Jason Lim

Kimberly Tom

Team 16

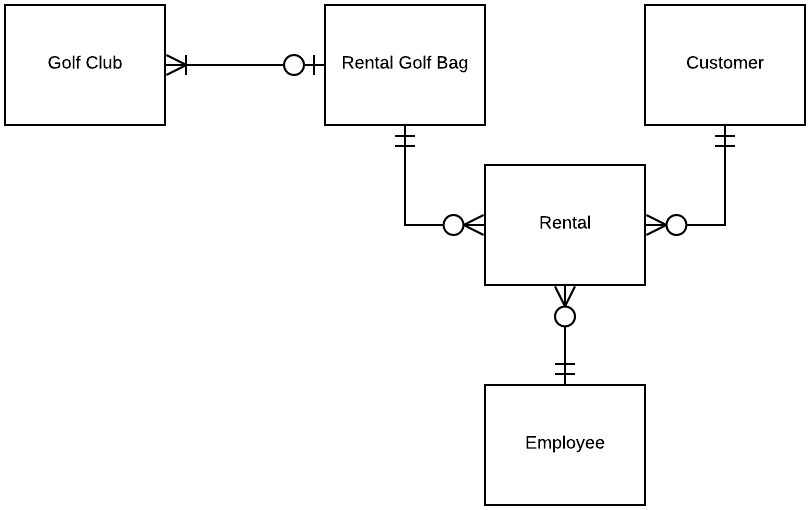
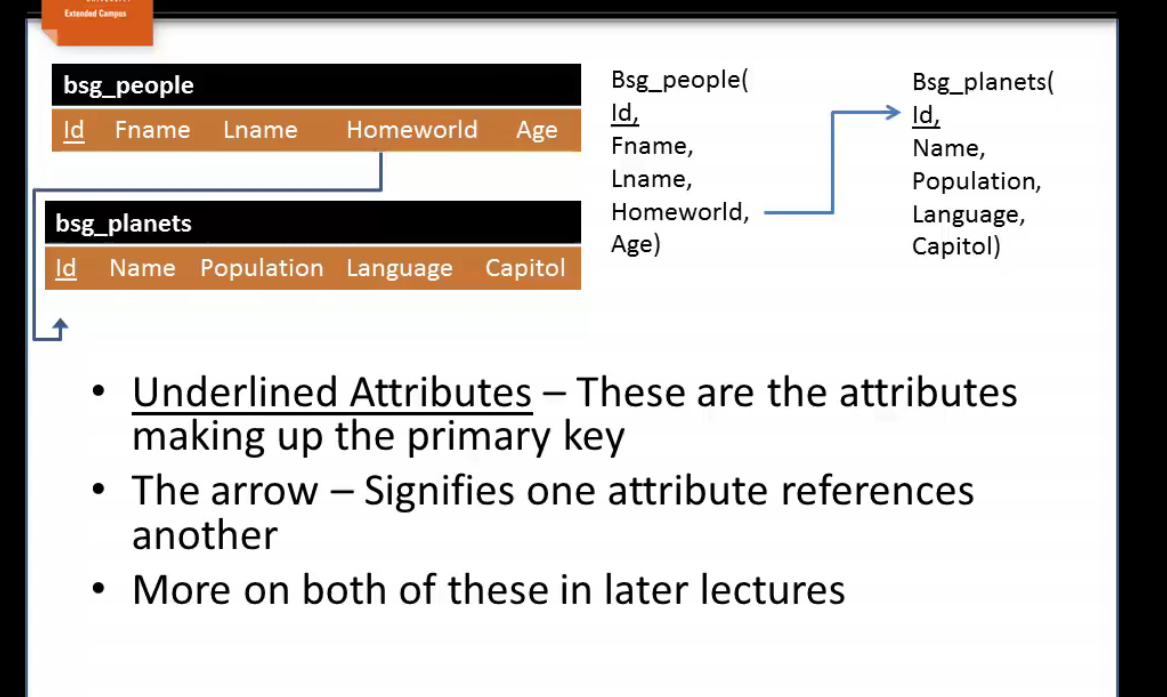
4/29/19

Project Step 2 FINAL: Draft ER Diagram and Schema

**Feedback by the peer reviewer**

Include verbatim the review that your Draft submission received in the Group Review. If you did not receive any reviews, mention that. You would include feedback that both members of the Project Group received.

[**Kin-Ho Lam**](https://piazza.com/class/jtkmn9se33l6cg?cid=58)

1. Please provide your Project Outline and database outline (submitted in step 1), this draft lacks context so it's hard to give feedback on the database. I understand this is may be a placeholder submission?
2. Your Entity-Relationship Diagram does not need verbal descriptions (buys a, goes to, has a, works on) so you can remove them. The ERD crows-feet notation as shown in Week 2's learning doesn't have this.   
   (See example from Week 2's lesson)  
     
   
3. In this case your schema should not use crows-feet notation. You should indicate what columns are primary keys by underlining the appropriate text. (See screenshot, Week 3 Lesson)  
   

Kelly Usenko

Hey Kimberly, very cool topic! Interesting youtube video about these supermarkets.

Looking at the schema and entity relationship diagrams, I think you may need a few more tables in the schema. In the Learn Week 3 module there is a section about many-to-many that basically says for many-to-many you almost always have to add in a relationship table. So products and customers relationship, products and workers, products and locations need a way to connect the dots (via a table with both relevant id keys on it).

The aisle attribute for worker was something that confused me at first for how it would connect to the products or what the idea is for it. For the worker to product relationship, if that should be many to many it would probably be good to have a table that links worker id with product id, and then the aisle id could maybe indicate which aisle the worker is currently in? I could totally just have something not clicking for that part so let me know if that’s not the intention!

For my group’s step 2 we ended up changing some of our wording of what the relationships were when we had many-to-many but thought about it, and a one-to-many or something like that made more sense, so that could also be a way to clarify the relationships too depending on what you are going for.

Last note is something I noticed because I did it wrong and my partner pointed it out (nicely! lol) I also put arrows on each end of my lines when we were making our schema diagrams. He let me know, and it is also in the Week 3 learn section if you scroll to some of the schema images, that the line starts at a foreign key and the point end goes to the primary key.

Let me know if I am misinterpreting and what you think. Again very fun idea and topic.

Ryan Trapp

Hi Kimberly,

I don't have a whole lot to add to what Kelly already pointed out. Seems like the only area that needs to be changed is the schema including adding the relationship tables and making the arros directional instead of bi-directional.

I'm nit-picking at this point but maybe indenting the bullets for readability would help as well but that's all that I can come up with.

#### **Actions based on the feedback**

List briefly the actions that you chose to take based on the above feedback. If you decided not to act on a specific suggestion, you need to describe in detail your reasoning.

We had fixed our ER and Schema diagram, Kin Ho had graded and reviewed our project step 2 early so we had already updated the work for these diagrams. The two reviews by Kelly and Ryan were done after the ER and Schema diagram were fixed and the Project Step 2 updated with the Project outline.

For Kelly’s comments we realized that perhaps many of our relationships may not be many to many. If there was a many to many relationship, we added a relationship table. We removed the relationship between workers and customers because we did not see that as necessary for our project. We also removed customers and location because this did not add value to what we think would be relevant data points.

From Ryan’s comments, we decided to make the arrows directional instead of bi-directional for the keys relationship. We also needed to create the relationship tables.

#### **Upgrades to the Draft version**

If you are making any changes to the files based on your own changed design decisions, they should be listed under this section.

We decided to implement “section” meaning section of the supermarket such as the produce section, dairy section, etc.

Removed paid variable from customer entity

We edited some of the relationships because some of the relationships were unnecessary or did not make sense.

Removed shipments from supplier and aisle from products entity

Updated variable names

We reimagined the entire schema to see through the lens of broad overview of the supermarket instead of getting in the nitty gritty details of the day to day.

**a) Fixes based on Feedback from Step 1:**

This section in your PDF should detail what all things you were told to fix in any feedback by the grader on Step 1 and how you fixed them. If you chose not to fix things based on the feedback, this would contain why you think the design should stay the same. Apart from this if you are making any changes based on your changed design decisions, they should also be listed here.

If you haven't received any feedback or grades yet, you can exclude this section in the Draft but would include it in the Final version.

We only got feedback stating Great job.

**b) Project Outline and Database Outline - Updated Version:**

This section should contain the updated version of your submission for Project Step 1 based on the feedback by the grader as well as any design decisions that you decided to make on your own.

If you haven't received any feedback or grades yet, you can exclude this section in the Draft but would include it in the Final version.

Only received feedback stating great job.

**Team 16: Tonkatsu**

Jason Lim

Kimberly Tom

CS 340

Project Step 1

a) Supermarket Database Outline

Many supermarkets need core processes and the ability to improve business performance whether it be increasing sales or decreasing costs in turnover of inventory. In case study of Tesco, one of the UK’s premium supermarket stores, technological adoption in big data allowed Tesco to successfully manage the transition of its multinational store from brick and mortar into online retailing.

Supermarkets like Tesco face many challenges such as lowering the cost of food waste, changing customer behavior and also making sure workers perform to against up and coming business competitors. The solution is to use new tech and quality data in order to use big data for analytics to model trends in customer behavior and forecast supply as well as improve overall worker performance. Supermarkets in general can use databases to leverage how individuals shop for products and the companies can buy products to make sure that they will be in stock at the right time at the right equilibrium. To take it even further, sensor data and hardware is used to monitor supermarkets items as well as container fridges and utilize predictive analytics, AI, and machine learning to automate temperature allocation.

In our project, we will create a comprehensive framework for the structure that defines the supermarket business through database management. We will look at the supply chain, customer demand, workers performance, as well as the sale items as the entities and major focus of our database.

Image Source: 1)<https://bit.ly/2Gg1dzP>, 2) https://bit.ly/2D4kXnW

Works Cited:

Tesco Video:<https://www.youtube.com/watch?v=AiwVhhfaEwM>

Consulting Case for Tesco:<https://www.bernardmarr.com/default.asp?contentID=687>

Trends in Supermarkets:<https://www.bptrends.com/publicationfiles/01-03-2012-ART-Supermarket%20Article-steeneken-Ackley%20111226.pdf>

### **b) Database Outline**

The entities in our Supermarket database are:

1) **customers** -- Customer participates with the workers as well as the items in the supermarket and the section of the supermarket.

It has the follow attributes:

* **cid**: Customer ID will be assigned to each customer and then recorded into the database. This will be auto incrementing number for each customer that we record. data type: int; constraint: 8 digits
* **cFirstName**: first name of the customer which is a string of maximum 50 characters. It

cannot be blank and no default name. Data Type: String; constraint: 50 chars

* **cLastName**: last name of the customer which is a string of maximum 50 characters. It

cannot be blank and no default name. Data Type: String; constraint: 50 chars

* **gender**: string value that can only be of two values: male or female

It cannot be blank and the default value is female Data Type: Boolean; Constraint: M/F

* **hometown**: Where the customer is located in, can be obtained from credit card information or transactional data. Must be filled as each customer should have a hometown/city they are from. There is no default hometown. This will contain the id of the city to which this customer belongs. A character cannot belong to a city which is not in our database. Data Type; String; constraint: 50 chars

2) **products**-- The products will interact with all the other entities such as the section where the product is located, the customer who buys the item from our supermarket, and the workers who restock the items onto the shelves and price the items accordingly.

* **pid**: Items ID will be assigned to each item and then recorded into the database. This will be auto incrementing number for each item and item type; data type: int; constraint: 8 digits
* **pName**: name of item which is a string of maximum 50 characters. It cannot be blank and no default name. Examples include “Frosted Flakes Cereal” or “Marshmallow Cookies”. Data Type: string; constraint: 50 characters
* **price**: we will keep our prices with 2 decimal places and a whole number. Each price will be attached to the specific item ID. Data Type: Float to nearest 2 decimal places; Constraint: 0 to $100
* **pSection**: section for item which is a string of maximum 50 characters. It cannot be blank and no default name. Examples of this could be fruits, vegetables, bakery goods, non-perishables, or household goods; Data Type: string; constraint: 50 characters
* **quantity**: quantity value of each item attached to the item id of how much we have in the supermarket. Integer value. Data type: int; constraint: 0 min 9999 max

3) **workers** -- The workers will interact with all the other entities such as the section, products, and customers. Workers will assist in stocking products, working at the sections, and assisting customers as a cashier.

* **wid:** Worker ID will be assigned to each worker and then recorded into the database. This will be auto incrementing number for each worker; Data type: int; constraint: 8 digits
* **wFirstName**: first name of worker, cannot be blank. Data type: string; Constraint: 50 characters
* **wLastName**: last name of worker, cannot be blank. Data type: string; Constraint: 50 characters
* **job**: Job that the worker has we will have stocker, cashier, and customer service specialist. It cannot be blank and no default name.
* **email**: email of worker cannot be blank Data type: string; Constraint: 50 characters
* **birthday**: birthday of worker, must be above 18 to work at this supermarket (constraint) Data Type: Data, constraint: month, day, year format
* **location:** the name of the location the worker works at. Data type: string; Constraint: 50 characters

4) **section** -- The supermarket will have sections such as the grocery section or the cereal section that each of the other entities will be interacting with such as the customers, products, and the workers.

* **sid** :section ID will be assigned to each section and then recorded into the database. This will be auto incrementing number for each section. Data Type: int;, constraint: up to 8 digits
* **name**: name of section which is a string of up to 50 chars and cannot be blank. Data type: String

5) **location** -The supermarket will have locations that each of the other entities will be interacting with such as the customers, products, and the workers.

* l **id** :location ID will be assigned to each location and then recorded into the database. This will be auto incrementing number for each location. Data Type: int;, constraint: up to 8 digits
* **city**: city of location which is a string of up to 50 chars and cannot be blank. Data type: String

The relationships in our Supermarket database are:

* **customers buy products** – A customer can buy 0 to many products as long as supply is there for the product and a product can have 0 to many customers so this is a many to many relationship. Each product can be part of an order for many different customers.
* **products are stocked in sections** – Multiple products can be in each section. A certain product can only be in one section. Therefore this is a one to many relationship between products and section
* **workers stock sections** – Workers to sections is a 1 to many relationship since each worker can work one section but a section can have many workers because of different locations. Therefore each section has multiple workers and each worker is assigned one section.
* **workers and locations** – Workers is a 1 to many relationship since each worker can only work in one location at a time and each location can have multiple workers at one time working.

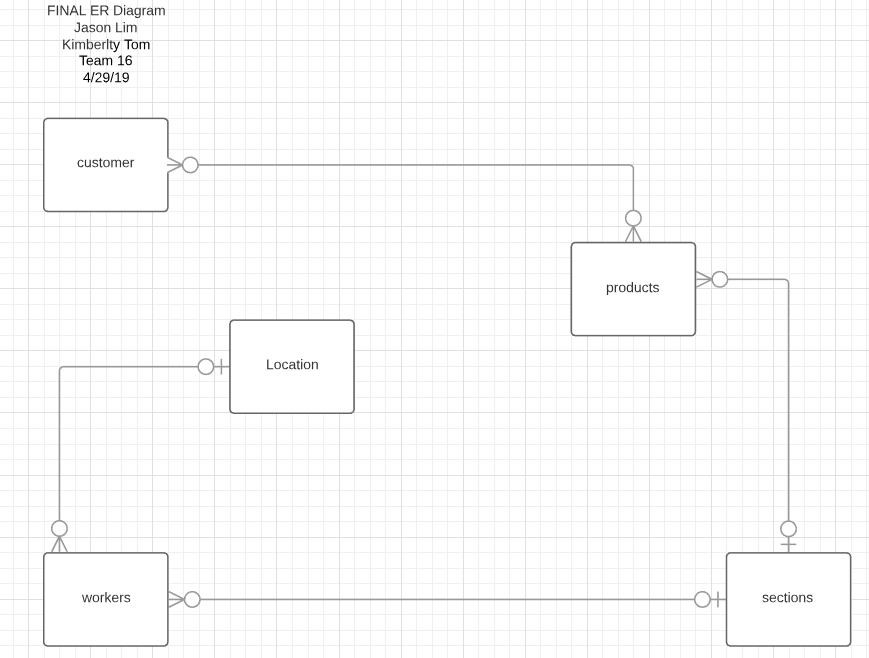
There can be 0 to a finite number of customers, items, workers, and sections per quantity.

We can use the built in dbms domains for the data types such as time, date, boolean, int, char for our project.

**c) Entity-Relationship Diagram:**

This should be an ER diagram that matches your database outline. Anything that does not match the Database outline or uses incorrect notation will cause a reduction in your points in the Final Version of this assignment.

You may use the tools listed on [Tools for this course](https://oregonstate.instructure.com/courses/1746998/assignments/syllabus#tools), or draw by hand and upload a scanned legible copy. Please stick to the notation from Week 2.



**d) Schema:**

This schema should follow the database outline and the ER diagram exactly. It will be graded in the Final Version for this Step on the extent with which it matches the database outline, with an emphasis on whether the relationships, tables, and keys are set up correctly. You may use the tools listed on [Tools for this course](https://oregonstate.instructure.com/courses/1746998/assignments/syllabus#tools), or draw by hand and upload a scanned legible copy. Please stick to the notation from Week 2.

