**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**/ Name: Divya Pramasani Mohandoss /**

**/ Course: CS6070 /**

**/ Assign: 5 /**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

/\*A\*/ **SHIPMENT Table:**

SHIPMENT(ShipmentNumber, Shipper, Phone, Contact, From, Departure, Arrival, Contents, InsuredValue)

**Multivalued Dependencies:** None

**Functional Dependencies:**

ShipmentNumber ⟶ Shipper

ShipmentNumber ⟶ Phone

ShipmentNumber ⟶ Contact

ShipmentNumber ⟶ From

ShipmentNumber ⟶ Departure

ShipmentNumber ⟶ Arrival

ShipmentNumber ⟶ Contents

Shipper ⟶ Phone

Shipper ⟶ Contact

**Candidate Keys:** ShipmentNumber

**Primary Key:** ShipmentNumber

**Foreign Key:** None

**ITEM Table:**

ITEM (Item, Date, City, Store, Salesperson, Price)

**Multivalued Dependencies:**

City ⟶ Store

(City, Store) ⟶ Item

(City, Store) ⟶ Salesperson

**Functional Dependencies:**

(Item, Date, City, Store) ⟶ Salesperson

(Item, Date, City, Store) ⟶ Price

**Candidate Keys:**

(Item, Date, City, Store)

**Primary Key**: Item, Date, City, Store

**Foreign Key:** No

**/\*B\*/ List questions you would ask Phillip to verify your assumptions.**

1. Do you always use one contact for each shipper?
2. Does each shipper have one phone number?
3. Do you buy from more than one store in each city in each country?
4. Can more than one store in the same country, but in a different city, have same name?
5. Can same city name in more than one country?
6. Can same item be purchased from different stores?
7. Are all the items purchased in one purchase from store shipped on the same shipment?

**/\*C/ Create tables as necessary to eliminate multivalued dependencies, if any.**

ITEM Table:

Case: 1 There are more than one item purchased from each store.

City ⟶ ⟶Store # multivalued dependencies

CITY\_STORE(City*,* Store)

(City, Store) ⟶ ⟶Item # multivalued dependencies

Case: 2 Item is available in many stores.

Item ⟶ ⟶ (City, Store) # multivalued dependencies

CITY\_STORE\_ITEM(Item, *City, Store*)

(City, Store) ⟶ ⟶Salesperson # multivalued dependencies

CITY\_STORE\_SALESPERSON(*City*, *Store*, Salesperson)

Separate ITEM table to track data on items purchased for import.

ITEM (Item, Date, City, Store, Salesperson, Price)

**/\*D/ The relationship between shipment and item data could be inferred by matching values in the From cells to values in the City cells. Describe two problems with that strategy.**

(1) A foreign key is supposed to represent the same thing in both tables in which it resides. In this case, the two attributes represent two different things. City in the Item purchase data is the city in which the item was purchased, but From in the Shipment data is the city from which the shipment departed. An item purchased in one city could be shipped from another.

(2) Data inconsistencies would make the connection useless. The data indicates that Worldwide shipper from the Philippines, but the Philippines is a country, not a city. QE Dining Sets were purchased in Manila, which is a city in the Philippines.

**/\*E/ Describe a change to this spreadsheet that does express the shipment/item relationship.**

Adding the ShipmentNumber into the Item purchase data. This will record which shipment contained the set of items purchased. This assumes that the entire purchase will be shipped on the same shipment, and if this is not true a more intricate foreign key system (an intersection table) would be needed.

**/\*F/ Assume that Phillip wishes to create an updateable database from this data. Design tables you think are appropriate. State all referential integrity constraints.**

The column SHIPMENT. Contents has been replaced with the SHIPMENT\_ITEM table, and that column has been eliminated to fix a general-purpose remarks column problem.

Referential Integrity Constraints are stated following each relation.

ITEM (Item, {additional columns – Description, Weight, etc.})

CITY\_STORE (City, Store, {additional columns – Country, Address, etc.})

CITY\_STORE\_SALESPERSON (City, Store, Salesperson)

WHERE CITY\_STORE\_SALESPERSON.(City, Store) must exist in CITY\_STORE.(City, Store)

CITY\_STORE\_ITEM (Item, City, Store)

WHERE CITY\_STORE\_ITEM.(City, Store) must exist in CITY\_STORE(City, Store), AND

WHERE CITY\_STORE\_ITEM.Item must exist in ITEM.Item

ITEM\_PURCHASE (Item, Date, City, Store, Salesperson, Price)

WHERE ITEM\_PURCHASE.(Item, City, Store) must exist in CITY\_STORE\_ITEM.(Item, City, Store)

SHIPPER (Shipper, Phone, Contact)

SHIPMENT (ShipmentNumber, Shipper, From, Departure, Arrival, Contents, InsuredValue)

WHERE SHIPMENT.Shipper must exist in SHIPPER.Shipper

SHIPMENT\_ITEM (ShipmentNumber, Item, Date, City, Store)

WHERE SHIPMENT\_ITEM.ShipmentNumber must exist in SHIPMENT.ShipmentNumber, AND

WHERE SHIPMENT\_ITEM.(Item, Date, City, Store) must exist in ITEM\_PURCHASE. (Item, Date, City, Store)

**/\*G/ Assume that Phillip wishes to create a read-only database from this data. Design tables you think are appropriate. State all referential integrity constraints**

The following table design denormalizes the tables back to almost the same set of tables that where implicit in the spreadsheet but provides the needed foreign key linkage. It also adds a table with Item and Store information. It would be possible to combine the first two tables into one.

Referential integrity constraints:

SHIPMENT (ShipmentNumber, Shipper, Phone, Contact, From, Departure, Arrival, Contents, InsuredValue)

SHIPMENT\_ITEM (ShipmentNumber, Item, Date, City, Store, Salesperson, Price)

WHERE SHIPMENT\_ITEM.ShipmentNumber must exist in SHIPMENT.ShipmentNumber, AND

WHERE SHIPMENT\_ITEM. (Item, City, Store) must exist in ITEM\_CITY\_STORE\_ITEM.(Item, City, Store)

ITEM\_ CITY\_STORE (Item, City, Store, Salesperson)

**/\*H/ Do these data have the multi value, multicolumn problem? If so, how will you deal with it?**

No, these data do not have a multi value, multi column problem.

**/\*I/ Do these data have the inconsistent data problem? If so, how will you deal with it?**

Yes, the data is inconsistent:

1. The Arrival date for Worldwide ShipmentNumber 49100300 is impossible because it is prior to the Departure date.

2. Singapore is misspelled.

3. Second, the shipment numbers from International seem reversed given the shipment dates.

4. Both city and country names are being recorded as data. For example, Manila is in the Philippines. Singapore is a city; Peru is a country.

To deal with these inconsistencies:

(1) The database should have the appropriate columns set with value constraints to ensure only correct data values are entered.

(2) We should call International and verify shipment numbers. If shipments with later shipping dates always have larger shipment numbers, we will need to write an application data checking routine.

(3) We need a business rule to determine how city and country information will be used. Once this is done, we can again add data constraints into the database.

**/\*J/ Do these data have a null value data problem? If so, how will you deal with it?**

Yes, the arrival date for ShipmentNumber 488955 is missing.

We need to call Marilyn, our contact at International, and ask if the shipment has arrived.

If so, we need to find out when it arrived, and enter the data.

**/\*K/ Do these data have the general-purpose remarks problem? If so, how will you deal with it?**

Yes, the Contents columns listed multiple values of items in each shipment. Not only does this obscure data, but it prevents a foreign key from being created.

Dealing with the problem by creating a separate ITEM table to store the data about each type of item, and a SHIPMENT\_ITEM table to store data about items in each shipment.