CPSC 304 Project Cover Page

Milestone #: 2

Date: February 24, 2023

Group Number: 93

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Makafui Amouzouvi	86537412	d0q3f	hortenseamouzouvi@gmail.com
Mankala	78389624	g3h2k	nilaym171002@gmail.com
Linh Pham	13150578	j8z2b	phamhlinh01@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

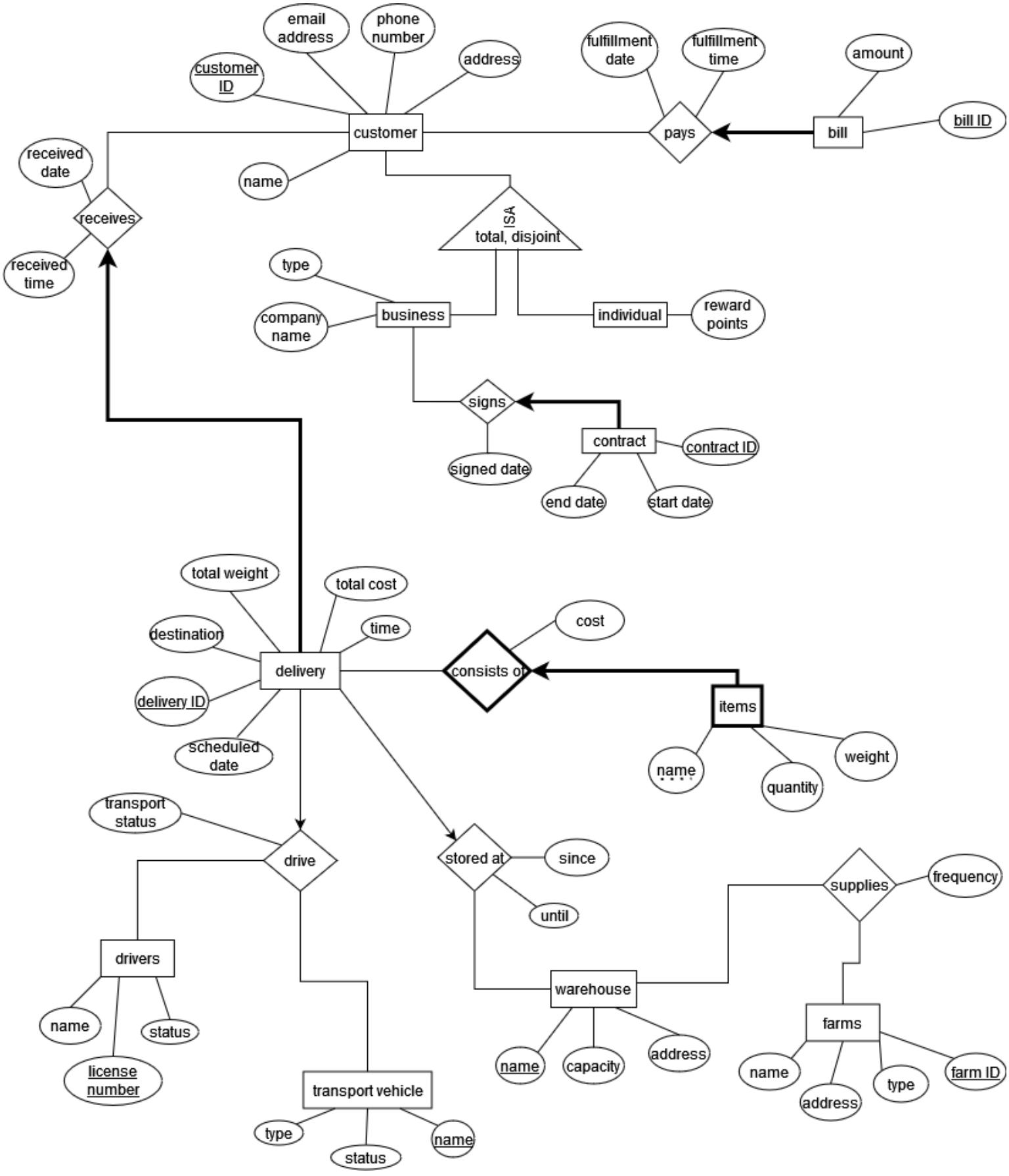
Each group must provide the following as a single PDF file:

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 details about your project.

The domain is food product supply management. So, it involves the logistics behind gathering and storing food products and supplying them to customers. The application is from the perspective of a supplier company managing different deliveries and orders from farmers to different customers (who could be either individuals or businesses). We're trying to address the different components of logistics behind delivering food products to customers.

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.


email address
phone number
customer ID
address
customer
fulfillment date
fulfillment time
pays
amount
bill
bill ID
received date
receives
received time
name
ISA
total, disjoint
type
company name
business
individual
reward points
signs
signed date
contract
contract ID
end date
start date
total weight
total cost
destination
time
delivery
delivery ID
scheduled date
consists of
cost
items
name
quantity
weight
transport status
drive
stored at
since
until
supplies
frequency
drivers
name
status
license number
transport vehicle
type
status
name
warehouse
name
capacity
address
farms
name
address
type
farm ID

If you have decided not to implement the suggestions given by your project mentor, please be sure to leave a note stating why. This is not to say that you must do everything that your project mentor says. In many instances, there are trade-offs between design choices and your decision may be influenced by different factors. That being said, your TAs will often leave suggestions that are meant to help massage your project into a form
that will fit with the requirements in future project milestones. If you choose not to take their advice, it would be helpful for them to know why in order to better assist the group moving forward.

Explanation for changing the diagram : Our diagram was slightly focusing on ordering, so we changed it to focus more on the shipping process as suggested by our TA. We removed the "Orders" entity set, so that the database doesn't manage customers placing orders anymore. We added a "Contracts" entity set for the Business customers, to represent a supplying contract between a business and the company.

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.

We're denoting primary keys as underlines, foreign keys in bold, candidate keys with the "UNIQUE" key word after the attribute, and other constraints also after the attribute name. We're also using the SQL "DATE" and "DATETIME" types for attributes that are dates or timestamps.

Delivery (deliveryID: integer, destination: string NOT NULL, deliveryTime: DATETIME, scheduledDate: DATE NOT NULL, totalWeight: real, totalCost: real, receivedTime: DATETIME, receivedDate: DATE, **customerID**: integer NOT NULL, transportStatus: string, **driverLicenseNumber**: integer, **transportVehicleName**: string, storedSince: string, storedUntil: string, **warehouseName**: string)

Customer(customerID: integer, name: string NOT NULL, emailAddress: string NOT NULL UNIQUE, phoneNumber: integer NOT NULL, address: string NOT NULL)

Business(**customerID**: integer, type: string, companyName: string NOT NULL)

Signs_Contract (contractID: integer, startDate: DATE NOT NULL, endDate: DATE NOT NULL, signedDate: DATE, **customerID**: string NOT NULL)

Individual(**customerID**: integer, rewardPoints: integer)

Farm(farmID: integer, address: string NOT NULL UNIQUE, name: string, type: string)

Farm_Warehouse_Supplies(**farmID**: integer, **warehouseName**: string, frequency: string)

Warehouse (name: string, capacity: integer, address: string NOT NULL UNIQUE)

Drivers (licenseNumber: integer, name: string NOT NULL, status: string)

Pays_Bill (billID: string, **customerID:** integer NOT NULL, fulfillmentDate: DATE, fulfillmentTime: DATETIME, amount: real NOT NULL)

TransportVehicle (name: string, type: string, status: string)

Items (**deliveryID**: integer, name: string, cost: real, weight: real, quantity: integer)
This contains both the relationship set Consists_of and entity set Items. The cost in Items is the cost of the item per pound. The "name" attribute here is a partial key.

5. Functional Dependencies (FDs)
a. Identify the functional dependencies in your relations, including the ones involving all candidate keys (including the primary key).
PKs and CKs are considered functional dependencies and should be included in the list of FDs. You do not need to include trivial FDs such as A → A.
Note: In your list of FDs, there must be some kind of valid FD other those identified by a PK or CK. If you observe that no relations have FDs other than the PK and CK(s), then you will have to intentionally add some (meaningful) attributes to show valid FDs. We want you to get a good normalization exercise. Your design must go through a normalization process.

In Delivery:

- deliveryID → {deliveryID, destination, deliveryTime, scheduledDate, totalWeight, totalCost, receivedTime, receivedDate, customerID, transportStatus, driverLicenseNumber, transportVehicleName, storedSince, storedUntil, warehouseName}
- {warehouseName, scheduledDate, destination} → {receivedDate}
    - Given a warehouse, a scheduled date for the delivery, and a destination, the day that the package received is always the same. We're adding the constraint that the supplier always reliably delivers a package to a certain destination within a certain amount of days, depending on which warehouse it's stored in.

In Customer:

- customerID → {customerID, name, emailAddress, phoneNumber, address}
- emailAddress → {customerID, name, emailAddress, phoneNumber, address}
- phoneNumber → {address}

In Business:

- customerID → {customerID, type, companyName}

In Signs_Contract:

- contractID → {contractID, startDate, endDate, signedDate, customerID}
- signedDate → {startDate}
    - We're adding the constraint that the start date of the contract is always one day after the contract is signed.

In Individual:

- customerID → {customerID, rewardPoints}

In Farm

- farmID → {farmID, address, name, type}
- address → {farmID, address, name, type}

In Farm_Warehouse_Supplies

- {farmID, warehouseName} → {farmID, warehouseName, frequency}

In Warehouse:

- name → {name, capacity, address}

In Drivers

- licenseNumber → {licenseNumber, name, status}

In Pays_Bill

- billID → {billID, customerID, fulfillementDate, fulfillmentTime, amount}

In TransportVehicle:

- name → {name, type, status}

In Items:

- {deliveryID, name} → {deliveryID, name, cost, weight, quantity}
- name → {cost}
    - Each item has a cost per pound, that is determined by knowing its name.

6. Normalization

a. Normalize 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 normalization. You should show the steps taken for the decomposition. Should there be errors, and no work is shown, no partial credit can be awarded without steps shown.
The format should be the same as Step 3, with tables listed similar to Table1(attr1: domain1, attr2: domain2, ...). ALL Tables must be listed, not only the ones post-normalization.

- Items (**deliveryID**: integer, name: string, cost: real, weight: real, quantity: integer)
  FD = { (deliveryID, name) → all attributes, name → cost}
  "name" is not a candidate key, so Items is not in BCNF. We have :
  R1(name, cost); R2(deliveryID, name, weight, quantity) [taking care of name → cost)]
  - R1 is now BCNF because "name" is the primary key and the only determinant
  - R2 is BCNF as well, since it doesn't have any FDs besides the PKs

  Final Tables:
  R1 renamed to ItemCosts(name: string, cost : real) with primary and candidate key "name".

  R2 renamed to Items (**deliveryID** : integer, **name**: string, weight: real, quantity: integer) with primary and candidate key (deliveryID, name) and foreign key "name" and deliveryID

- Signs_Contract (contractID: integer, startDate: DATE NOT NULL, endDate: DATE NOT NULL, signedDate: DATE, **customerID**: string NOT NULL)
  FD = { contractID → all attributes, signedDate → startDate}

signedDate is not a candidate key, so Signs_Contract is not in BCNF. so we have :

R3(signedDate, startDate); R4(contractID, signedDate, endDate, customerID) [taking care of signedDate → startDate]

- R3 is now BCNF because signedDate is the primary key and the only determinant
- R4 is BCNF too

Final Tables :
R3 renamed to Contract_Dates(signedDate, startDate) with primary key signedDate.

R4 renamed to Signs_Contract(contractID, **signedDate**, endDate, **customerID**) with primary key contractID and foreign keys signedDate and customerID

- Delivery (deliveryID: integer, destination: string NOT NULL, deliveryTime: DATETIME, scheduledDate: DATE NOT NULL, totalWeight: real, totalCost: real, receivedTime: DATETIME, receivedDate: DATE, **customerID**: integer NOT NULL, transportStatus: string, **driverLicenseNumber**: integer, **transportVehicleName**: string, storedSince: string, storedUntil: string, **warehouseName**: string)

  FD = { deliveryID → all attributes, (warehouseName, scheduledDate, destination) → receivedDate }

  (warehouseName, scheduledDate, destination) is not a candidate key, so Delivery is not in BCNF.

So we have:

R5(warehouseName, scheduledDate, destination, receivedDate)

and R6(deliveryID, destination, deliveryTime, scheduledDate, totalWeight, totalCost, customerID, transportStatus, driverLicenseNumber, transportVehicleName, storedSince, warehouseName) [taking care of (warehouseName, scheduledDate, destination) → receivedDate]

- R5 is now BCNF because (warehouseName, scheduledDate, destination) is a primary key and the only determinant
- R6 is BCNF too

Final Tables:
R5 renamed to DeliveryReceived(warehouseName, scheduledDate, destination, receivedDate) with primary key (warehouseName, scheduledDate, destination)

R6 renamed to Delivery(deliveryID : integer, **destination** string NOT NULL, deliveryTime : string, **scheduledDate** : string NOT NULL, totalWeight : real, totalCost : real, **customerID**: integer NOT NULL, transportStatus : string, **driverLicenseNumber**: integer, **transportVehicleName** : string, storedSince : string, **warehouseName :** string) with primary key deliveryID, foreign keys customerID, destination, scheduledDate, driverLicenseNumber, transportVehicleName, and warehouseName

- Customer(customerID: integer, name: string NOT NULL, emailAddress: string NOT NULL UNIQUE, phoneNumber: integer NOT NULL, address: string NOT NULL)

  FD = { customerID → all attributes, phoneNumber → address}
  phoneNumber is not a candidate key so Customer is not in BCNF.
  So we have :
  R7(phoneNumber, address), R8(customerID, name, emailAddress, phoneNumber)
  - R7 is now in BCNF because phoneNumber is a primary key and the only determinant
  - R8 is also in BCNF

Final Tables:
R7 renamed to CustomerPhoneAddress(phoneNumber :integer NOT NULL, address : string NOT NULL) with primary key phoneNumber

R8 renamed to Customer(customerID : integer, name : string NOT NULL, emailAddress : string NOT NULL, **phoneNumber :** integer NOT NULL) with primary key customerID and foreign key phoneNumber

---

All tables after normalization: We'll use the same format as in part 4, where primary keys are underlined, candidate keys have the "UNIQUE" keyword listed after the attribute, and foreign keys are in bold

Delivery(deliveryID : integer, **destination:** string NOT NULL, deliveryTime : string, **scheduledDate** : string NOT NULL, totalWeight : real, totalCost : real, **customerID**: integer NOT NULL, transportStatus : string, **driverLicenseNumber**: integer, **transportVehicleName** : string, storedSince : string, **warehouseName :** string)

DeliveryReceived(warehouseName, scheduledDate, destination, receivedDate)

Customer(customerID : integer, name : string NOT NULL, emailAddress : string NOT NULL, **phoneNumber :** integer NOT NULL)

CustomerPhoneAddress(phoneNumber :integer NOT NULL, address : string NOT NULL)

Business(**customerID**: integer, type: string, companyName: string NOT NULL)

Signs_Contract(contractID, **signedDate**, endDate, **customerID**)

Contract_Dates(signedDate, startDate)

Individual(**customerID**: integer, rewardPoints: integer)

Farm(farmID: integer, address: string NOT NULL UNIQUE, name: string, type: string)

Farm_Warehouse_Supplies(**farmID**: integer, **warehouseName**: string, frequency: string)

Warehouse (name: string, capacity: integer, address: string NOT NULL UNIQUE)

Drivers (licenseNumber: integer, name: string NOT NULL, status: string)

Pays_Bill (billID: string, **customerID:** integer NOT NULL, fulfillmentDate: DATE, fulfillmentTime: DATETIME, amount: real NOT NULL)

TransportVehicle (name: string, type: string, status: string)

Items (**deliveryID** : integer, **name**: string, weight: real, quantity: integer)

ItemCosts(name: string, cost : real)

7. The SQL DDL statements required to create all the tables from item #5. The statements should use the appropriate foreign keys, primary keys, UNIQUE constraints, etc.

```
CREATE TABLE Delivery(
    deliveryID INTEGER,
    destination CHAR(256) NOT NULL,
    deliveryTime DATETIME,
    scheduledDate DATE NOT NULL,
    totalWeight REAL NOT NULL,
    totalCost REAL,
    customerID INTEGER NOT NULL,
    transportStatus CHAR(16),
    driverLicenseNumber INTEGER,
```

```
    transportVehicleName CHAR(64),
    storedSince DATE,
    warehouseName CHAR(64),
    PRIMARY KEY (deliveryID),
    FOREIGN KEY (customerID) REFERENCES Customer ON DELETE
CASCADE,
    FOREIGN KEY (driverLicenseNumber) REFERENCES Driver,
    FOREIGN KEY (transportVehicleName) REFERENCES TransportVehicle,
    FOREIGN KEY (warehouseName) REFERENCES Warehouse,
    FOREIGN KEY (scheduledDate) REFERENCES DeliveryReceived,
    FOREIGN KEY (destination) REFERENCES DeliveryReceived
)

CREATE TABLE DeliveryReceived(
    warehouseName CHAR(64),
    scheduledDate DATE NOT NULL,
    destination CHAR(256) NOT NULL,
    receivedDate DATE,
    PRIMARY KEY (warehouseName, scheduledDate, destination)
    FOREIGN KEY (warehouseName) REFERENCES Warehouse
)

CREATE TABLE Customer(
    customerID INTEGER,
    name CHAR(256) NOT NULL,
    emailAddress CHAR(256) NOT NULL,
    phoneNumber INTEGER,
    PRIMARY KEY (customerID),
    UNIQUE (emailAddress),
    FOREIGN KEY (phoneNumber) REFERENCES CustomerPhoneAddress
)

CREATE TABLE CustomerPhoneAddress(
    phoneNumber INTEGER NOT NULL,
    address CHAR(512) NOT NULL,
```

```
    PRIMARY KEY (phoneNumber)
)

CREATE TABLE Business(
    customerID INTEGER,
    companyName CHAR(128) NOT NULL,
    type CHAR(16),
    PRIMARY KEY (customerID)
    FOREIGN KEY (customerID) REFERENCES Customer ON DELETE
CASCADE
)

CREATE TABLE Individual(
    customerID INTEGER,
    rewardPoints INTEGER,
    PRIMARY KEY (customerID)
    FOREIGN KEY (customerID) REFERENCES Customer ON DELETE
CASCADE
)

CREATE TABLE Signs_Contract(
    contractID INTEGER,
    signedDate DATE,
    endDate DATE,
    customerID INTEGER NOT NULL,
    PRIMARY KEY (contractID)
    FOREIGN KEY (customerID) REFERENCES Customer ON DELETE
CASCADE
    FOREIGN KEY (signedDate) REFERENCES Contract_Dates
)

CREATE TABLE Contract_Dates(
    signedDate DATE,
    startDate DATE,
    PRIMARY KEY(signedDate))
```

```
CREATE TABLE Farm(
    farmID INTEGER,
    name CHAR(64),
    address CHAR(512) NOT NULL,
    type CHAR(32),
    PRIMARY KEY (farmID)
    UNIQUE (address)
)

CREATE TABLE Farm_Warehouse_Supplies(
    farmID INTEGER,
    frequency CHAR(32),
    warehouseName CHAR(64),
    PRIMARY KEY (farmID, warehouseName),
    FOREIGN KEY (warehouseName) REFERENCES Warehouse ON DELETE
CASCADE,
    FOREIGN KEY (farmID) REFERENCES Farm ON DELETE CASCADE
)

CREATE TABLE Warehouse(
    name CHAR(256),
    capacity INTEGER,
    address CHAR(512) NOT NULL,
    PRIMARY KEY (name),
    UNIQUE (address)
)

CREATE TABLE Pays_Bill(
    billID INTEGER,
    customerID INTEGER NOT NULL,
    fulfillmentDate DATE,
    fulfillmentTime DATETIME,
    amount REAL NOT NULL,
    PRIMARY KEY (billID),
```

```
    FOREIGN KEY (customerID) REFERENCES Customer ON DELETE
CASCADE
)

CREATE TABLE Drivers(
    licenseNumber INTEGER,
    name CHAR(64) NOT NULL,
    status CHAR(32),
    PRIMARY KEY (licenseNumber)
)

CREATE TABLE TransportVehicle(
    name CHAR(64),
    type CHAR(32),
    status CHAR(32),
    PRIMARY KEY (name)
)

CREATE TABLE Items(
    deliveryID INTEGER,
    name CHAR(32),
    weight REAL,
    quantity INTEGER,
    PRIMARY KEY(deliveryID, name)
    FOREIGN KEY (deliveryID) REFERENCES Delivery
    FOREIGN KEY (name) REFERENCES ItemCosts
)

CREATE TABLE ItemCosts(
    name CHAR(32),
    cost REAL, // cost per pound of item
    PRIMARY KEY (name)
)
```

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 on.

```
INSERT INTO ItemCosts VALUES(“apples”, 25)
INSERT INTO ItemCosts VALUES(“mango” , 20)
INSERT INTO ItemCosts VALUES(“mango” , 35)
INSERT INTO ItemCosts VALUES(“orange” , 12)
INSERT INTO ItemCosts VALUES(“banana”, 15)

INSERT INTO Items VALUES(1, “apples”, 12.2, 10)
INSERT INTO Items VALUES(2, “mango”, 5, 2)
INSERT INTO Items VALUES(3, “mango”, 10, 4)
INSERT INTO Items VALUES(4, “orange”, 2,  1)
INSERT INTO Items VALUES(5, “banana”, 2, 3)

INSERT INTO Warehouse VALUES(“warehouse_1”, 500 ,” 5021 Granville St”)
INSERT INTO Warehouse VALUES(“warehouse_2”, 750 ,” 302 Davie St”)
INSERT INTO Warehouse VALUES(“warehouse_3”, 400 ,” 229 English Bay”)
INSERT INTO Warehouse VALUES(“warehouse_4”, 550 ,” 5959 Student Union Blvd”)
INSERT INTO Warehouse VALUES(“warehouse_5”, 1090 ,” 5612 Newton Wynd”)

INSERT INTO Contract_Dates VALUES(2023-01-23, 2023- 02-01)
INSERT INTO Contract_Dates VALUES(2022-04-03, 2022- 04-09)
INSERT INTO Contract_Dates VALUES(2022-05-23, 2022- 06-07)
INSERT INTO Contract_Dates VALUES(2022-03-13, 2022- 03-19)
INSERT INTO Contract_Dates VALUES(2022-10-17, 2022- 10-29)

INSERT INTO Signs_Contract(1234, 2023-01-23, 2024-01-23, 5001)
INSERT INTO Signs_Contract(1236, 2022-04-03, 2024-02-24, 5002)
INSERT INTO Signs_Contract(1237, 2022-05-23, 2023-03-23, 5003)
INSERT INTO Signs_Contract(1238, 2022-03-13, 2023-08-23, 5004)
INSERT INTO Signs_Contract(1239, 2022-10-17, 2024-01-23, 5005)


INSERT INTO Pays_Bill VALUES (356, 4511, 2023- 02-23, 23:44, 100.7)
INSERT INTO Pays_Bill VALUES (357, 4512, 2023- 02-23, 23:44, 356.7)
INSERT INTO Pays_Bill VALUES (358, 4513, 2023- 02-24, 13:22, 200.0)
INSERT INTO Pays_Bill VALUES (359, 4562, 2023- 02-25, 18:41, 1500.45)
INSERT INTO Pays_Bill VALUES (360, 4564, 2023- 02-28, 10:44, 100.7)
```

INSERT INTO TransportVehicles (“truck_1”, “semi-truck”, “in-use”)
INSERT INTO TransportVehicles (“truck_2”, “refrigerated”, “free”)
INSERT INTO TransportVehicles (“truck_3”, “box truck”, “in-use”)
INSERT INTO TransportVehicles (“truck_4”, “refrigerated”, “in-use”)
INSERT INTO TransportVehicles (“truck_5”, “semi-trailer”, “free”)

INSERT INTO Drivers(1234567, “Joe Smith”, “delivering”)
INSERT INTO Drivers(1234568, “Brenda Fox”, “free”)
INSERT INTO Drivers(1234569, “Terry Daniels”, “delivering”)
INSERT INTO Drivers(1234570, “Bob Newman”, “free”)
INSERT INTO Drivers(1234571, “Sara Taylor”, “delivering”)

INSERT INTO Farm_Warehouse_Supplies(234, “bi-weekly”, “warehouse_1”)
INSERT INTO Farm_Warehouse_Supplies(235, “monthly”, “warehouse_1”)
INSERT INTO Farm_Warehouse_Supplies(236, “weekly”, “warehouse_2”)
INSERT INTO Farm_Warehouse_Supplies(237, “bi-weekly”, “warehouse_3”)
INSERT INTO Farm_Warehouse_Supplies(238, “monthly”, “warehouse_3”)

INSERT INTO Farm(234, “Maplewood Farm”, “405 Seymour River Pl, North Vancouver”, “fruits”)
INSERT INTO Farm(235, “Maan Farms”, “790 McKenzie Rd, Abbotsford”, “fruits”)
INSERT INTO Farm(236, “Aldor Acres”, “24990 84 Ave, Langley Twp”, “vegetables”)
INSERT INTO Farm(237, “Southland Farms”,”6767 Balaclava St, Vancouver”, “pumpkins”)
INSERT INTO Farm(238, “Eagle Acres Dairy Farm”, “8796 240 St, Langley Twp”, “meat”)

INSERT INTO Customer VALUES (4560, “Alex Ba”, “alexba@gmail.com” , 7783321234)
INSERT INTO Customer VALUES (4561, “Alicia Keys”, “cust2@gmail.com” , 7786321234)
INSERT INTO Customer VALUES (4562, “Justin B”, “cust3@gmail.com” , 7789321234)
INSERT INTO Customer VALUES (4563, “Riha Sed”, “cust4@gmail.com” , 7782321234)
INSERT INTO Customer VALUES (4564, “Whitney Houston”, “cust5@gmail.com” , 7781321234)

INSERT INTO Customer VALUES (4510, “cust4”, “alexba@gmail.com” , 7783321231)
INSERT INTO Customer VALUES (4511, “cust5”, “cust5@gmail.com” , 7786321232)

INSERT INTO Customer VALUES (4512, “cust6”, “cust6@gmail.com” , 7789321233)
INSERT INTO Customer VALUES (4513, “cust7”, “cust7@gmail.com” , 7782321234)
INSERT INTO Customer VALUES (4514, “cust8”, “cust8@gmail.com” , 7781321235)

INSERT INTO Individual VALUES(4560, 2000)
INSERT INTO Individual VALUES(4561, 900)
INSERT INTO Individual VALUES(4562, 2030)
INSERT INTO Individual VALUES(4563, 1003)
INSERT INTO Individual VALUES(4564, 4031)

INSERT INTO Business VALUES(4510, “Browns Crafthouse”, “Restaurant”)
INSERT INTO Business VALUES(4511, “Save on Foods”, “Grocery Store”)
INSERT INTO Business VALUES(4512, “Costco”, “Wholesale Store”)
INSERT INTO Business VALUES(4513, “Uncle Fatih’s”, “Restaurant”)
INSERT INTO Business VALUES(4514, “Safeway”, “Grocery Store”)

INSERT INTO Delivery VALUES (456, “2345 West Mall”, 11:16, 2023-02-25, 45.5, 1600.56, 4560, “in-delivery”, 5634538, 2023-02-23, “warehouse_1”)
INSERT INTO Delivery VALUES (457, “1245 West Mall’, 12:16, 2023-02-26, 45.5, 1700.56, 4561, “delivered”, 4535768, 2023-02-12, “warehouse_2”)
INSERT INTO Delivery VALUES (458, “1345 West Mall”, 14:16, 2023-02-25, 45.5, 1800.56, 4510, “in warehouse”, 453535375, 2023-02-22, “warehouse1”)
INSERT INTO Delivery VALUES (459, “1545 West Mall”, 16:16, 2023-02-28, 45.5, 1900.56, 4511, “delayed”, 453456356, 2023-02-24, “warehouse3”)
INSERT INTO Delivery VALUES (460, “1645 West Mall”, 17:16, 2023-02-25, 45.5, 1600.56, 4513, “in-delivery”, 23534553, 2023-02-22, “warehouse4”)

INSERT INTO CustomerPhoneAddress VALUES(7783321234, “2345 West Mall”)
INSERT INTO CustomerPhoneAddress VALUES(7786321234, “1245 West Mall”)
INSERT INTO CustomerPhoneAddress VALUES(7783321231, “1345 West Mall”)
INSERT INTO CustomerPhoneAddress VALUES(7786321232, “1545 West Mall”)
INSERT INTO CustomerPhoneAddress VALUES(7782321234, “1645 West Mall”)

INSERT INTO DeliveryReceived VALUES (“warehouse_1”, 2023-02-26, “2345 west mall”, 2023- 02- 26)
INSERT INTO DeliveryReceived VALUES (“warehouse_1”, 2023-02-25, “1245 west mall”, 2023- 02- 25)

INSERT INTO DeliveryReceived VALUES (“warehouse_2”, 2023-02-28, “1345 West Mall”, 2023- 02- 29)
INSERT INTO DeliveryReceived VALUES (“warehouse_2”, 2023-02-25, “1545 West Mall”, 2023- 02- 26)
INSERT INTO DeliveryReceived VALUES (“warehouse_3”, 2023-02-25, “1645 West Mall”, 2023- 02- 26)