**Relational Model Rational**

**NOTE: There are not many checks, assertions, triggers because we decided that what these would accomplish, we will do in the functions which handle transactions (adding an auction, making a bid, etc.).**

There are 7 tables: Auction, Bid, Customer, Employee, Invoice, Item, User.

Below is a description for each table.

User:

A user can be an Employee or a Customer. This table will hold the login information for a user

when they sign up for the auction house as well as other general information (Address, etc.). The primary key is a composite key because UserID will either be the primary key of Employee (which is SSN) or Customer (which is CustomerID) and the UserType will distinguish what table to look in. So if there's a Customer with CustomerID = 1 then UserType = Customer and UserID = 1. We added an index on UserName because when a user is logged in we will be doing many queries to select User information based on the UserName. Another index, FirstName, LastName has been added, one reason being that we will be adding a transaction to produce a list of sales by customer name.

Item:

An item is what is being sold in an auction. ItemType is an enum which can't be null because when an item is added, we will have to make sure it is of a certain type and that it is of a certain type. In order to easily keep track of what item types are supported in the auction house, having the enum will be beneficial. We added three indices, two of which are by ItemType and ItemName. There were added for one reason being because we will be adding a transaction to produce a summary listing of revenue generated by a particular item and item type. Another index by CopiesSold was added for one reason being because we will be adding a transaction to produce a Best-Sellers list of items.

Invoice:

An invoice will hold financial information about an auction that has been successfully completed. Out of all the current tables, only Invoice has rows which can be deleted. An invoice will be deleted when an invoice is voided. One attribute, Revenue, was added to hold how much money the auction house made on an order. By summing all the revenues together, we can see how much the auction house has made in total. It also allows us to figure out which buyer/seller/customer/item generated the most revenue. The four foreign keys automatically have indices created for them, which is useful for the queries to see how much one of the entities has generated in revenue. They are also useful for a transaction to produce a list of sales by item name or by customer name (using a join to the Item, Customer table, respectively). This table will also be beneficial for the transaction to get a sales report for a particular month.

Employee:

An employee oversees an auction. An employee can either be a Manager or a Customer-Representative. This distinction is made by the EmployeeType attribute. When we add Manager-level and Customer-Representative transactions, we can check the EmployeeType attribute to see which transactions we will allow for the selected employee. We did not see it necessary to add an index besides the automatically created primary key index because the SSN is how we will be selecting employees in queries for the most part.

Customer:

A customer will act as either a buyer or a seller in an auction. We did not see it necessary to add an index besides the automatically created primary key index because the CustomerID is how we will be selecting customers in queries for the most part.

Bid:

A bid is the bid made for an item in an ongoing auction. A bid is not allowed to be deleted once it is made. This table will be very beneficial for when we add a transaction to get the bid history of a particular auction, because all we will have to do is know the AuctionID and then we get select information from this table where the AuctionID = the desired AuctionID. Also having BuyerID as an

attribute will allow us to know the Customer who made the bid, so we can also get the bid history made by a desired customer. The primary key is a composite key of AuctionID, TimeOfBid to prevent the chance of two bids being made at the same time on the same auction. However, it is perfectly fine for two bids to be placed at the same time if the auctions are different.

Auction:

An auction cannot be deleted from the database. An auction has an employee, buyer, seller, and item attached to it. Buyer and seller are both Customer entities. Whichever corresponding employee to the EmployeeID in the Auction row is the employee who is overseeing the auction. Whichever corresponding item to the ItemID in the Auction row is the item being sold. Whichever corresponding customer to the BuyerID in the Auction row is the buyer who won the auction. BuyerID is null until the very end of the auction which is when the buyer is decided. Whichever corresponding customer to the SellerID in the Auction row is the customer who posted the auction. InvoiceID can be null because an Auction only has an invoice when the auction is completed. When an Invoice is deleted, then the InvoiceID becomes null. This can occur when an invoice is voided, in which case the invoice is deleted. To deal with time left for an auction, we do the following:

A user will specify TimeLeft, which we will make sure is 3, 5, or 7, to specify how many days this will last (3 days, 5 days, 7 day) and then we will fill in the CloseTime attribute based on TimeLeft.

**Below is an explanation for the views ActiveAuctionView, CustomerInfoView, AuctionInfoView**

ActiveAuctionView:

This view displays all Auctions in which the closing time is null. All Auctions with no closing time is still active and ongoing. So for all active auctions, we display the fields AuctionID, the UserName of the seller, the ItemName of the item being sold, the Auction's OpeningTime, the TimeLeft, and the current public bid. These are the fields relevant to an active auction.

CustomerInfoView:

This view displays Customer information but hides the Credit Card Number column for security reasons.

AuctionInfoView:

This view displays Auction information but hides the Increment and Highest Bid columns