1) A short description of the final project, and what it accomplished.

This is a basic foodbank database which holds quantities of an arbitrary foodbanks donated

money, food, and logins for volunteers and administrators. It also contains information

on shifts worked by volunteers.

It supports queries such as seeing the total number of donations from donors, sorting food based on category (protein, carbs etc.), and seeing which administrators have total participation to volunteer accounts.

2) A description of how your final schema differed from the schema you turned in.

I don't think the final schema differed too much from the one we turned in. I had

to adjust somethings once I was removed from the group as I got to customize the

project as I saw fit.

There is one case where I got rid of the group entity for simpler tracking of how many food items there were based on category.

It was different to make it simpler to track foods based on quantities. I didn’t feel the need to have a separate entity set for a foods primary nutrient.

Identification of functional dependencies

Distribution\_shift AND Collection\_shift:

- startTime, length, letter → startTime, length, letter

- The shift is identified by the startTime, letter and length, which are also the only attributes in the table that are not foreign keys

Mon\_Collects:

- did → name, phone, date, amount, medium, did

- From the donation ID, we can retrieve all the information about the donation, as it is unique

- There is no FD name → phone because multiple people could be using the same phone number

(i.e. multiple people in the same household using a landline) or one person

could be using more than one phone number or two different people could have the

same name (and therefore different phone numbers, etc).

Item\_Collects:

- did → name, phone, date, did

- From the donation ID, we can retrieve all

the information about the donation, as it is unique

- There is no FD name → phone because multiple people could

be using the same phone number (i.e. multiple people in the same household using a landline) or one person

could be using more than one phone number or two different people could have the same name (and therefore different phone numbers, etc).

expirationDate:

- date → date

- A date representing the date an item expires

GroupE:

- value → value

- It is just unique by value (Carbs, protein, nutrition etc)

Item:

- did, Category → did, Category

- Connects the name of the item and its category to its donation ID

Administrator:

- userName → userName

- The only attribute in this table is the primary key;

it is also a foreign key that references the Employee object

that the Administrator is a part of. userName is the username of the admin/employee.

Volunteer\_Add:

- userName → userName

- userName represents the username of the volunteer that

was added, and A\_userName represents the userName of

the admin that logged it into the system

Adds:

* String, purchaseID → String, purchaseID
* This is the table for the relation between a purchase and the item that was bought. Both attributes here are foreign keys referencing the Item and the purchase (the actual transaction) respectively.

Purchase\_Makes:

* purchaseID → amount, item
* From the purchase ID, we can determine the item(s) purchased and the amount spent, as the purchaseID is unique to the purchase

Shift\_LogDelete:

* startTime, length, letter → startTime, length, letter
* The shift is identified by the startTime, letter and length, which are also the only attributes in the table that are not foreign keys

Works:

* startTime, length, letter, userName → startTime, length, userName, letter
* Represents: relationship between a shift and a volunteer; all attributes in this table are part of the primary key and reference other objects