**Project Report for CMPSC 431W**

**JJNW**

John Grant

Josh Whitelaw

Nicholas Denaro

William Bittner

### 

### 

Table of Contents

[List of Figures iii](#_Toc430255483)

[1. Introduction 1](#_Toc430255484)

[2. Function Definitions 2](#_Toc430255485)

[2.1. Sale Items: 2](#_Toc430255486)

[2.2. Categories 2](#_Toc430255487)

[2.3. Registered Users 3](#_Toc430255488)

[2.4. Suppliers 3](#_Toc430255489)

[2.5. Rating 3](#_Toc430255490)

[2.6. Browsing 3](#_Toc430255491)

[2.7. Searching 3](#_Toc430255492)

[2.8. Sale 4](#_Toc430255493)

[2.9. Biding 4](#_Toc430255494)

[2.10. Order and Sale Report 4](#_Toc430255495)

[2.11. Delivery 4](#_Toc430255496)

[2.12. Shop 4](#_Toc430255497)

[2.13. Wish list 4](#_Toc430255498)

[3. Entity-Relation Diagram 5](#_Toc430255499)

[4. Technology Survey 26](#_Toc430255499)

[5. Conclusion 27](#_Toc430255501)

**List of Figures**

Figure 1 - User and User Transaction 5

Figure 2 - User\_Transaction SQL & Schema 6

Figure 3 - Item 7

Figure 4 - Item SQL & Schema 8

Figure 5 - Category 9

Figure 6 - Category SQL & Schema 10

Figure 7 - Address 11

Figure 8 - Address SQL & Schema 12

Figure 9 - Phone 13

Figure 10 - Phone SQL & Schema 14

Figure 11 - Credit Card 15

Figure 12 - Credit Card SQL & Schema 16

Figure 13 - Supplier and Supplier Transaction 17

Figure 14 - Keyword & Supplier SQL & Schema 18

Figure 15 - Shop 19

Figure 16 - Shop SQL & Schema 20

Figure 17 - Wish list 21

Figure 18 - Wish list SQL & Schema 22

Figure 19- Stocked 23

Figure 20 - Stocked SQL & Schema 24

Figure 21 - Full ER Diagram 25

# 1. Introduction

We will be creating a web application, named maetS, which is used to buy and sell content related to games. The users will be able to buy, sell, and bid on games, merchandise, expansion packs, and user created content. Sellers will be able to set the price of the content they are selling. An effect of this is that some content may be put up for free. The web application will have a front end for users to interact with and will sit on top of a backend that does the processing and connects to the database.

# 2. Function Definitions

## 2.1. Sale Items:

The sale items will be the items that companies or users put up for sale. They will have the option to be either put up for a direct sale or for auction. Items that are being sold directly may be grouped if they are the same, and displayed from a single page. The page will act as an abstraction and define the group of the items, and when purchasing an item from that page it will sell the cheapest available copy.

## 2.2. Categories

The web application will have categories, which will help to organize the wide variety of items that we have to offer. Some examples of categories include:

* Video Games > Strategy > Single Player
* Merchandise > Clothing > Women's > T-shirts
* Merchandise > Collectables > Figurines
* Video Games > User Created Content> Mods > Paid > Skyrim > Weapons > Swords

## 2.3. Registered Users

Users who wish to use this web application will be required to register an account. This account will give them the ability to buy a sell items. Account information will include things such as email address, name, phone number, and credit cards. The users will be able to manage this information and make changes to it if necessary.

## 2.4. Suppliers

Suppliers will need to create an account in order to sell or auction their items. They will be able to view and manage the items they are selling.

## 2.5. Rating

Users will be able to rate each other based on previous sales.

## 2.6. Browsing

Users will be able to browse the content that sellers have put up for sale or auction. Browsing will be done based on the categories that were explained above.

## 2.7. Searching

Users will be able to search for content based on key words and categories.

## 2.8. Sale

Users will be able to buy content that is put up for sale.

## 2.9. Biding

Users will be able to bid on auctions.

## 2.10. Order and Sale Report

Weekly reports will be generated which describe the sales based on the category of the items.

## 2.11. Delivery

The delivery of items will be dependent on the type of content. Digital content will be delivered via email, whereas other content will be shipped from the seller to the buyer.

## 2.12. Shop

Users will be able to view a "shop" that lists all of the items that a seller has up for sale.

## 2.13. Wish list

Users will be able to save items that they want to buy later on their wish list. They will be able to set a ranking on each of the content, describing how much they want it.

# 3. Entity-Relation Diagram

Pictured here is the entity "User" and the relationship "User\_Transaction." The "User" entity represents any individual person who is registered with the site. This entity has all of the required attributes: username, which is a primary key and therefore is underlined, password, email address, name, date of birth, gender, income. The "User\_Transaction" relationship is between two users and one item. Transactions store both the transaction ID and the date of the transaction, which is used for the sales reports. The transaction ID, which is a primary key and therefore is underlined, will be a randomly generated number. As the transaction is between two users and one item, we have three bold arrows coming in to the transaction relationship to conform to the total participation and key constraints.

User

Email

DOB

Gender

Password

Username

Name

**Figure 1 – User and User Transaction**

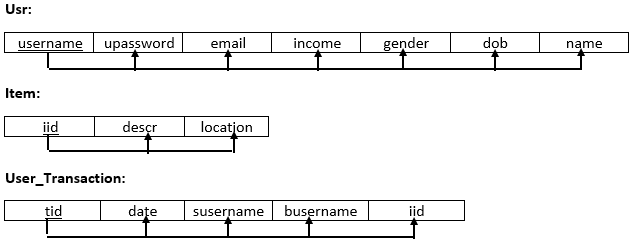
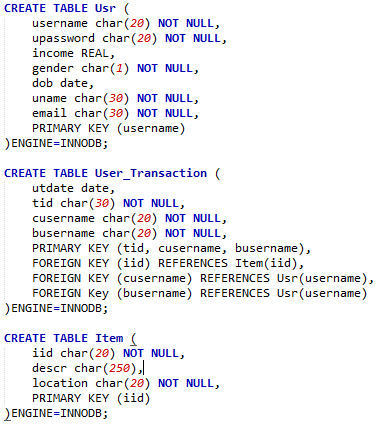
tid

User\_Transaction

Date

Item

Income

 Depicted below are the SQL statements used to represent “Usr” (User), “User\_Transaction”, and “Item” as well as the third normal form diagrams to represent the tables. The necessary conditions are met for the “User\_Transaction” relationship to hold the two users for every one item condition for the transaction by having two foreign keys of separate users and a foreign key for the item ID.

**Figure 2 – User\_Transaction SQL & Schema**

Pictured here are the "Item," "Sale\_Item," "Auction\_Item." The "Item" entity represents a general item that exists in the store. Items have an item id (iid), a description, and a location, which is where the item ships from. The "Sale\_Item" and "Auction\_Item" entities have an ISA relationship with "Item" and therefore inherit an item's description and location and can be used in any relationship that involves an "Item" entity. We have auction items separate from sale items because they require additional information that is not relevant to typical sale items and vice-versa. Sale items have a price attribute whereas auction items have a bid attribute. Auction items also have a bidder and an optional minimum price.

iid

Description

Item

Bid

Location

ISA

ISA

**Figure 3 - Item**

**– Item**

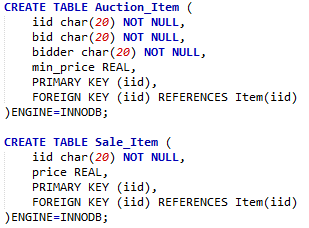
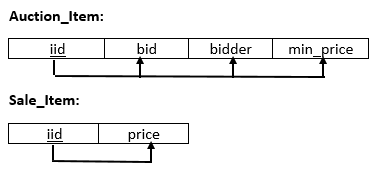
Min\_Price

Bidder

Price

Sale\_Item

Auction\_Item

 Below are the SQL statements for “Auction\_Item” and “Sale\_Item” depicted in the ER Diagram above. It is necessary that “Auction\_Item” and “Sale\_Item” have primary key of “Item” because an item in our web application can either be sold at a fixed price or be auctioned. The third normal form diagrams also represent the entities depicted on Figure 3 as well.

**Figure 4 – Item SQL & Schema**

**– Item**

Shown here are the "Category" entity and the "Categorized" and “Connected\_To” relationships. The entity has a name and id (cid) attribute. The attribute “cid” is the primary key. The relationship “Connected\_To” enables for a hierarchy of categories where one “parent” has many “children.” The “Categorized” relationship ties an item to a category, which will be used when browsing, or searching.

cid

name

Category

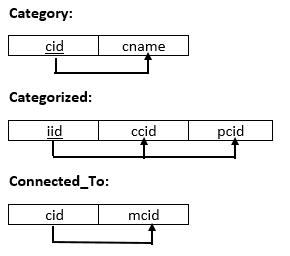
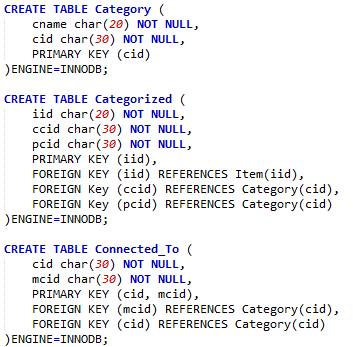
**Figure 5 – Category**

Item

Connected\_To

Categorized

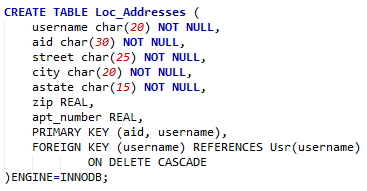
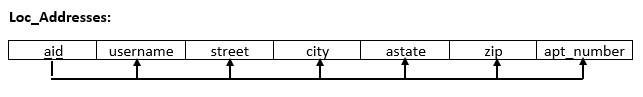
Shown here are the SQL statements to create the tables for “Category”, “Categorized”, and “Connected\_To” depicted in the ER Diagram for Figure 5. The conditions for “Categorized” are met by keeping the one-to-many participation constraint intact with foreign keys from “Item” and two from separate “Category” entities. Also, the third normal form diagrams representing the entities and relationships are depicted below.



**Figure 6 – Category SQL & Schema**

Shown here is the weak entity "Address" and the relationship "Located." An address has the attributes id (aid), street, city, apartment number, stat, and zip. An address has a total participation and key constraint with the “Located” relationship. The “Located” relationship also has total participation with users. An address is a weak entity and is therefore bolded and as such the “Located” relationship is bolded as well.

aid

“Loc\_Addresses”, which represents the “Located” relationship in Figure 7, is shown in the SQL statement. Because addresses is a weak entitiy, we have the foreign key from “Usr” to take on the condition ON DELETE CASCADE. The third normal form diagram is also blelow to represent the entitiy.

**Figure 7 – Address**

Apt\_Num

State

Zip

City

Street

User

Address

Located

**Figure 8 – Address SQL & Schema**

The “Phone” entity has an id (phid), which is the partial key and as such has a dotted underline. It also has a number attribute. The “Contacted\_At” relationship has total participation from the user and phone entities, as well as having a key constraint with the phone entity. Because the phone entity is a weak entity, it is bolded as well as the “Located” relationship.

**Figure 9 – Phone**

number

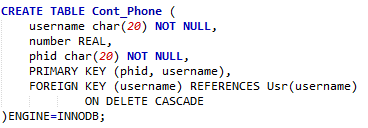
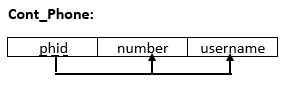
phid

Phone

Contacted\_At

User

“Cont\_Phone”, representing the “Contacted\_At” relationship for “User” and “Phone” in the ER diagram in Figure 9, is show in the SQL statement below. The weak relationship is held with the foreign key from “Usr” having the ON DELETE CASCADE condition to represent the weak relationship. The third normal form diagram for “Cont\_Phone” is also depicted below.



**Figure 10 – Phone SQL & Schema**

Shown here is the weak entity "Credit\_Card" and the relationship "Cashable." A credit card has a card number, which is a partial key as a credit card is a weak entity, card type, and expiration attributes. The “Cashable” relationship has total participation from the user and credit card and also has a key constraint on the credit card.

**Figure 11 – Credit Card**

type

number

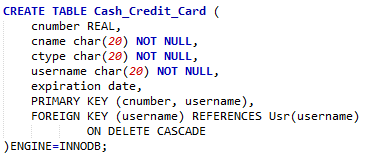
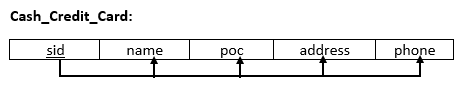
Expiration

Credit\_Card

Cashable

User

The “Cashable” relationship for a “User” and “Credit\_Card”, represented in the ER Diagram in Figure 11, is shown below in the SQL statement creating the “Cash\_Credit\_Card” table. The weak relationship is held with the foreign key from “User” having the ON DELETE CASCADE effect with it. The third normal form diagram representing the “Cash\_Credit\_Card” is depicted below as well.

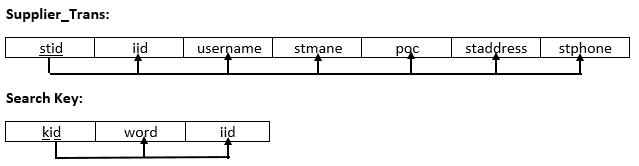
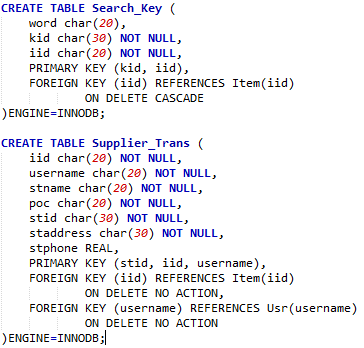


**Figure 12 – Credit Card SQL & Schema**

Shown here is the "Supplier" entity, "Keyword" weak entity, and the "Supplier\_Trans" and "Searched\_By" relationships. The supplier has an id (sid), which is the primary key, company name, address, person of contact, and phone number attributes. The “Supplier\_Trans” relationship has total participation and key constraints on the supplier, user, and item entities. The keyword entity has an id (kid), which is the partial key, and word attributes. The “Searched\_By” relationship has no participation or key constraints because an item can have multiple keywords and keywords can be applied to multiple items.

Address

POC

“Search\_Key” and “Supplier\_Trans”, depicted in the third normal form diagrams and SQL statements that create their tables below, represent the Supplier and Supplier Transaction in Figure 13 above. The weak relationship between “Item” and “Keyword” are held (with ON DELETE CASCADE with the foreign key referencing “Item”) as well as the participation constraints required by “Supplier\_Trans” which the foreign keys referenced in “Usr”, “Item”, and “Supplier.” 

**Figure 13 – Supplier and Supplier Transaction**

Keyword

Searched\_By

Item

User

Supplier

Supplier\_Trans

kid

word

sid

Phone

Name

Item

**Figure 14 – Keyword & Supplier SQL & Schema**

Shown here is the weak entity "Shop" and the relationships "Run\_By" and "Stocked\_By." Shops have an id (shid), which is the partial key, and name attributes. The “Run\_By” relationship has total participation and key constraints on both the user and shop. The “Stocked\_By” relationship has total participation and key constraint on items, but no constraints on the shop because many items can be in one shop, but an item can’t be in multiple shops.

**Figure 15 – Shop**

User

Runs\_Shop

Shop

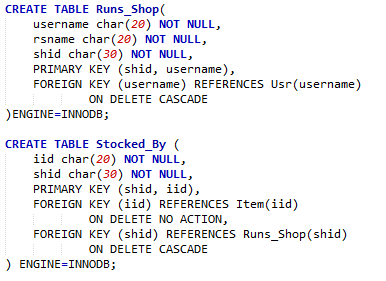
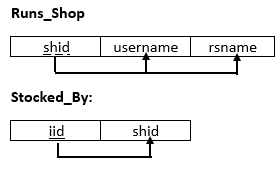
Item

Stocked\_By

shid

Name

“Runs\_Shop” and “Stocked\_By” described in the SQL statements below represent our Shop function. The weak relationship by “Runs\_Shop” is held with the foreign key referencing “Usr” with the condition of ON DELETE CASCADE. The “Stocked\_By” table also has the condition when referencing “Runs\_Shop” to have the effect of ON DELETE CASCADE. These have our conditions for the two relationships hold true. There are also the third normal form diagrams representing these tables as well.



**Figure 16 – Shop SQL & Schema**

Shown here is the weak entity "Wish\_List" and the relationships "Wishes\_For" and "Filled\_With." Wish lists have an id (wid), which is the partial key, attribute. The “Wishes\_For” relationship has total participation and key constraints on both the user and shop. The “Filled\_With” relationship has no constraints on the items or shop because many items can be in one wish list, and the same item can be in multiple wish lists.

wid

**Figure 17 – Wish List**

Ranking

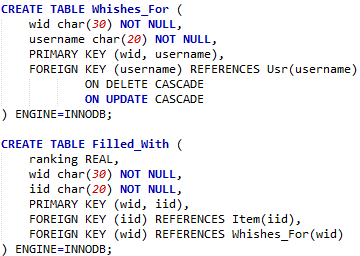
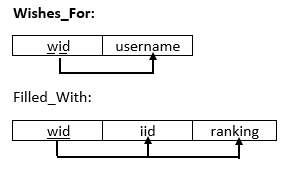
Filled\_With

Item

User

Wishes\_For

Wish\_List

 “Wishes\_For” and “Filled\_With” tables in the SQL statements described below represent our Wish List function we wished to implement. The weak relationship represented in “Wishes\_For” is held true when referencing the foreign key in “Usr” by having the effect of ON DELETE CASCADE as well as ON UPDATE CASCADE (incase the user is updated). The third normal form diagrams are depicted below representing these tables.

**Figure 18 – Wish List SQL & Schema**

The figure below shows the “User\_Stocked” and the “Supplier\_Stocked” relationships. The “User\_Stocked” relationship has total participation and key constraint on the item, but has no constraint on the user because one user can stock many items, but an item belongs to only one user. The “Suplier\_Stocked” relationship has total participation and key constraint on the item, but has no constraint on the supplier because one supplier can stock many items, but an item belongs to only one supplier.

**Figure 19 – Stocked**

Supplier

Supplier\_Stocked

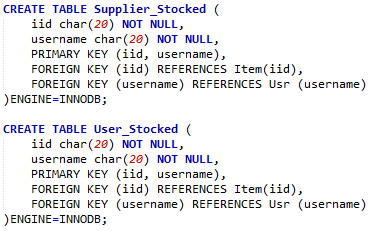
Item

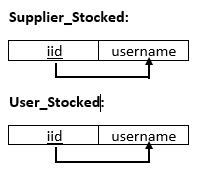
User

User\_Stocked

Item

# “Supplier\_Stocked” and “User\_Stocker” representing our item stocking functions and are described in both the SQL statements below as well as the third normal form diagrams. The participation constraints are held in both tables with foreign keys representing their respective item.





**Figure 20 – Stocked SQL & Schema**

# 

**Figure 21 – Full ER Diagram**

# 4. Technology Survey

Our final finished project for maetS will consist of various interfaces for both the frontend and backend of the web application. The languages we are leaning towards for the front end include the use of Javascript (JQuery, RequireJS, AngularJS, or Google Web Kit), HTML5, CSS (with Bootstrap). On the backend for maetS, we will being using some software called PHPMyAdmin to hold our Databases codes using MySQL. Another language we intend to use involves Java and Java resources. These resources include JSP (next Java in HTML) and to have Java interact with the database using JDBC.

# 5. Conclusion

Our goal for maetS is to have one concise platform to sell games as well as game related content. The ability for users to auction or sell directly will enable them to distribute their content as they see fit. The wish list feature will allow people, who are at the time unable to purchase content, a place to store it for the future. The wish list is also a great feature because it allows others to see what sort of content you want and allows for the option of gifting. The personal shop feature enables those who like content of a specific user to easily find content that they are selling. This also allows users to promote their content without others needing to search through an entire catalog of content.