con.ContainerID
      GROUP BY g.ContainerID
      ) congo
WHERE TotalContainer > 50;
```

- Calculate the total value (revenue) of each item in 2020.

```
SELECT goodsDate.ID, goodsDate.Name,
       MAX(goodsDate.totalAmount) AS MaxTotalAmount
FROM (
    SELECT g.GoodsID AS ID, g.Name AS Name,
           SUM(Quantity * Price) AS totalAmount
    FROM goods g
    INNER JOIN containergoods cong
        1<->1..n: ON g.GoodsID = cong.GoodsID
    INNER JOIN container con
        1..n<->1: ON cong.ContainersID = con.ContainerID
    INNER JOIN ship si
        1..n<->1: ON con.ShipID = si.ShipID
    INNER JOIN dockship dosi
        1<->1..n: ON si.ShipID = dosi.ShipID
    WHERE YEAR(ArrivalDate) = 2020
       OR YEAR(DepartureDate) = 2020
    GROUP BY g.GoodsID, g.Name
) AS goodsDate
GROUP BY goodsDate.ID, goodsDate.Name;
```

- Query using group by and aggregate functions:

- Count how many docks the port has.

```
SELECT PortID, Name,
       (SELECT COUNT(DockID)
        FROM dock
        WHERE dock.PortID = port.PortID) as TotalDock
FROM port
WHERE PortID IN (SELECT PortID FROM dock)
GROUP BY PortID
```

- Display the number of invoices for each customer.

```
SELECT cus.CustomerID,
       CONCAT(cus.FirstName, ' ', cus.LastName),
       COUNT(iv.InvoiceID) AS totalInvoice
FROM customer cus
INNER JOIN invoice iv
 1<->1..n: ON cus.CustomerID = iv.CustomerID
GROUP BY cus.CustomerID;
```

Transaction:

```sql
-- TRANSACTION for customer
START TRANSACTION ;

DELETE FROM customer
WHERE Country = 'USA' OR Country = 'Germany';

ROLLBACK ;
COMMIT ;
```

Trigger:

When you insert the new goods for a customer, the update_invoice trigger will be update automatic customer's invoice.

```
create definer = root@localhost trigger update_invoice
    after insert
    on goods
    for each row
BEGIN
    UPDATE invoice
    SET amount = amount + (
        SELECT NEW.Price * gin.Quantity
        FROM goodsinvoice gin
        WHERE gin.GoodsID = NEW.GoodsID
        LIMIT 1
    )
    WHERE InvoiceID IN (
        SELECT InvoiceID
        FROM goodsinvoice
        WHERE GoodsID = NEW.GoodsID
    );
END;
```

Update amount when event occurs:

```
DELIMITER //

CREATE TRIGGER after_update_invoice_amount
    AFTER UPDATE ON invoice
    FOR EACH ROW
    BEGIN
        IF NEW.Amount = OLD.Amount THEN
            UPDATE invoice
                SET Amount = OLD.Amount * 1.1;
        END IF ;
    END //

DELIMITER ;

-- trigger for TaskID - Task
DELIMITER //
```

Change country when event occurs:

```
-- trigger for country - customer
DELIMITER //

CREATE TRIGGER after_delete_country_customer
    BEFORE DELETE ON customer
    FOR EACH ROW
    BEGIN
        IF OLD.Country = 'USA' THEN
            UPDATE customer
                SET Country = 'USK'
                WHERE CustomerID = OLD.CustomerID;
        END IF;
    END //

DELIMITER ;
```

Store Procedure:

When the task has a completed status, it will automatically delete the task.

```
-- procedure for task
DELIMITER //

CREATE PROCEDURE delete_task()
BEGIN
    DELETE FROM Task
        WHERE Status = 'Completed';
END //

DELIMITER ;
```

Print all customer information:

```
DELIMITER //

CREATE PROCEDURE get_customer()
BEGIN
    SELECT * FROM customer;
END //

DELIMITER ;

-- procedure for container
DELIMITER //
```

Change container information when event occurs:

```
DELIMITER //

CREATE PROCEDURE change_container(
    IN NewStatus VARCHAR(50),
    IN NewLocation VARCHAR(255),
    IN NewType VARCHAR(50)
)
BEGIN
    UPDATE container
        SET Status = NewStatus, Location = NewLocation, Type = NewType
        WHERE ShipID = '80';
END //

DELIMITER ;
```