Ryan Pate
Nathan Raymon
5/2/2020

Project Step 3 Final Version

URL:http://flip.engr.oregonstate.edu:6187/

Project Outline and Database Outline - Updated Version:

Overview

We are going to make a simple town simulator for use in a tabletop roleplaying game. Players of tabletop games often have to design their own towns and handle which NPCs have access to different goods and quests themselves. Our system will make this easy by simulating the town and the goods and services offered in it so that players can interact directly with the town. Our system will be designed to handle only a few interactions occasionally since it is designed to be used as a gameplay tool. Our system will help players navigate through the town by selecting shop locations to visit. At each shop the players will be able to interact with NPCs who will give them quests and shopkeepers which will allow them to purchase goods. Our system will NOT process orders or modify player inventory. It is expected that our tool will be used by a game master as a part of a larger game such as Dungeons & Dragons or Pathfinder. So, we expect the players and game master to handle adding items to the player inventories and adjusting their money after each purchase. Our system will only display the goods a player decides to purchase as well as the total cost of the sale after the purchase is complete. Orders will not be stored in the database or accessible at all once the player has left the sale screen.

Database Outline

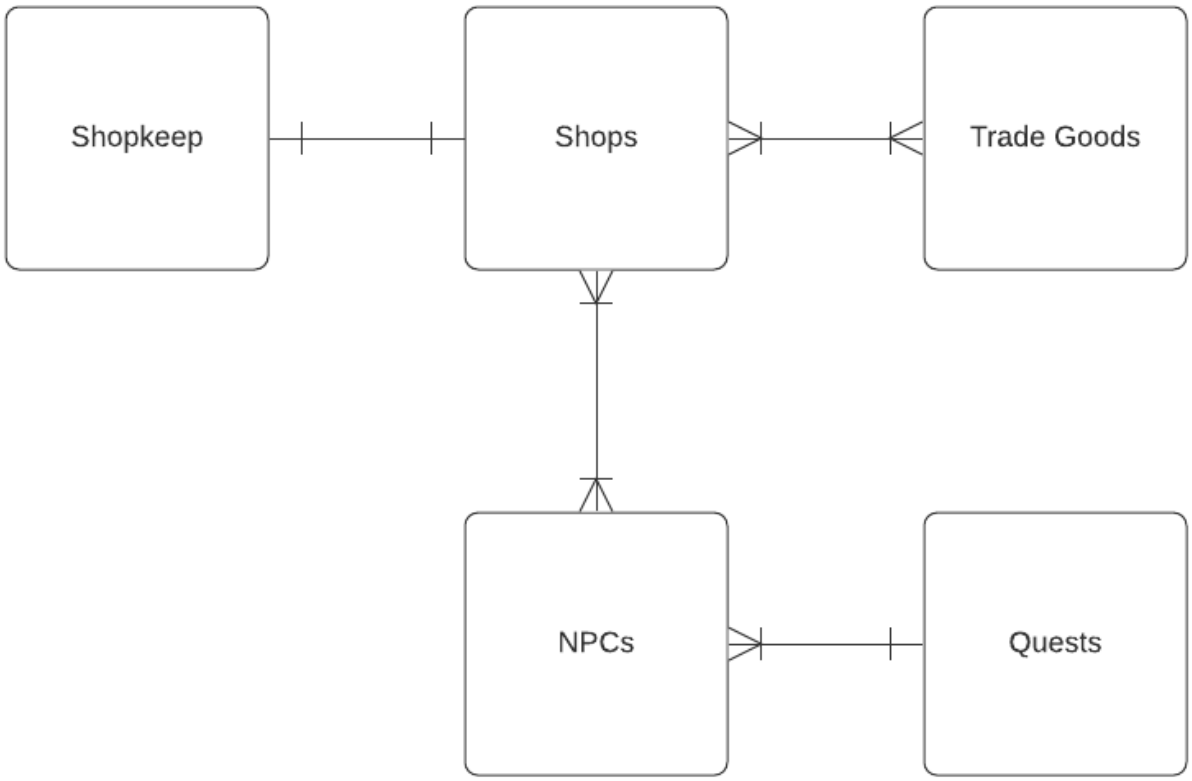
- Shops - Ryan
  Stores the shops that can be visited in the town as well as a name of the shop, a brief description, and a 1:1 relationship with the NPC that runs the shop called the "shopkeep".
  - id: int, auto_increment, unique, not null, primary key
  - shopkeep: varchar, not null
  - shopkeepId: int, not null
  - shopName: varchar, not null
  - shopDescription: varchar, not null
  - Relationships: M:M relationship with NPCs implemented in locations table, M:M relationship with trade goods implemented in Inventories table, 1:1 relationship with shopkeep NPC implemented as shop attribute shopkeepId.
- TradeGoods - Ryan
  Stores all the different trade goods that can be bought at the different shops in town. Each good has a name, value, quantity available, description, and weight.
  - id: int, auto_increment, unique, not null, primary key

    - name: varchar, not null
    - value: int, not null
    - quantity: int, not null
    - description: varchar, not null
    - weight: int, not null
    - Relationships: M:M relationship with Shops implemented in the Inventories table
- Locations: - Ryan
  Stores the relationships between non shopkeep NPCs and shops. Each NPC can be found in a variety of shops so this table stores those M:M relationships. This table will not store the shopkeeps relationship with a shop since that is a 1:1 relationship and will be stored in the Shop entity for convenience.
    - id: int, auto_increment, unique, not null, primary key
    - npcId: int, not null, foreign key specifying the id of the NPC in the NPCs table
    - shopId: int, not null, foreign key specifying the id of the shop in the Shops table
- Inventories - Ryan
  Stores the relationships between trade goods and shops. Each trade good can be bought in a variety of shops around town so this table stores those M:M relationships.
    - id: int, auto_increment, unique, not null, primary key
    - goodId: int, not null, foreign key specifying the id of the good in the TradeGoods table
    - shopId: int, not null, foreign key specifying the id of the shop in the Shops table
- NPCs - Nathan
  This table stores the name and greeting of the NPC when communicating with the user.
    - Many to many relationship with Shops
    - id: int, auto_increment, unique, not NULL
    - name: varchar, not NULL
    - greeting: varchar, not NULL
    - Relationships: a M:M relationship between NPCs and Shops implemented in the Locations table. 1:1 relationship with the shopkeep NPC implemented with shopkeep npcId as a FK inside of Shops under the shopkeepId attribute.
- Quests - Nathan
  This table will hold the name, what the quest will entail (description) and what they reward will be if the user completes the quest.
    - id: int, auto_increment, unique, not NULL
    - name: varchar, not NULL
    - description: varchar, not NULL
    - reward: varchar not, not NULL
    - Relationships: A 1:M relationship between Quests and NPCs is implemented with questId and npcId as a FK inside of npcQuests.
- NpcQuests - Nathan

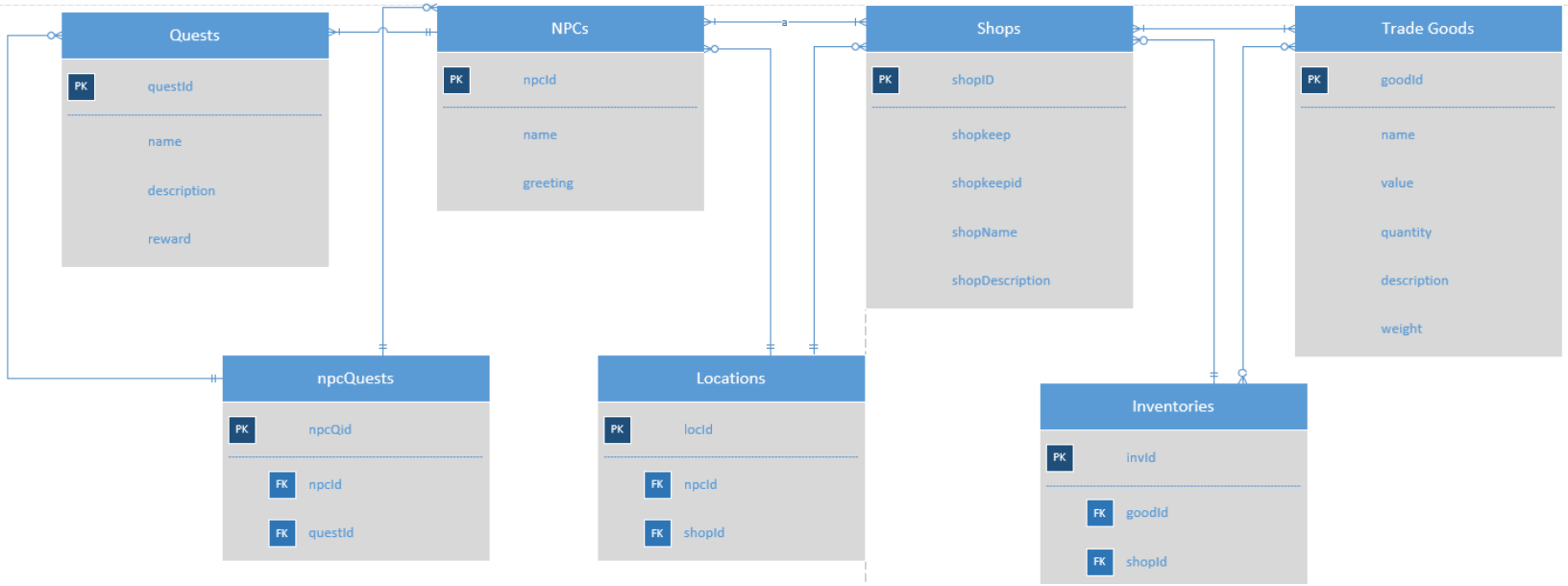
This table holds the ids of NPCs and Quests, correlating to which NPCs are holding which Quests.

- id: int, auto_increment, unique, not NULL
- npcId: int, not null, foreign key specifying the id of the NPC in the NPCs table.
- questId: int, not null, foreign key specifying the id of the quest in the Quest table.

Entity-Relationship Diagram:



Schema



Feedback by the peer reviewer

Review 1:

- **Does the UI utilize a SELECT for every table in the schema?**
    - Most of them. It's not clear to me where the trade goods and locations tables are located.
- **Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?**
    - Yes. They filter by shop in inventory management.
- **Does the UI implement an INSERT for every table in the schema?**
    - Most of them. It's not clear to me where the trade goods and locations tables are located.
- **Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?**
    - I don't see that yet, but then again, this is the first time I am seeing that requirement, so my group doesn't have this satisfied either.
- **Is there at least one DELETE and does at least one DELETE remove things from a M:M relationship?**
    - Yes they are built in as buttons in the rows.
- **Is there at least one UPDATE for any one entity?**
    - Yes they are built in as buttons in the rows.
- **Is at least one relationship NULLable?**
    - I don't see that yet, but then again, this is the first time I am seeing that requirement, so my group doesn't have this satisfied either.
- **Do you have any other suggestions for the team to help with their HTML UI?**
    - Your site is so fun! I like the picture on the front page. When you get the " update" button hooked up to the database, it would be helpful to populate the edit form with placeholder values so that the user doesn't need to retype in the entire entry.

Review 2:

- **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 display them.
    - I noticed Shops and Attractions can be selected from the Main Page, and from the selected Shop, you can then select an NPC
- **Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?**
    - Yes. Selecting a specific ShopKeepId ShopName can be used as search filter
- **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.
    - I see you can add price, Quantity, Description, Name, and Weight for the Inventory

- **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).
    - NPC to a Shop can be SELECTED BY; however, I do not see where you can INSERT
- **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.
    - I see that in the Shop Inventory Management there is the ability to DELETE rows
- **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 see that in the Shop Inventory Management, there is the ability to edit rows
- **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.
    - I do not see this option in your webpage. I understand this requirement is not very clear.
- **Do you have any other suggestions for the team to help with their HTML UI?**
    - I like the project. it is quite creative. Hopefully all the pages are tie together, and the UPDATING/DELETING functionality will allow you to enter data into the DB.

Review 3:

- **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
    - Yes they are all covered on the shop page
- **Does at least one SELECT utilize a search/filter with a dynamically populated list of properties?**
    - I believe this will be handled with the shop inventory management. Selecting a shop will populate the goods
- **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.
    - Yes this is all handled on the ship 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).

    - I'm not sure if the trade goods and shops relationship is working the way it's shown in the ERD. It looks like you create a new item and add it to the shop at the same time rather than adding a trade good and then adding that good to a shop's inventory. The way it is set up it looks like there could be many rows of battle axes in the trade goods table, for example. So the way it is set up right now appears to be more of a 1:M relationship rather than M:M.
- **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.
    - This functionality exists for the shops and trade goods relationship, but as I mentioned above it seems to have been implemented as a 1:M relationship rather than M:M. Unless I am misunderstanding the UI.
- **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?
    - It looks like you can update rows in the shop's inventory
- **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.
    - I don't believe so. I it may make sense to have NPCs that have a nullable quest
- **Do you have any other suggestions for the team to help with their HTML UI?**
    - As mentioned above, I think it would make sense to have a place to add a trade good and then a separate place to add trade goods to a shop's inventory.

Actions based on the feedback

We had some issues with how we set up our shop inventory editing functionality. We intended users to select a shop from the drop down, view and edit the shop's inventory, and be able to add trade goods to the trade good table for the **whole town**. Once a trade good has been added to the town users would then be able to add that trade good to whichever shops they want it to be in. However, our form for this had some missing elements that made the intention of the UI unclear for reviewers so we adjusted some of the naming in the form to make it clearer and added a form to add a good to a shop.

Some reviewers also mentioned that they didn't think we had a nullable relationship in our UI. Our nullable relationship is the M:M relationship between shops and trade goods represented in the inventories table. When a new item is added it has no relationship with any shop. When that good is added to a shop a relationship is created between the shop and item. Then when an item is deleted from a shop, the relationship is removed. So we believe that covers the requirement of a NULLable relationship, because the relationship between an item and a shop can be nullified by deleting that item from the shop. We'd like feedback from our grader if this is

not the case. Since this requirement is covered by our items to shops relationship, the changes described in the previous paragraph make this more clear as well.

Upgrades to the Draft version

The NPC page of our frontend needed a little more work on the design aspect, so we added a picture of a generic NPC, made the text boxes look a little nicer, and moved them around for a more appealing look to the page.

Fixes based on previous feedback

We only received a few pieces of feedback that recommended changes to our project outline during step two of the project. So, we have largely left the outline as it is except for a few changes we made for our own reasons. We did however receive feedback recommending we change some of our naming schemes to better differentiate entities and attributes as well as maintain consistency throughout the document. We ended up going with camel case for our naming scheme with entities being capitalized. We believe this will allow us to effectively differentiate between them. So we went through the outline and adjusted everything to fit that naming scheme. We also received a few bits and pieces of miscellaneous issues with the document like the naming of the entity id attributes which didn’t need to include the actual name of the entity. So we fixed a few issues like that as well.