Shakira Johnson
3/14/2023
Database Normalization

Normalization and Database Design

The goal of this project is to perform database normalization to achieve a table database design that is in third normal form. I will include explanations on how I took a table that was in originally in first normal form, to second normal, and then finally third normal form.

First Normal Form (1NF)

BAGEL ORDER	
PK	Bagel Order ID
PK	Bagel ID
	Order Date
	First Name
	Last Name
	Address 1
	Address 2
	City
	State
	Zip
	Mobile Phone
	Delivery Fee
	Bagel Name
	Bagel Description
	Bagel Price
	Bagel Quantity
	Special Notes

Second Normal Form (2NF)

BAGEL ORDER	
PK	Bagel Order ID
	Order Date
	First Name
	Last Name
	Address 1
	Address 2
	City
	State
	Zip
	Mobile Phone
	Delivery Fee
	Special Notes

1:M

BAGEL ORDER LINE ITEM	
PK / FK	Bagel Order ID
PK / FK	Bagel ID
	Quantity

M:1

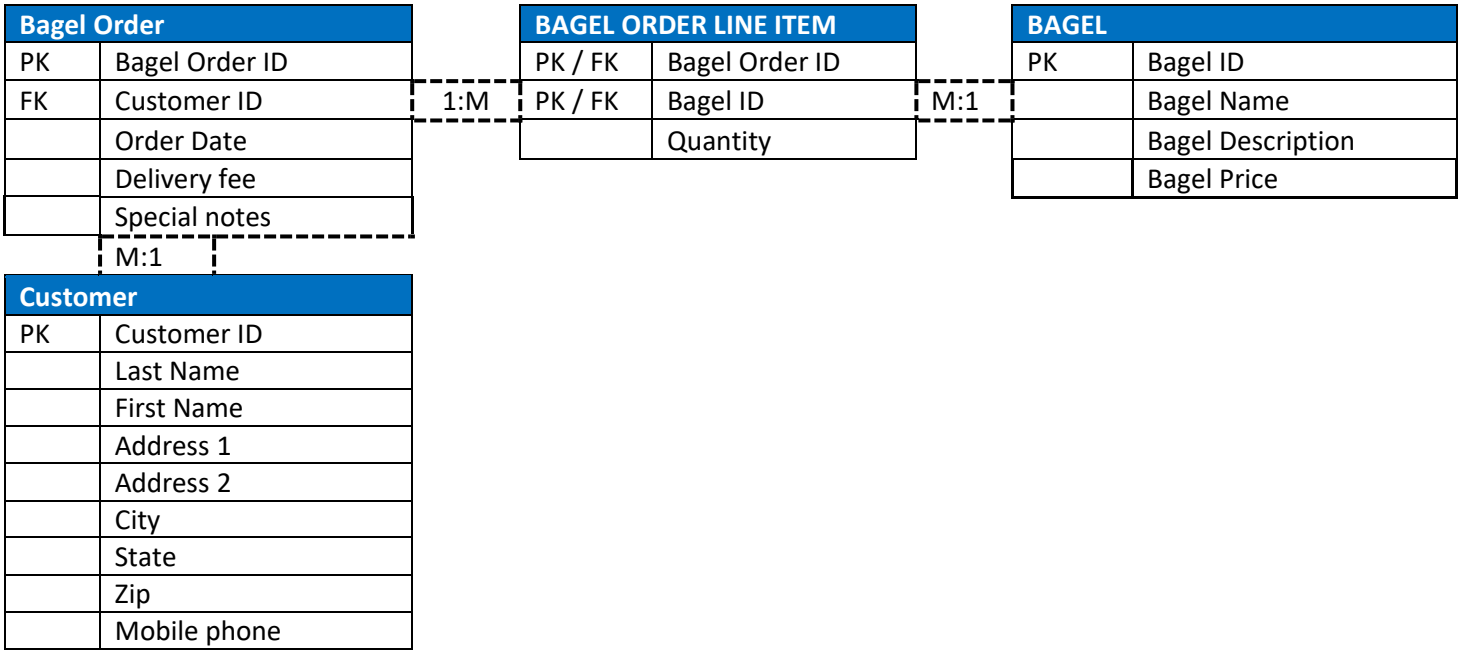
BAGEL	
PK	Bagel ID
	Bagel Name
	Bagel Description
	Bagel Price

From First Normal Form to Second Normal Form

I assigned attributes to the second normal form relation by looking at the attributes of the first normal form table and seeing how they were related to the composite primary key of Bagel ID and bagel order ID in the 1NF table. My goal was to make sure that the attributes are functionally dependent on the composite primary key and not just part of the composite primary key according to the definition of second normal form. I noticed that the attributes bagel name, description, and price in the 1NF table could be determined by the Bagel ID alone instead of using both primary keys. I then determined that these attributes could be put into a separate table using Bagel ID as the primary key. Another table was also added because there were attributes in the 1NF table that could be determined with only the bagel Order ID. The table is called “Bagel Order” and uses Bagel Order ID as the primary key. The orders quantity could only be determined by using both the bagel Order Id and the Bagel Id as a composite primary key in the Bagel Order Line Item table.

The cardinality was determined by looking at the relationship between the relations. A bagel order can contain many line items within one order, so I determined that the cardinality between Bagel Order and Bagel Order Line Item is 1 to many. There is a many to one relationship between Bagel Order Line Item and Bagel because many bagel order line items can have 1 bagel type and 1 bagel may be in many different order line items.

Third Normal Form (3NF)



From Second Normal Form to Third Normal Form

While observing the second normal form table I noticed that there were two separate entities being represented in the bagel order table. The second normal form Bagel Order table includes order information and customer information. It is a general rule that a well-formed table will not encompass more than one business concept. I decided that it was best to split the attributes related to bagel orders and the attributes related to customers into separate tables. To achieve third normal form the table 'Customer' was added. The Customer table also prevents transitive dependencies where an attribute can be determined not only by the primary key but also by another non key attribute. I also wanted to prevent repeating data because if a customer were to place multiple orders in the second normal form Bagel Order table, then their name, address, and phone would have to be repeated within the table. An attribute called Customer ID was added to the Customer table as a primary key for uniqueness and to make each specific customer easily identifiable.

Because there is another table added there must be another cardinality. I determined that the cardinality between Bagel Order and Customer is Many to One because 1 customer can have many bagel orders or in other words many bagel orders can only have one customer per order.

Final Physical Database Model with Data Types

Bagel Order		
PK	bagel_order_id	INT
FK	bagel_id	CHAR(2)
	order_date	TIMESTAMP
	delivery_fee	INT
	special_notes	VARCHAR(255)

1:M

BAGEL ORDER LINE ITEM		
PK / FK	bagel_order_id	INT
PK / FK	bagel_id	CHAR(2)
	quantity	INT

M:1

BAGEL		
PK	bagel_id	CHAR(2)
	bagel_name	VARCHAR(30)
	bagel_description	VARCHAR(30)
	bagel_price	NUMERIC(3,2)

M:1

Customer		
PK	customer_id	INT
	last_name	VARCHAR(50)
	first_name	VARCHAR(50)
	address_1	VARCHAR(50)
	address_2	VARCHAR(50)
	city	VARCHAR(30)
	state	CHAR(2)
	zip	VARCHAR(10)
	mobile_phone	VARCHAR(10)