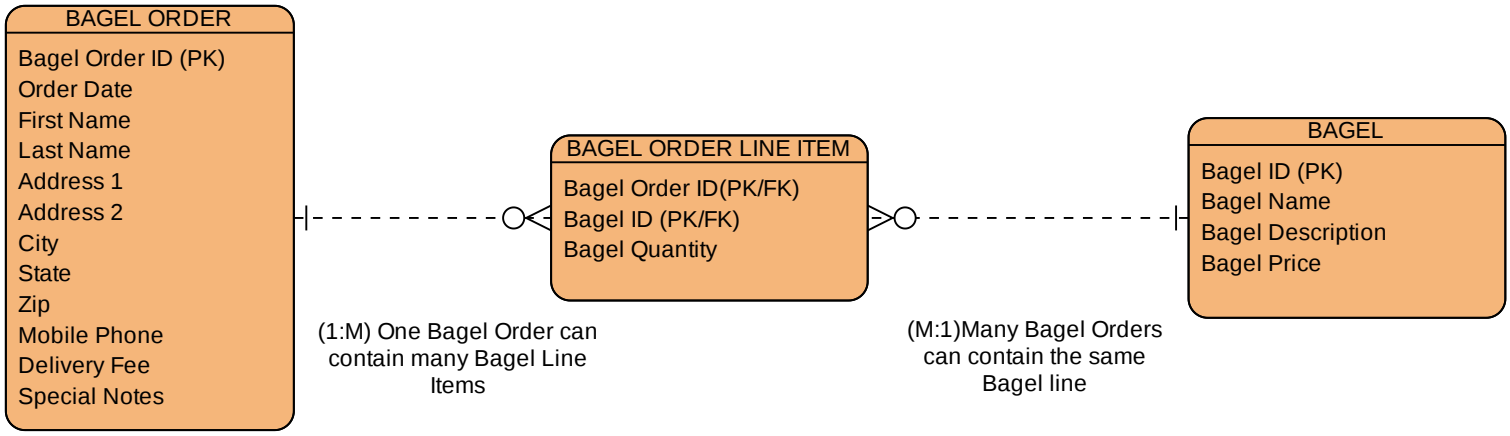
PART A VHT2 DUSTIN PHAN
NORA'S BAGELS
DATABASE PHYSICAL DIAGRAM
CREATED WITH:
https://online.visual-paradigm.com/

CONVERTING 1NF TO 2NF



In 2NF there cannot be partial dependencies to the primary key, in this case it was a composite primary key, so I seperated:
Order Date,First Name, Last Name,Adress 1 and 2, City,State,Zip,Mobile Phone, Delivery Fee and special notes to the Bagel Order. The reason is, those can be determined by the Bagel Order ID alone (unique), it does not need to depend on the Bagel ID which is another primary key.

The Same Logic with Bagel Name,Description,Price and Quantitiy applies,except with the Bagel ID(PK and unique within BAGEL table).
Note: the special note is under order because the customer will specifiy what they want catered to them witin the order. Also, the bagel quantitiy belongs in the join table because its " how many of that bagel , in a bagel order", identified by both the primary keys of that table, which is Bagel Order ID and Bagel ID.

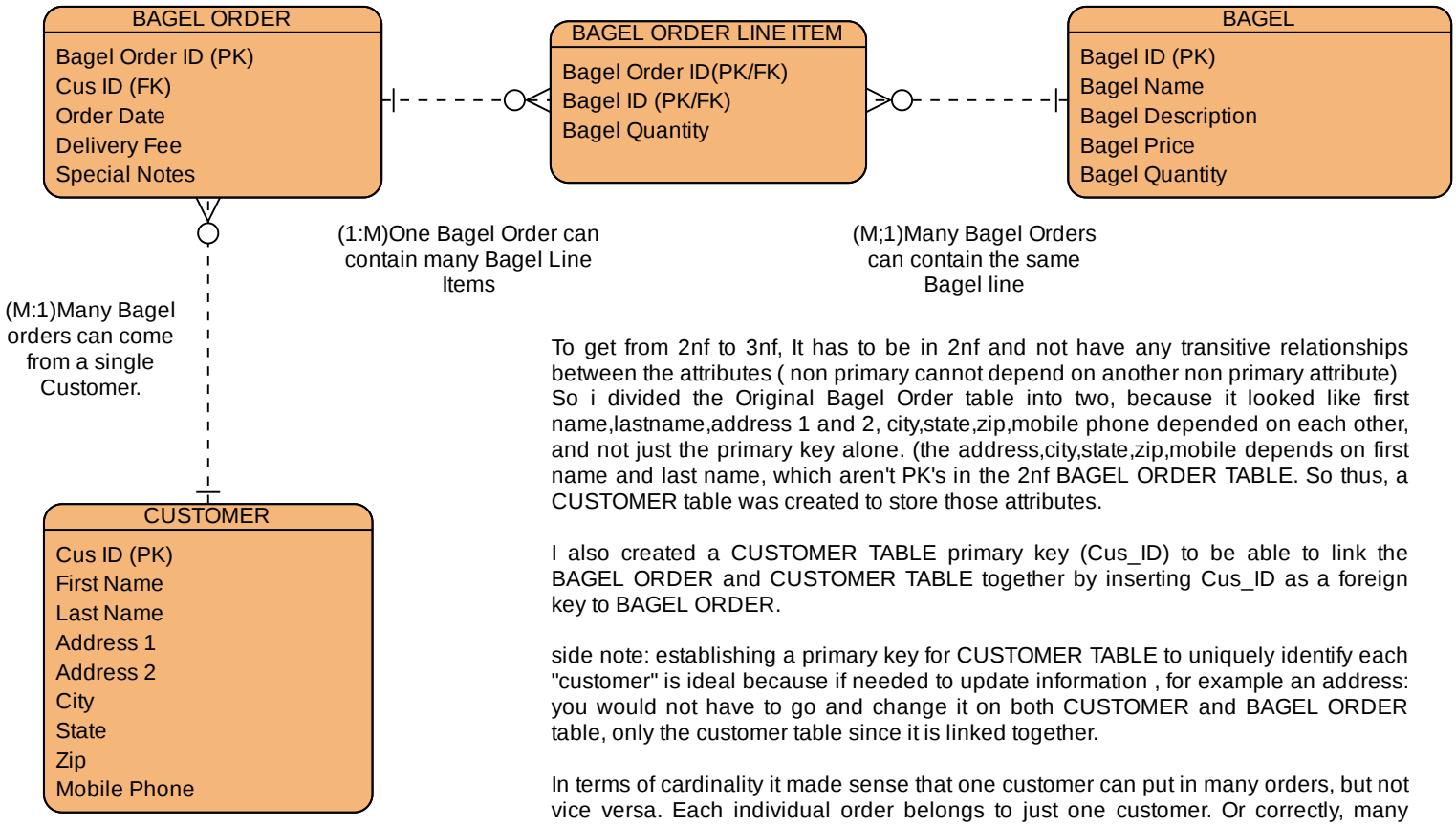
I created the Bagel Order Line Item table to link the two other tables to represent a many to many relationship.
Having the Bagel Order ID, Bagel ID as both the primary key and foreign keys, you're able to identify which Bagel ID connects to Bagel Order ID and vice versa, thus resulting in a relationship between the tables.

As for the Cardinality, One Bagel Order can contain many Bagel Order Lines.
For example: lets say an order with Bagel Order ID :1 has Bagel ID 1,2,3 within the order. We can figure out the associated name,description,price and quantity with the linking table.

Reading left to right, Many Bagel orders can have the same Bagel line within them. For example: Bagel Order ID: 1,2,3 all contain the Bagel ID 2. Lets say ID 2 is a chocolate bagel with associated description,price,quantity. We also find this through the linking table. In other words a single bagel line can be contained in many orders.

All in all, there is a possibility where a single order can contain multiple bagel order lines and also, the possibility of multiple orders containing the same Bagel line is prevalent , making this a many to many relationship. (M:N).

CONVERTING 2NF to 3NF



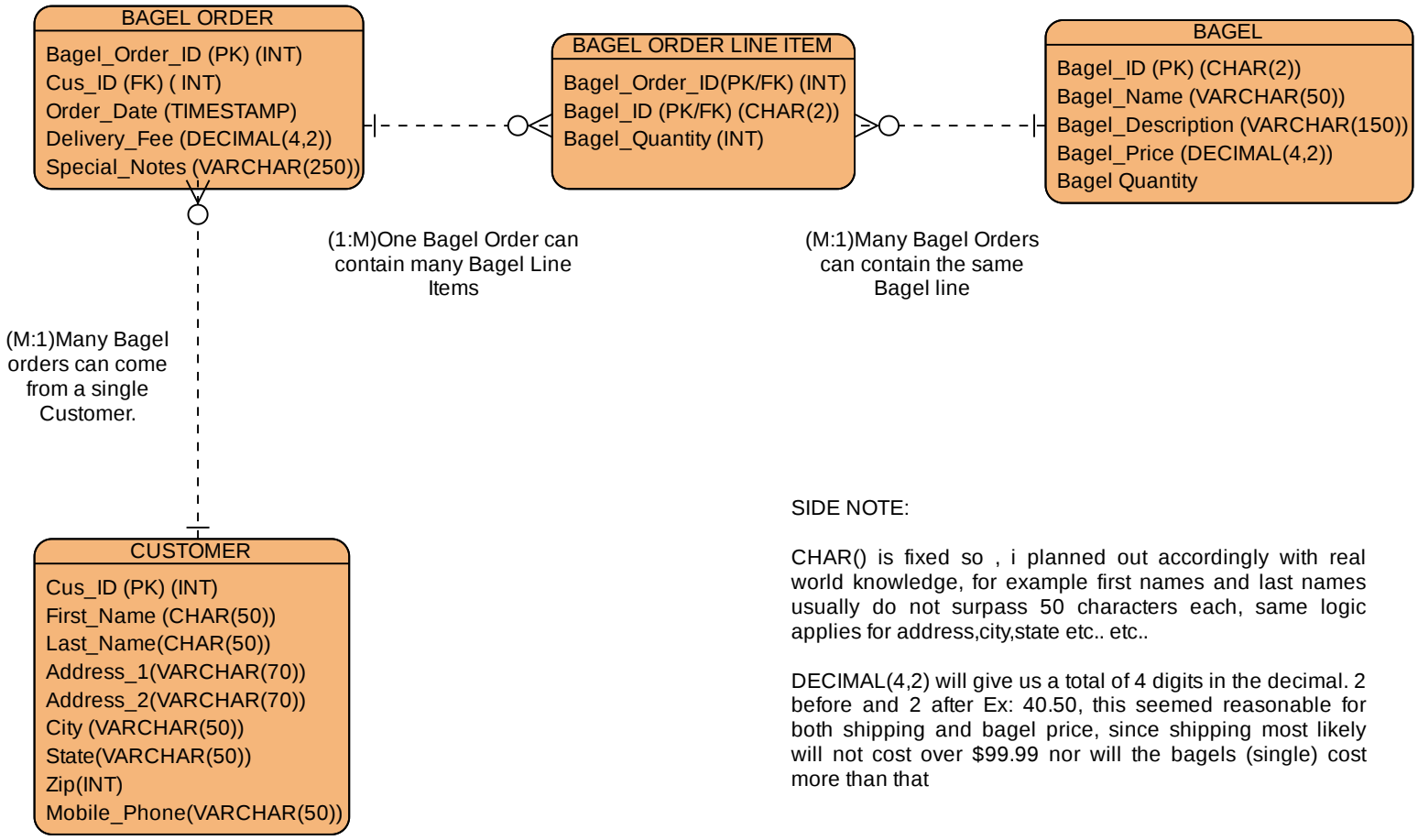
To get from 2nf to 3nf, It has to be in 2nf and not have any transitive relationships between the attributes (non primary cannot depend on another non primary attribute)
So i divided the Original Bagel Order table into two, because it looked like first name,lastname,address 1 and 2, city,state,zip,mobile phone depended on each other, and not just the primary key alone. (the address,city,state,zip,mobile depends on first name and last name, which aren't PK's in the 2nf BAGEL ORDER TABLE. So thus, a CUSTOMER table was created to store those attributes.

I also created a CUSTOMER TABLE primary key (Cus_ID) to be able to link the BAGEL ORDER and CUSTOMER TABLE together by inserting Cus_ID as a foreign key to BAGEL ORDER.

side note: establishing a primary key for CUSTOMER TABLE to uniquely identify each "customer" is ideal because if needed to update information , for example an address: you would not have to go and change it on both CUSTOMER and BAGEL ORDER table, only the customer table since it is linked together.

In terms of cardinality it made sense that one customer can put in many orders, but not vice versa. Each individual order belongs to just one customer. Or correctly, many orders can come from a single customer (if reading top to bottom).
With that said, i figured it is a 1:M relationship.

FINAL DATABASE PHYSICAL DESIGN



SIDE NOTE:

CHAR() is fixed so , i planned out accordingly with real world knowledge, for example first names and last names usually do not surpass 50 characters each, same logic applies for address,city,state etc.. etc..

DECIMAL(4,2) will give us a total of 4 digits in the decimal. 2 before and 2 after Ex: 40.50, this seemed reasonable for both shipping and bagel price, since shipping most likely will not cost over $99.99 nor will the bagels (single) cost more than that