CS4380/7380 Database Management Systems –I
Final Project Planning Report (due 3/6/2015 by midnight)
Worth 5% of your final project

Project title: Warehouse management system

Team members

Last name	Fist name	E-mail	Major
Du	Yudu	ydw95@mail.missouri.edu	CS
Xu	Chunhui	cx9p9@mail.missouri.edu	Bioinformatics
Du	Ming	mdk86@mail.missouri.edu	CS
Xu	Yihan	yx6h3@mail.missouri.edu	CS

Content needed in your report (follow the order)

1. Data collection and client information.
2. E-R Diagram (any format)
3. Create tables using DDL in your group accounts (copy and paste your SQL statements for table creations). All members in the team should participate in creating tables.
4. List at least 10 ***useful*** queries in English sentences, as well as in relational algebra and SQL. Make sure they are different types and ***really useful***.
5. Contribution by each member so far. (Members never showed up for meeting or difficult to work with and voted by the majority will be removed from the group and placed in a "turkey farm.")
6. Workload plan for each team member. (I expect all of you work on database planning, design, implementation, and report writing.)
7. Weekly schedule

(8 pages max including this cover page.)

Introduction

warehouse management is a key part of the supply chain and primarily aims to control the movement and storage of materials within a warehouse and process the associated transactions, including shipping, receiving, and picking. Basically, a warehouse management database system could help people monitor the progress of products through the warehouse, also, it could provide a set of computerized procedures for management of warehouse inventory, space, equipment and people with the goal of minimizing cost and fulfillment times.

Client Information

As for China, even though there are a lots of smaller factories have capacity to accept the huge orders of light industry products, ineffective manual management caused in poor profits. Based on this, some smaller factories/companies are trying to find some advanced management method, and the systematic warehouse database is a really useful tool for them.

Our target client is a smaller plastics company, Ningbo Wenxiang Plastic Technology Company Which is located in Ningbo, China. This company specialized in processing plastic lotion pump. For more details: http://nbwxsy.1688.com

Data Collection

We will obtain the original data from the company directly. The original data should be in Excel format, which includes several tables: The information of suppliers, raw materials; the work schedule of warehouse manager; warehouse warrant; sales order; costumer information,etc. In database design part, we will provide more details for the data information and organization of data. But we have to mention that, for some data involved trade secret, such as the factory gate price and sales, it should be historical data not the data up to date.

E-R Diagram

In this warehouse storage scenario, we defined 6 basic entities: supplier, item, stocks, warehouse, customer, and staff. Figure 1 is a simply E-R Diagram to show the relation between these basic entities.
In addition, Figure 2 shows the details of the current version of our database.

SQL Statements

SQL statements for creating tables and triggers are attached at end of the proposal.

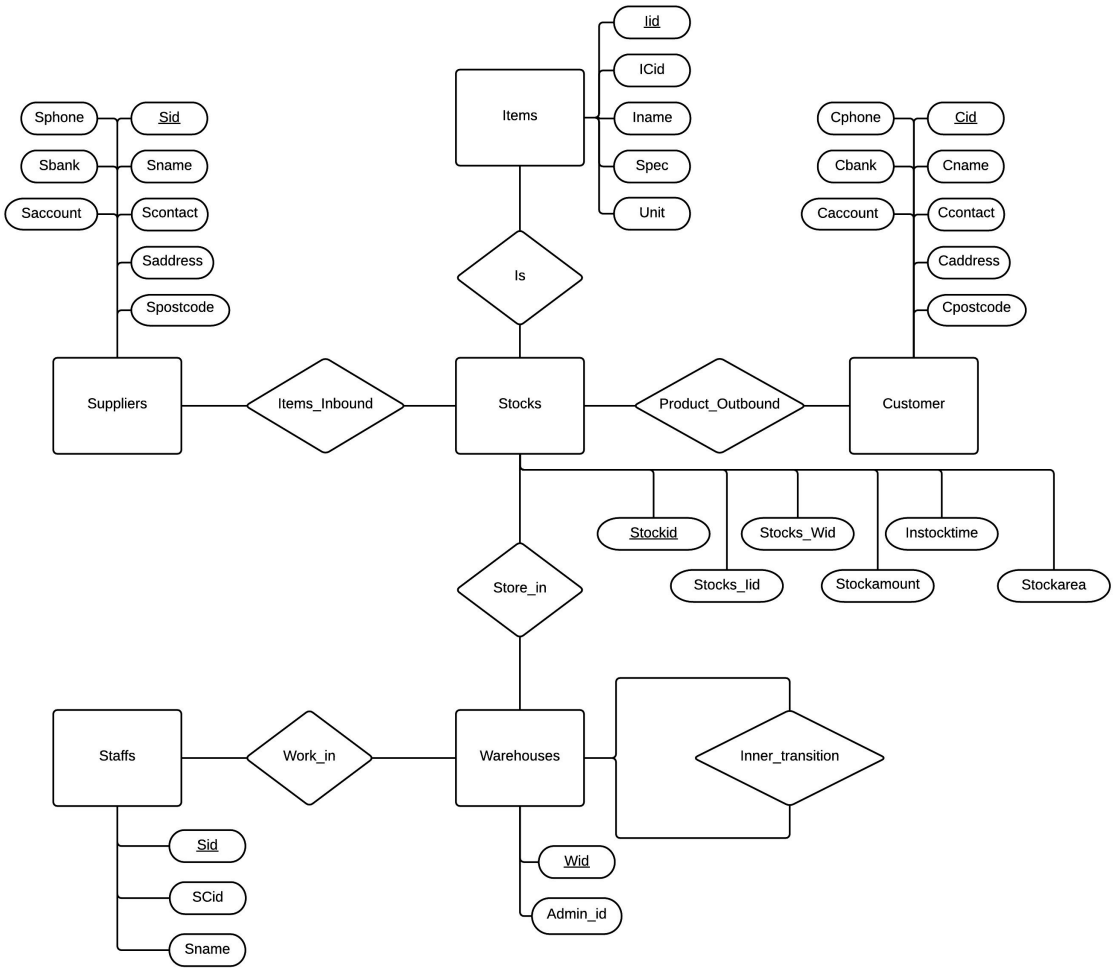

Figure 1

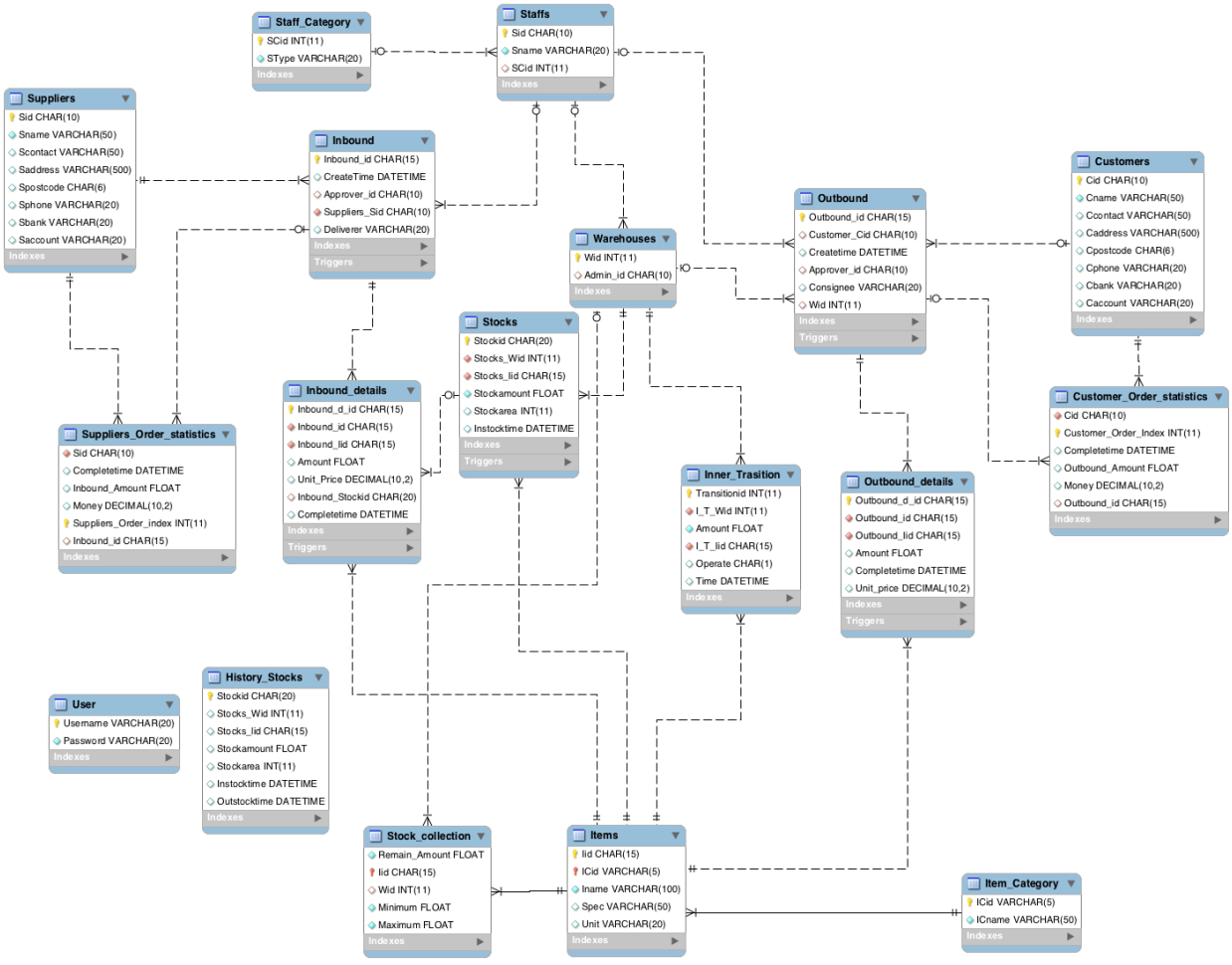

Figure 2

10 queries

1. Check how many products are left in inventory according to its product id

   $(\sigma_{<iid>} Items) \infty Stocks$

   ```
   SELECT Remain_Amount
   FROM Stock_collection
   WHERE Iid= <id>;
   ```

2. List the warehouse entry details in a certain period

   $(\sigma_{Start<CreateTime,Eng>CreateTime} Inbound) \infty Inbound\_details \infty Items$

   ```
   SELECT *
   FROM Inbound ib,Inbound_details ibd,Items it
   WHERE ib.CreateTime>= <start>
         AND ib.CreateTime<= <end>
         AND ib.Iid = ibd.Inbound_Iid;
   ```

3. Check the identity information of the warehouse-keeper who created the inbound record of a certain batch of goods.

   $\pi_{Sid,Sname} Staffs \infty (\sigma_{Approver\_id=<id>} Inbound)$

   ```
   SELECT sid,sname
   FROM Staffs
   WHERE sid IN
    (
   SELECT Approver_id
   FROM Inbound ib,Inbound_details ibd
   WHERE ib.Inbound_id=ibd.Inbound_id
   AND ibd.Inbound_Iid=<item id>
    );
   ```

4. List records of the material checking in events created by a certain warehouse-keeper in a certain period.

   $(\sigma_{Sid=<id>} Staff) \infty (\sigma_{<st><CreateTime<<end>} Inbound)$

   ```
   SELECT *
   FROM Inbound ib,Staffs stf
   WHERE ib.CreateTime>= <start>
         AND ib.CreateTime<= <end>
         AND stf.sid=ib.sid;
   ```

5. Check the amount and id of the material provided by a certain supplier.

   $(\sigma_{Sid=<Sid>} Inbound) \infty Inbound\_details \infty Items$

   ```
   SELECT COUNT(*)
   FROM Inbound ib, Inbound_details ibd,Items it
   WHERE ib.Inbound_id=ibd.Inbound_id
         AND ibd.Inbound_Iid=it.Iid
         AND ib.sid=<sid>
   GROUP BY ibd.Inbound_Iid;
   ```

6. List the amount and id of products purchased by a certain client in a certain period

   $(\sigma_{CreateTime} Inbound) \infty Inbound\_details \infty Customers \infty Items$

   ```
   SELECT it.name, it.Iid,COUNT(*)
   FROM Customers C, Outbound ob,Outbound_details obd,Items it
   WHERE C.cid=ob.cid
         AND obd.Outbound_id=ob.Outbound_id
         AND obd.Outbound_Iid=it.Iid
         AND ob.createTime<= <start>
   ```

```
        AND ob.CreateTime>= <end>
    GROUP BY it.lid;
```

7. List all clients that have purchased a certain product according to its product id.

$$(\sigma_{Iid=<id>} Items) \infty Outbound\_details \infty Outbound \infty Customers$$

```
SELECT distinct *
FROM Customers
WHERE cid in
(
        SELECT cid FROM Outbound ob, Outbound_details obd
        WHERE ob.Outbound_id = obd.Outbound_id
                AND obd.Outbound_Iid IN
                (
                        SELECT cid FROM item
                        WHERE cid= <cid>
                )
);
```

8. Check the location of a batch of material in the warehouse according to the warehouse entry number

$$\pi_{StockArea}(\sigma_{Inbound\_id=id} Inbound\_details) \infty Stocks$$

```
SELECT Stockarea
FROM Inbound_details, Stocks
WHERE Inbound_id= <id>
        AND Inbound_Stockid = Stockid;
```

9. Check the amount of item in a certain area of warehouse

```
Relational Algebra: N/A
SELECT Stockarea, SUM(Stockamount)
FROM Stocks
WHERE Stockarea = <stock_area>;
```

10. List the id of material in inventory whose current amount is lower than safe value.

```
Relational Algebra: N/A
SELECT SC.Iid, COUNT (*) AS CT
FROM Stocks_collection SC
GROUP BY SC.Iid
HAVING CT< SC.Minimum;
```

Contributes by each member

Ming Du:
- Database planning
- Decided development environment and languages to be used
- Discussed database structure and queries

Yihan Xu:
- Database planning
- Created tables in database
- Discussed database structure and queries
- Project schedule

Yudu Du:
- Database planning
- ER Diagram and model
- Created and revised tables and relationship in database
- Created triggers in database

Chunhui Xu:
- Database planning
- Acquired detailed information about client
- Discussed database structure and queries

Workload Plan

Ming Du:
- Mainly responsible for database implementation. Also participate in the entire progress of the development

Yudu Du:
- Mainly responsible for database planning and designing. Also partcipate in the entire progress of the development.

Yihan Xu:
- Mainly responsible for project planning and managing. Also participate in the entire progress of the development.

Chunhui Xu:
- Mainly responsible for report writing. Also participate in the entire progress of the development.

Every team member has his main responsibility and is supposed to dispatch his small tasks and complete other members' small tasks to complete the whole project.

Weekly Schedule

Mar 2 - Mar 8:
- Database Structure
- Data Collection Method and Client information
- E-R Diagram
- Pre-Preposal report

Mar 9 - Mar 15:
- Database Planning
- Database Designing

Mar 16 - Mar 22:
- Software Planning Contd.
- Software Designing Contd.
- Software Prototying

Mar 23 - Mar 29:
- Database Implementation
- Software Implementation

Mar 30 - Apr 5:
- Database Implementatioin Contd.
- Software Implementation Contd.

Apr 6 - Apr 12:
- Database Implementation Contd.
- Software Implementation Contd.

Apr 13 - Apr 19:
- Database Implementation Contd.
- Software Implementation Contd.

Apr 20 - Apr 26:
- System Testing
- System Refinement

Apr 27 - May 3:
- System Refinement
- Report Writing
- Presentation Preparing

May 4 - May 10:
- Report Refinement
- Presentation Refinement

```
DROP DATABASE IF EXISTS final_project;
CREATE DATABASE IF NOT EXISTS final_project;
USE final_project;

-- create tables

DROP TABLE IF EXISTS Customers ;
DROP TABLE IF EXISTS Staff_Category ;
DROP TABLE IF EXISTS Staffs ;
DROP TABLE IF EXISTS Warehouses;
DROP TABLE IF EXISTS Outbound ;
DROP TABLE IF EXISTS Customer_Order_statistics ;
DROP TABLE IF EXISTS Item_Category ;
DROP TABLE IF EXISTS Items ;
DROP TABLE IF EXISTS Outbound_details ;
DROP TABLE IF EXISTS Stock_collection ;
DROP TABLE IF EXISTS Stocks ;
DROP TABLE IF EXISTS Suppliers ;
DROP TABLE IF EXISTS Inbound ;
DROP TABLE IF EXISTS Suppliers_Order_statistics ;
DROP TABLE IF EXISTS User ;
DROP TABLE IF EXISTS History_Stocks ;
DROP TABLE IF EXISTS Inner_Trasition ;

-- Table Customers

CREATE TABLE IF NOT EXISTS Customers (
  Cid CHAR(10) NOT NULL,
  Cname VARCHAR(50) NOT NULL,
  Ccontact VARCHAR(50) NULL DEFAULT NULL,
  Caddress VARCHAR(500) NULL DEFAULT NULL,
  Cpostcode CHAR(6) NULL DEFAULT NULL,
  Cphone VARCHAR(20) NULL DEFAULT NULL,
  Cbank VARCHAR(20) NULL DEFAULT NULL,
  Caccount VARCHAR(20) NULL DEFAULT NULL,
  PRIMARY KEY (Cid))
ENGINE = InnoDB;

-- Table Staff_Category

CREATE TABLE IF NOT EXISTS Staff_Category (
  SCid INT NOT NULL,
  SType VARCHAR(20) NOT NULL,
  PRIMARY KEY (SCid))
ENGINE = InnoDB;

-- Table Staffs

CREATE TABLE IF NOT EXISTS Staffs (
  Sid CHAR(10) NOT NULL,
  Sname VARCHAR(20) NOT NULL,
  SCid INT NULL DEFAULT NULL,
  PRIMARY KEY (Sid),
  INDEX Scid_idx (SCid ASC),
  CONSTRAINT Staffs_SCid
    FOREIGN KEY (SCid)
    REFERENCES Staff_Category (SCid)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Warehouses

CREATE TABLE IF NOT EXISTS Warehouses (
  Wid INT NOT NULL,
  Admin_id CHAR(10) NULL DEFAULT NULL,
  PRIMARY KEY (Wid),
  INDEX Admin_idx (Admin_id ASC),
  CONSTRAINT Warehouses_Admin_id
    FOREIGN KEY (Admin_id)
    REFERENCES Staffs (Sid)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Outbound

CREATE TABLE IF NOT EXISTS Outbound (
  Outbound_id CHAR(15) NOT NULL,
  Customer_Cid CHAR(10) NULL DEFAULT NULL,
  Createtime DATETIME NULL DEFAULT NULL,
  Approver_id CHAR(10) NULL DEFAULT NULL,
  Consignee VARCHAR(20) NULL DEFAULT NULL,
  Wid INT NULL,
  PRIMARY KEY (Outbound_id),
  INDEX Cid_idx (Customer_Cid ASC),
  INDEX Approver_idx (Approver_id ASC),
  INDEX Outbound_Warehouse_Wid_idx (Wid ASC),
  CONSTRAINT Outbound_Cid
    FOREIGN KEY (Customer_Cid)
    REFERENCES Customers (Cid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Outbound_Approver_id
    FOREIGN KEY (Approver_id)
    REFERENCES Staffs (Sid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Outbound_Wid
    FOREIGN KEY (Wid)
    REFERENCES Warehouses (Wid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Customer_Order_statistics

CREATE TABLE IF NOT EXISTS
Customer_Order_statistics (
  Cid CHAR(10) NOT NULL,
  Customer_Order_Index INT NOT NULL
AUTO_INCREMENT,
  Completetime DATETIME NULL,
  Outbound_Amount FLOAT NULL,
  Money DECIMAL(10,2) NULL,
  Outbound_id CHAR(15) NULL,
  PRIMARY KEY (Customer_Order_Index),
  INDEX Customers1_Cid (Cid ASC),
  INDEX Outbound_id (Outbound_id ASC),
  CONSTRAINT Cid
    FOREIGN KEY (Cid)
    REFERENCES Customers (Cid)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT Customer_Outbound_id
    FOREIGN KEY (Outbound_id)
    REFERENCES Outbound (Outbound_id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- Table Item Category

CREATE TABLE IF NOT EXISTS Item_Category (
  ICid VARCHAR(5) NOT NULL,
  ICname VARCHAR(50) NOT NULL,
  PRIMARY KEY (ICid))
ENGINE = InnoDB;

-- Table Items

CREATE TABLE IF NOT EXISTS Items (
  Iid CHAR(15) NOT NULL,
  ICid VARCHAR(5) NOT NULL,
  Iname VARCHAR(100) NOT NULL,
  Spec VARCHAR(50) NULL DEFAULT NULL,
  Unit VARCHAR(20) NULL DEFAULT NULL,
  PRIMARY KEY (Iid, ICid),
  INDEX Pcid_idx (ICid ASC),
  CONSTRAINT Items_ICid
    FOREIGN KEY (ICid)
    REFERENCES Item_Category (ICid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
-- Table Outbound_details

CREATE TABLE IF NOT EXISTS Outbound_details (
  Outbound_d_id CHAR(15) NOT NULL,
  Outbound_id CHAR(15) NOT NULL,
  Outbound_lid CHAR(15) NOT NULL,
  Amount FLOAT NULL DEFAULT NULL,
  Completetime DATETIME NULL DEFAULT NULL,
  Unit_price DECIMAL(10,2) NULL,
  PRIMARY KEY (Outbound_d_id),
  INDEX Outbound_id_idx (Outbound_id ASC),
  INDEX Outbound_lid_idx (Outbound_lid ASC),
  CONSTRAINT Outbound_id
    FOREIGN KEY (Outbound_id)
    REFERENCES Outbound (Outbound_id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Outbound_lid
    FOREIGN KEY (Outbound_lid)
    REFERENCES Items (Iid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Stock_collection

CREATE TABLE IF NOT EXISTS Stock_collection (
  Remain_Amount FLOAT NOT NULL,
  Iid CHAR(15) NOT NULL,
  Wid INT NULL,
  PRIMARY KEY (Iid),
  Minimum FLOAT NOT NULL,
  Maximum FLOAT NOT NULL,
  INDEX Stock_collection_Items1_idx (Iid ASC),
  INDEX Stock_collection_Wid_idx (Wid ASC),
  CONSTRAINT Stock_collection_Iid
    FOREIGN KEY (Iid)
    REFERENCES Items (Iid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Stock_collection_Wid
    FOREIGN KEY (Wid)
    REFERENCES Warehouses (Wid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Stocks

CREATE TABLE IF NOT EXISTS Stocks (
  Stockid CHAR(20) NOT NULL,
  Stocks_Wid INT NOT NULL,
  Stocks_Iid CHAR(15) NOT NULL,
  Stockamount FLOAT NOT NULL,
  Stockarea INT NULL DEFAULT NULL,
  Instocktime DATETIME NULL DEFAULT NULL,
  PRIMARY KEY (Stockid),
  INDEX Wid_idx (Stocks_Wid ASC),
  INDEX Pid_idx (Stocks_Iid ASC),
  CONSTRAINT Stocks_Wid
    FOREIGN KEY (Stocks_Wid)
    REFERENCES Warehouses (Wid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Stocks_Iid
    FOREIGN KEY (Stocks_Iid)
    REFERENCES Items (Iid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Suppliers

CREATE TABLE IF NOT EXISTS Suppliers (
  Sid CHAR(10) NOT NULL,
  Sname VARCHAR(50) NOT NULL,
  Scontact VARCHAR(50) NULL DEFAULT NULL,
  Saddress VARCHAR(500) NULL DEFAULT NULL,
  Spostcode CHAR(6) NULL DEFAULT NULL,
  Sphone VARCHAR(20) NULL DEFAULT NULL,
  Sbank VARCHAR(20) NULL DEFAULT NULL,
  Saccount VARCHAR(20) NULL DEFAULT NULL,
  PRIMARY KEY (Sid))
ENGINE = InnoDB;

-- Table Inbound

CREATE TABLE IF NOT EXISTS Inbound (
  Inbound_id CHAR(15) NOT NULL,
  CreateTime DATETIME NULL DEFAULT NULL,
  Approver_id CHAR(10) NULL DEFAULT NULL,
  Suppliers_Sid CHAR(10) NOT NULL,
  Deliverer VARCHAR(20) NULL,
  PRIMARY KEY (Inbound_id),
  INDEX Approver_idx (Approver_id ASC),
  INDEX Warehouse_entry_Suppliers1_idx (Suppliers_Sid
ASC),
  CONSTRAINT Inbound_Approver_id
    FOREIGN KEY (Approver_id)
    REFERENCES Staffs (Sid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Inbound_Suppliers_Sid
    FOREIGN KEY (Suppliers_Sid)
    REFERENCES Suppliers (Sid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table Suppliers_Order_statistics

CREATE TABLE IF NOT EXISTS
Suppliers_Order_statistics (
  Sid CHAR(10) NOT NULL,
  Completetime DATETIME NULL DEFAULT NULL,
  Inbound_Amount FLOAT NULL,
  Money DECIMAL(10,2) NULL,
  Suppliers_Order_index INT NOT NULL
AUTO_INCREMENT,
  Inbound_id CHAR(15) NULL,
  PRIMARY KEY (Suppliers_Order_index),
  INDEX Sid (Sid ASC),
  INDEX fk_Suppliers_Order_statistics_Inbound1_idx
(Inbound_id ASC),
  CONSTRAINT Inblound_id
    FOREIGN KEY (Inbound_id)
    REFERENCES Inbound (Inbound_id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Sid
    FOREIGN KEY (Sid)
    REFERENCES Suppliers (Sid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table User

CREATE TABLE IF NOT EXISTS User (
  Username VARCHAR(20) NOT NULL,
  Password VARCHAR(20) NOT NULL,
  PRIMARY KEY (Username))
ENGINE = InnoDB;

-- Table Inbound_details

CREATE TABLE IF NOT EXISTS Inbound_details (
  Inbound_d_id CHAR(15) NOT NULL,
  Inbound_id CHAR(15) NOT NULL,
  Inbound_Iid CHAR(15) NOT NULL,
  Amount FLOAT NULL DEFAULT NULL,
  Unit_Price DECIMAL(10,2) NULL DEFAULT NULL,
  Inbound_Stockid CHAR(20) NULL,
  Completetime DATETIME NULL,
  PRIMARY KEY (Inbound_d_id),
  INDEX Inbound_idx (Inbound_id ASC),
```

```
  INDEX Iid_idx (Inbound_Iid ASC),
  INDEX Warehouse_entry_details_Stocks1_idx
(Inbound_Stockid ASC),
  CONSTRAINT Inbound_id
    FOREIGN KEY (Inbound_id)
    REFERENCES Inbound (Inbound_id)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Inbound_Iid
    FOREIGN KEY (Inbound_Iid)
    REFERENCES Items (Iid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT Inbound_d_stockid
    FOREIGN KEY (Inbound_Stockid)
    REFERENCES Stocks (Stockid)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- Table History_Stocks

CREATE TABLE IF NOT EXISTS History_Stocks (
  Stockid CHAR(20) NOT NULL ,
  Stocks_Wid INT NULL,
  Stocks_Iid CHAR(15) NULL,
  Stockamount FLOAT NULL,
  Stockarea INT NULL,
  Instocktime DATETIME NULL,
  Outstocktime DATETIME NULL,
  PRIMARY KEY (Stockid))
ENGINE = InnoDB;

-- Table Inner_Trasition

CREATE TABLE IF NOT EXISTS Inner_Trasition (
  Transitionid INT NOT NULL,
  I_T_Wid INT NOT NULL,
  Amount FLOAT NOT NULL,
  I_T_Iid CHAR(15) NOT NULL,
  Operate CHAR(1) NULL,
  Time DATETIME NULL,
  PRIMARY KEY (Transitionid),
  INDEX fk_Inner_trasition_Products1_idx (I_T_Iid ASC),
  INDEX I_T_Wid_idx (I_T_Wid ASC),
  CONSTRAINT I_T_Wid
    FOREIGN KEY (I_T_Wid)
    REFERENCES Warehouses (Wid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT I_T_Iid
    FOREIGN KEY (I_T_Iid)
    REFERENCES Items (Iid)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

-- create triggers

DROP TRIGGER IF EXISTS Outbound_AFTER_INSERT;
DROP TRIGGER IF EXISTS
Outbound_details_AFTER_INSERT;
DROP TRIGGER IF EXISTS Stocks_AFTER_INSERT;
DROP TRIGGER IF EXISTS Stocks_BEFORE_UPDATE;
DROP TRIGGER IF EXISTS Stocks_AFTER_UPDATE;
DROP TRIGGER IF EXISTS Stocks_BEFORE_DELETE;
DROP TRIGGER IF EXISTS Inbound_AFTER_INSERT;
DROP TRIGGER IF EXISTS
Inbound_details_AFTER_INSERT;

DELIMITER $$
CREATE TRIGGER Outbound_AFTER_INSERT AFTER
INSERT ON Outbound FOR EACH ROW
BEGIN
        INSERT INTO Customers_Order_statistics SET
Cid=NEW.Customer_Cid,Outbound_id=NEW.Outbound_id
;
END;

CREATE TRIGGER Outbound_details_AFTER_INSERT
AFTER INSERT ON Outbound_details FOR EACH ROW
BEGIN
        UPDATE Customer_Order_statistics SET
Completetime=NEW.Completetime,
Outbound_Amount=NEW.Amount,
Money=cast(NEW.Amount AS
DECIMAL(10,2))*NEW.Unit_price WHERE
Outbound_id=NEW.Outbound_id;
END;

CREATE TRIGGER Stocks_AFTER_INSERT AFTER
INSERT ON Stocks FOR EACH ROW
BEGIN
 INSERT INTO Stock_collection (Iid, Remain_Amount,Wid)
VALUES (NEW.Stocks_Iid, NEW.Stockamount,
NEW.Stocks_Wid) ON DUPLICATE KEY UPDATE
Remain_Amount=Remain_Amount+NEW.Stockamount;
END;

CREATE TRIGGER Stocks_BEFORE_UPDATE BEFORE
UPDATE ON Stocks FOR EACH ROW
BEGIN
        UPDATE Stock_collection set
Remain_Amount=Remain_Amount-OLD.Stockamount
WHERE Iid=OLD.Stocks_Iid;
END;

CREATE TRIGGER Stocks_AFTER_UPDATE AFTER
UPDATE ON Stocks FOR EACH ROW
BEGIN
        UPDATE Stock_collection set
Remain_Amount=Remain_Amount+NEW.Stockamount
WHERE Iid=OLD.Stocks_Iid;
END;

CREATE TRIGGER Stocks_BEFORE_DELETE BEFORE
DELETE ON Stocks FOR EACH ROW
BEGIN
        UPDATE Stock_collection SC SET
Remain_Amount=Remain_Amount-OLD.Stockamount
WHERE SC.Iid=OLD.Stocks_Iid;
   INSERT INTO History_Stocks SET
Stockid=OLD.stockid, Stocks_Wid=OLD.Stocks_Wid,
Stocks_Iid=OLD.Stocks_Iid,
Stockamount=OLD.Stockamount,
Stockarea=OLD.Stockarea, Instocktime=OLD.Instocktime,
Outstocktime=NOW();
END;

CREATE TRIGGER Inbound_AFTER_INSERT AFTER
INSERT ON Inbound FOR EACH ROW
BEGIN
        INSERT INTO Suppliers_Order_statistics SET
Sid=NEW.Suppliers_Sid, Inbound_id=NEW.Inbound_id;
END;

CREATE TRIGGER Inbound_details_AFTER_INSERT
AFTER INSERT ON Inbound_details FOR EACH ROW
BEGIN
        UPDATE Suppliers_Order_statistics SET
Completetime=NEW.Completetime,
Inbound_Amount=NEW.Amount,
Money=cast(NEW.Amount AS
DECIMAL(10,2))*NEW.Unit_Price WHERE
Inbound_id=NEW.Inbound_id;
END; $$

DELIMITER $$
```