Project Step 4 Draft Version: DML and DDL Queries (Group on Ed Discussion)

URL: http://web.engr.oregonstate.edu/~schulbra/projectgroup85_Step3_Draft/index.html

1. **Names**: Brandon Schultz, Robert Collins

   Project Title: Pharma-Mate: A Precursor to an Automated Pharmacy Experience

Feedback by the peer reviewer:

Mason Stiller 3 days ago

Hi Team RCBS,

Overall I thought your user interface was aesthetically pleasing.

- ***Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.**

Every table is represented in the UI but not every table can utilize a SELECT.

- ***Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?***

Yes there are two tables that are able to be searched. Both the medication side effects and the diseases treated tables have a search function. While they weren't populated when I loaded the page I inferred that they will be populated in the final product.

- ***Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.**

The two tables that have search functionality do not have an insert function.

- ***Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).**

Yes. There is a many to many relationship between Customers and Medication.

- ***Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.**

Several pages are labeled as Add/ Delete but there is no Delete button. The labeling of these pages suggests that a delete button will be added later.

- ***Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?**

I think that every table that is labeled add/delete has an update function. The tables labeled search have no update functionality.

- ***Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.**

Yes. I believe you can add a customer without having to specify what medications they are connected to.

- ***Do you have any other suggestions for the team to help with their HTML UI?***

Overall, I think the UI looks really nice. Adding update and delete functionalities to all table could help the UI fully meet the criteria.

M Morgen Mhike 4 days ago

Hi Brandon and Robert,

Good work on the HTML UI, here is my feedback on your work and i hope you find it helpful.

- ***Does the UI utilize a SELECT for every table in the schema?*** **In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.**

The diseases treated and Side Effects are two separate entities in your schema but you combined them into one page in the HTML UI. I think it will be better to keep them separate in the HTML UI as you did in the schema so people can easily navigate to it.

- ***Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?***

This was missing, there was no dynamically populated list of properties. One recommendation will be to do this on the Diseases treated table and dynamically populate the diseases for which you commonly treat and then maybe an "other" option in case someone does not see the choice on the list

- ***Does the UI implement an INSERT for every table in the schema?*** **In other words, there should be UI input fields that correspond to each table and attribute in that table.**

There is no INSERT for the DiseasesTreated and SideEffects page

- ***Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?*** **In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).**

No issues here

- ***Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?*** **In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.**

The customer, customers order, Provider information all have either Add/Delete capability but uses one submit button. I think having a delete and add as separate buttons will be better.

- ***Is there at least one UPDATE for any one entity?*** **In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?**

There is an alter DB function which provides the update functionality

- ***Is at least one relationship NULLable?*** **In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.**

Yes

- ***Do you have any other suggestions for the team to help with their HTML UI?***

D Dylan Bodvig 4 days ago

- *Does the UI utilize a SELECT for every table in the schema?*
  - **There is no SELECT functionality for every table in the schema. I can not select certain customerOrders as an example. I believe all of the tables do not posses a SELECT functionality as of now. I can only alter the DB. So an update is what is being done here.**
- *Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?*
  - **Yes, on the page Disease Treated & Side Effects allows for the user to search using a SELECT on a list that is dynamically populated.**
- *Does the UI implement an INSERT for every table in the schema?*
  - **There is not an insert for every table. There is only an insert for Customers and Inventory Tables. They also have SELECT for only a few tables. They have stated that they plan to add the INSERT, UPDATE, DELETE functions in the future.**
- *Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?*
  - **The INSERTS correctly add the FK attributes that are needed to make sure the tables stay connected. There is also at least one M:M relationship that is shown.**
- *Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?*
  - **There is no DELETE function on any tables right now. As stated in my notes, I saw that they are aware of the problem and will have it fixed as soon as possible.**
- *Is there at least one UPDATE for any one entity?*
  - **They also have stated that they need to add the UPDATE functionality.**
- *Is at least one relationship NULLable?*
  - **They have a Nullable relationship in the customerOrder Page. For example, when searching for a Disease you can leave the Secondary Side Effect NULL and you will be able to search for a disease treated by a certain medicine.**
- *Do you have any other suggestions for the team to help with their HTML UI?*
  - **The UI is pretty well laid out. It really is not too fancy, but also makes sure to get across all of the information that is needed. I would recommend trying to get all of the different tabs in one row. This would give off a little bit cleaner look. Other than that great job!**
- *NOTES:*
  - **On the website they have stated they plan to create the SELECT, DELETE, and UPDATE on most of the tables, so they will fix a lot of issues that were brought up in the review.**

Reply •••

M Martin Gastelum-Valenzuela 5 days ago

- ***Does the UI utilize a SELECT for every table in the schema?* In other words, data from each table in the schema should be displayed on the UI. Note: it is generally not acceptable for just a single query to join all tables and displays them.**
    - Currently there does not seem to be functionality to SELECT for every table within the outline. For example I cant SELECT for a Customer table.
- ***Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?***
    - Yes, In this version I can search for Medicine side effects and diseases treated by medicine.
- ***Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table.**
    - No it does not appear that the user can INSERT for every table in the schema. Currently I only see functionality to INSERT into Customers and Inventory tables.
- ***Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total).**
    - Yes each of the INSERTs successfully includes addition of the FK attributes.
- ***Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?* In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers.**
    - In the current version there is no functionality to DELETE. However It is stated in the website they are aware of this and plan to add this in future versions.
- ***Is there at least one UPDATE for any one entity?* In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record?**
    - Similar to the DELETE buttons the authors of this project plan to add UPDATE functionality in future versions.
- ***Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus it should be feasible to edit an Order and change the value of Employee to be empty.**
    - Yes, I believe the composite tables accomplish this.
- ***Do you have any other suggestions for the team to help with their HTML UI?***
    - I really enjoyed the theme of your website and think youre off to a great start. I think it can get a little disorienting navigating sometimes due to the 'Index' page always being highlighted in the navigation bar. I think updating that so the current page is highlighted will greatly help orient the user

Reply ···

Actions and Upgrades based on peer feedback from 4-29-21:

- Implemented fixes mentioned above, those being:
    - Made various changes to entity names and properties to better align with PowerPoint provided as feedback for "Project Step 2 Final Version: ERD & Schema (Group / On Canvas)" assignment.
        - Customer was renamed Customers.
            - IdNum was renamed to customerID.
            - Removed CustomerMedication property from Customer entity.
            - Changed FirstName and LastName properties to fname, lname.
        - Replaced CustomerRecords entity with CustomerRecords entity.
        - Replaced MedicationProviderInformation entity with Medictions entity.
        - Changed property names from DiseasesTreated and SideEffects entities to better reflect their functionality and make using the db easier by users.
            - DiseaseTreatedTableIDNum is now diseaseID.
            - MedicationName is now medID.
            - Primary Disease is now disease.
            - SideEffectsTableIDNum is now effectID.
            - MedicationName is now medID.
            - PrimarySideEffect is now sideEffect.
            - Removed SecondaryDisease and SecondarySideEffects properties.
        - InventoryMedicationItem was renamed to Inventories.
            - InventoryTableIDNum property is now inventoryID.
            - InventoryMedicationName property is now medID.
            - CustomerRecordsMedName is now stock.
- Attempted, but failed to add buttons allowing for modifications of rows that will be added successfully before final submission.
    - Waited too long to add these row by row for each tables various properties and will figure out a way to not break the projects html interface before final submission with their inclusion.
- Reduced number of pages on site to make navigation and appearance better by adding Customers and Customers Records entities to one page titled Customer Info
- Updated sites current buttons, and tables to reflect changes in entity and property names.
- Updated ER diagram and schema to reflect changes mentioned above.
- Updated table terminology on website to reflect changes above.

a) Project Outline and Database Outline, ERD and Schema Updated Version

Overview:

Historically, employee efficiency has increased dramatically in response to the addition of new technologies. Computers have revolutionized the workforce of the last 50 years and will continue to do so as progress in fields like automation is made. Ideally, automation of tasks traditionally completed by employees results in more effective work being completed to a degree of quality not previously possible. This is not always the case, as made evident by the existence of self-checkout lanes at nearly all grocery stores. Eventually, we will be able to complete a purchase without requiring the assistance of an in-person employee and no longer will their presence be required, nor will their job need to exist.

Pharmacies and the employees that work in them appear vulnerable to a similar, automated model. In their current state, pharmacies exist primarily as a means of supplying some item(s) to a customer, much like a grocery store. Pharmacy Techs exist in a role like that of a cashier, however, pharmacists and the medicines they manage resemble a more complex system of tasks. Our project will explore the feasibility of mimicking the role and knowledge of a pharmacist by attempting to manage and link an individual customer to their prescribed medicines and associated documentation through the creation of a web app containing a database backend.

The project aims to resemble the functionality of a mobile app, in that one user would be associated with one “Customer” entity, and thus also be associated with a single “CustomerRecords” entity.

The “Medications” entity represents the initial interaction between an individual customer’s prescription and the worker tasked with determining if said prescriptions requirements can be met and consequentially filled, then provided to the customer/user. It is a branching point for the unique specifications regarding the customer/user’s order, as it’s PK value will be the name of one of ten medicines that is used by “DiseasesTreated”, “SideEffects”, and “Inventories” entities.

“DiseasesTreated” and “SideEffects” will consist of the most common ailments treated and potential side effects caused by one of the ten medicines depicted by the “medName” property value of the “Medications” entity. “Inventories” will use the “medID” property value of the “Medications” entity to determine if the users medicine is in stock or carried by the database and associate a medicine name to a medicine ID number. “Inventories” serves as the gross inventory of medicines that a pharmacy would contain and will contain information that associates a user/customer’s medicine to a medicine in the database.

Database Outline, in Words:

Entity #1:

- **Customers**: Keeps track of various customer values and associated medications. A customer has a unique ID, a first and last name, and medication prescribed to them. SELECT, DELETE and UPDATE functions are planned to be used to change CustomerMedication and to remove customers from DB if medicine fails them. IdNum is used as PK to implement, and link Customer and CustomerOrder.
  - **Properties**:
    - **PK = *customerID**: int, auto_increment, unique, not NULL. Tables ID number property.
    - **fname**: Customers first name. Uses varchar, not NULL.
    - **lname**: Uses varchar, not NULL. Customers last name.

    - **Relationships**: M:1 to **CustomerRecords**

Entity #2:

- **CustomerRecords:** Information linking customers idNum to Medication entity. Contains information regarding individual, customers prescription.
    - **Properties**:
        - **PK = *recordID:** int, auto_increment, unique, not NULL Tables idNum property.
        - **FK = customersID:** int, auto_increment, unique, not NULL. FK connecting customerID property to CustomerRecords.
        - **FK2 = medID**: int, auto_increment, unique, not NULL. FK connecting to composite, Medication's entity.
        - **prescriptionCount**: varchar(255), not NULL. Number of medicines prescribed to patient, unique to prescribed medicine of customer.
        - **prescriptionDosage:** varchar(255), not NULL. Customers prescribed medicine's specific dosing criteria.
    - **Relationships**: 1:M to **Customers, Medications**

Entity #3:

- **Medications**: Prescribing information for medicines in inventory.
    - **Properties**:
        - **PK = *medID**: int, auto_increment, unique, not NULL. CustomerRecords, Inventories, DiseasesTreated, and SideEffects all are linked to Medications table via this property.
        - **medName:** Uses varchar(255), not NULL. Name of medicine used as composite to form M:M between CustomerRecords and DiseasesTreated, SideEffects and inventories entities.
    - **Relationships**:
        - M:1 to **CustomerRecords, DiseasesTreated, SideEffects, Inventories**

Entity #4:

- **DiseasesTreated:** Diseases treated by medicine.
    - **Properties:**
        - **PK = *diseaseID:** int, auto_increment, unique, not NULL. Tables ID num.
        - **FK= medID: :** int, not NULL. . ID of medicine used to associate multiple diseases that can be treated using a single medicine.
        - **disease**: varchar(255), not NULL. The most common illness treated by medication is returned.
    - **Relationships**: 1:M to **Medications**

Entity #5:

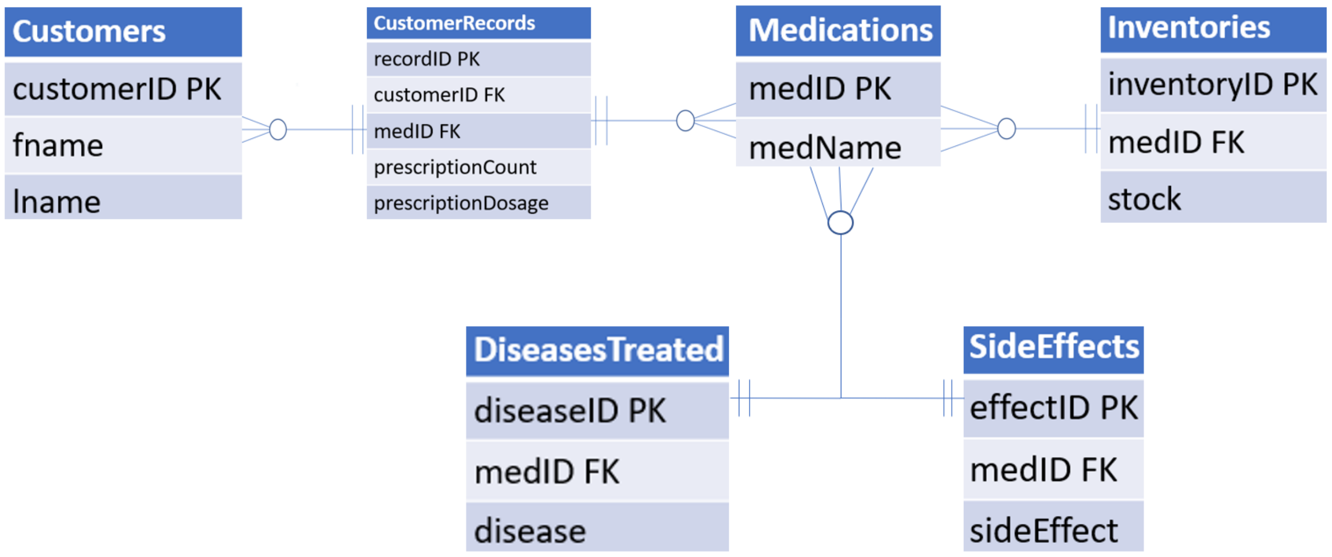
- **SideEffects:** Side effects of prescribed medicine
    - **Properties:**
        - **PK = * effectID:** int, auto_increment, unique, not NULL. Tables ID num.
        - **FK = medID:** int, not NULL. . ID of medicine used to associate multiple side effects to a single medicine.

        - **sideEffect:** varchar(255), NOT NULL**.** Most common side effect of medication is returned via a string of chars.
    - **Relationships**: : 1:M to **Medications**

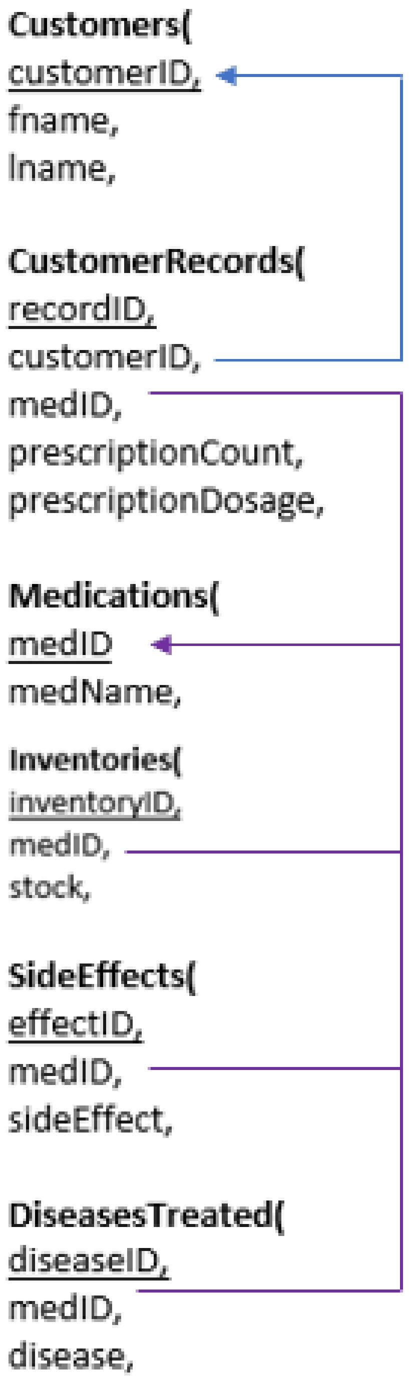
Entity #6:

- **Inventories**: Keeps track of medicine inventory available for dispensing to patients.
    - **Properties**:
        - **PK = *inventoryID**: int, auto_increment, unique, not NULL. ID num that distinguishes each medication in inventory.
        - **FK = medID:** varchar, not NULL. ID of medication stocked.
        - **stock:** varchar(255), NOT NULL**.**. Value of medicine held in stock.
    - **Relationships**: : 1:M to **Medications**

Entity-Relationship Diagram:


Customers
customerID PK
fname
lname
CustomerRecords
recordID PK
customerID FK
medID FK
prescriptionCount
prescriptionDosage
Medications
medID PK
medName
Inventories
inventoryID PK
medID FK
stock
DiseasesTreated
diseaseID PK
medID FK
disease
SideEffects
effectID PK
medID FK
sideEffect

Schema:



Customers(
customerID,
fname,
lname,

CustomerRecords(
recordID,
customerID,
medID,
prescriptionCount,
prescriptionDosage,

Medications(
medID
medName,

Inventories(
inventoryID,
medID,
stock,

SideEffects(
effectID,
medID,
sideEffect,

DiseasesTreated(
diseaseID,
medID,
disease,

b) Fixes based on Feedback from Previous Steps:

Actions based on feedback from Step 1:

- Added additional relationships for all entities.
- Added prior and new relationships between entities as bullet points.
- Removed prefixes from most entity property names.
- Changed various entity property names to better represent their purpose.
- Added plans/details regarding composite table specs.
- Added foreign key details to all entity descriptions.
- Fixed typos and removed unnecessary descriptions of entities and their properties.
- Added CustomerOrderR entity to serve as composite between Customer and Medications entities.
- Added Dosage entity to link and potentially act as composite between Inventories and Medications entities.
- Added Manufacture entity. In hindsight this probably was not needed. Initial project plan was very ambitious, and this entity is an artifact that was added when we first started making changes in order to have a more realistic project idea.
- Switched order of entities in project outline in order to accommodate new entities.
- Changed/removed properties from various entities to make project data more manageable.
- MedicationCustomerInformation entity now serves as summary of customer prescription info instead of a dosing-inventory mess.
- Medications is now used as primary method for linking customers prescription names to medicines in inventory, as well as still containing information about medicines effects and diseases treated.

Actions based on Peer feedback/ Step 2:

- Provided specific facts in overview regarding problem solved by website with db backend.
- Included specific values regarding number of prescriptions/medicines and customers website will accommodate.
- Removed now irrelevant information from overview.
- Added clarification on PK and FK attributes found in various entities. In addition to “*” indicating a PK, we’ve also included “PK = … “to better highlight attributes

that are used as primary keys. FK values now include additional context and have been formatted to make distinguishing them easier. Also, all entity keys have been arranged in DB outline to reflect their placement in the ER diagram and schema.

- Changed naming of some attributes to better reflect functionality of entity in context of problem being solved by website with db backend. Many names contained carryover from prior, more ambitious plans for our project and have been renamed to only contain necessary information.
- Left 0/1:M to “other entity “in descriptions of relationships amongst entities in DB outline.
- Kept “MedicationCustomerInformation” entity. Its purpose was to mimic the documentation one receives with most over the counter medicines. That documentation is specific to the customer, their condition and the medicine being received to treat their ailment. Although this entity will not do all of that, it will provide a centralized table of info specific to the customer/ user of website based off of info from the other, applicable tables in db.

Upgrades to the Draft based on Peer feedback/ Step 2:

- Split MedicalProviderInformation entity in order to better normalize database.
    - Originally contained information regarding medicine name of customers prescription, diseases medicine treated and side effects of medicine. Added separate “DiseasesTreated” and “SideEffects” entities, both of which are former properties from the original MedicalProviderInformation entity.
- Removed “Manufacture” entity.
    - Was a filler table that served no real purpose for scope of project.
- Removed entity properties that were unnecessary and/or complicated normalization of database.
    - InventoryStockCount from Inventories
    - MedCustomerRefill from MedicationCustomerInformation
- Added DosageTableIDNum property to dosage entity to act as fk linking it and InventoryTableIDNum property of Inventories entity.

- Adjusted entity numbers to accommodate addition of “DiseasesTreated” and “SideEffects” entities.
- Updated entity relationship diagram and schema to reflect addition of “DiseasesTreated” and “SideEffects” entities.

Actions and Upgrades based on feedback from 4-22-21:

- Removed Dosage and MedicationCustomerInformation entities.
    - These were causing unnecessary work and complications for our project.
- Added CustomeraOrderMedName property to Inventories entity. This will act as an fk linking it to CustomerOrderTableIDNum property of the CustomerOrder entity.
- We did get a comment on “Project Step 2 Final Version: ERD & Schema” assignment suggesting some alterations that weren’t able to be implemented in time for submission of this assignment, as we only had a day to make those changes and due to scheduling, most of the submission for this week had already been worked on.
    - Many of those changes focused on changing entity/property names. As of now the names are unchanged but will be adjusted for readability. The HTML site currently masks some of these naming complications, but this wont be the case for the final project as we plan to create something more similar to what was provided in the attached database example pptx.
    - Removal of the Dosage and MedicationCustomerInformation entities was partially inspired by that feedback.

Actions and Upgrades based on feedback from 5-6-21:

- Added forms for project requirements to website.
- Combined Medications and Inventory Pages.
    - Medications table is composite for m:m relationship and page looked awkward by itself.
        - Site still contains link to inventory page in navbar but wont for final version.
- Completed data definition queries for generating database.
    - This includes insert queries for sample data.
- Completed draft of Data Manipulation Queries for database.