Nicklaus Yao
Database Management

Part 3 of the Online Auction System

Adjustments from Previous Steps:

Separating the Auctioneer entity from the User entity reduces the potential for redundancy. The only additional attribute in the original Auctioneer entity was a PaymentCollection, which can be synonymous with the PaymentMethod attribute which already exists under User.
With this change, we can also remove the ActioneerID in the Credibility Entity, Item Entity, and Auctions Entity to simplify the ER Diagram. Lastly, we can also remove the need for the relation AUCT_CRED that used to link Auctioneer and Credibility entities as this should now be included in the USER_CRED relation. Additionally, we can update some of the relationships that involved the Auctioneer entity and reflect this in the table below.
Attached below is the new ER Diagram.

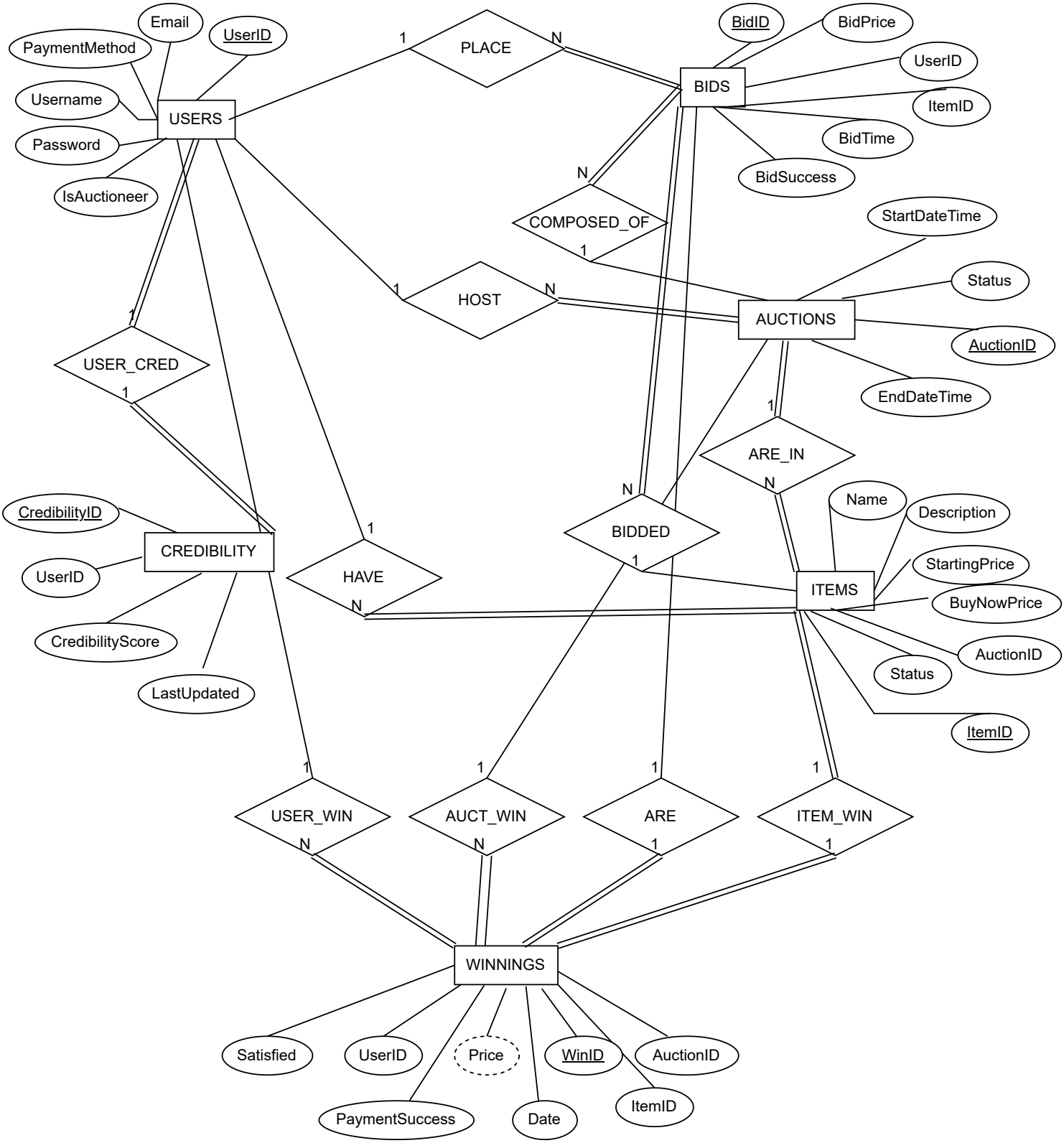
Users/Bidder:

Attributes:

- UserID
  - Domain: Unique alphanumeric value (starting with 5)
  - Data Type: VARCHAR
- Username
  - Domain: Unique alphanumeric combination (min 3, max 12)
  - Data Type: VARCHAR
- Password(Encrypted)
  - Domain: Any alphanumeric combination (min 3, max 12)
  - Data Type: VARCHAR
- Email
  - Domain: Must have domain + @ + _____. ____
  - Data Type: VARCHAR
- IsAuctioneer
  - Domain: True or False
  - Data Type: Boolean
- Payment Method (not 100% safe)
  - Domain: Must not be Null, Credit Card Number
  - Data Type: Numeric

Keys:
Primary Key: UserID
Foreign Key: None

Relationship	Type	Name	Participation
Users to Items	1:N	HAVE	Partial to Full
Users to Auction	1:N	HOST	Partial to Full (if IsAuctioneer True)


Email
UserID
PaymentMethod
Username
Password
IsAuctioneer
USERS
PLACE
1
N
BidID
BidPrice
UserID
ItemID
BidTime
BidSuccess
BIDS
N
COMPOSED_OF
1
StartDateTime
Status
AuctionID
EndDateTime
AUCTIONS
1
HOST
N
1
USER_CRED
1
1
ARE_IN
N
CredibilityID
UserID
CredibilityScore
LastUpdated
CREDIBILITY
1
HAVE
N
N
BIDDED
1
Name
Description
StartingPrice
BuyNowPrice
AuctionID
Status
ItemID
ITEMS
1
1
1
1
USER_WIN
N
AUCT_WIN
N
ARE
1
ITEM_WIN
1
WINNINGS
Satisfied
UserID
Price
WinID
AuctionID
PaymentSuccess
Date
ItemID

ER-to-Relational Mapping:

Step 1: Mapping of Regular Entity Types

All of the entities present in my previous ER diagram are regular entities. This means that we need to create the relations for Users, Items, Auctions, Bids, Credibility, and Winnings. There are also no composite attributes present which means that all our attributes are simple attributes. This also means that none of the primary keys are composite so we do not need to worry about that. We do not need to include the foreign keys or the relationship attributes yet.

USERS:

UserID	Username	Password	Email	PaymentMethod	IsAuctioneer

ITEMS:

ItemID	Name	Description	StartingPrice	BuyNowPrice	Status

AUCTIONS:

AuctionID	StartDateTime	EndDateTime	Status

BIDS:

BidID	BidPrice	BidTime	BidSuccess

CREDIBILITY:

CredibilityID	CredibilityScore	LastUpdated

WINNINGS:

WinID	Date	Satisfied	PaymentSuccess

Step 2: Mapping of Weak Entity Types

Based on my ER diagram and all the descriptions of the entities done in Project Steps 1 and 2, there are no Weak Entity Types present thus there is no need to map any weak entity types.

Step 3: Mapping of Binary 1:1 Relation Types

Based on class notes, it is best to try to use the foreign key approach unless we have special conditions.

1. Winnings to Bids:

We map the relationship type **ARE** by choosing the primary key of BIDS relation as the foreign key of the WINNINGS relation because all winnings need to be bids (total/full). We can rename this to **Winning_bid.**

2. Credibility to Users:

We map the relationship type **USER_CRED** by choosing the primary key of USER as the foreign key for Credibility because although this is total participation on both sides and either or can be used as the primary key, I prefer not to overload the Users with more attributes and have the foreign key in Credibility. We can rename this to **Users_cred.**

3. Winnings to Items:

We map the relationship type **ITEM_WIN** by choosing the primary key for Items as the foreign key of the WINNINGS relation because all winnings need to be of an item. You cannot win something that is not an item. We can rename this to **Won_item.**

Step 4: Mapping of Binary 1:N Relationship Types

1. Auctions to Items:

The relationship **ARE_IN** is known to be full to partial because all auctions must have at least 1 item in it but not all items need to be in an auction just yet. We choose the primary key of Auction (AuctionID) as a foreign key for Items. We call this **Auc_item.**

2. Users to Bids:

The relationship where Users **PLACE** Bids is known to be partial to full as not all users must place a bid down. We choose the primary key of the Users Entity as a foreign key for Bids. We call this **Users_bid.**

3. Users to Winnings:

The relationship **USER_WIN** is known to be Partial to Full because not all users have won but all winners must be associated with a user. We will choose the primary key for the Users entity as the foreign key for Winnings. We call this **Winning_user.**

4. Auctions to Winnings:

The relationship **AUCT_WIN** is known to be Partial to Full because Auctions can be canceled with no winners. There can be multiple different winners per auction because there can be multiple items within an auction. We will choose the primary key of Auctions to be the foreign key for Winnings. We call this **Auc_wins.**

5. Auctions to Bids:

The relationship **COMPOSED_OF** is known to be Partial to Full because Auctions can have multiple bids due to an item having multiple bids in an auction. We will choose the primary key of Auctions to be the foreign key for Bids. We call this **Auc_bids.**

6. Items to Bids

The relationship **BIDDED** represents how each item can have multiple bids. Without the many different bids, then the item is just sold to the first person(buying not bidding anymore). This is partial to full because not all items need to have been bid on yet. We will choose the primary key of Items to be the foreign key for Bids. We call this **Items_bids.**

7. Users to Auction

The relationship **HOST** represents how an auction can only be associated with one user but a single user can host many auctions throughout their time. We will choose the primary key of Users to be the foreign key for Auctions. We call this **Users_auc.**

8. Users to Items

This relationship is represented by **HAVE** where Users can own multiple items to put up on auction. Not all users have to have items (some people just like to bid and not sell) but all items must be associated with an owner(User). We will choose the primary key of Users to be the foreign key of Items. We call this **Users_items**

Step 5: Mapping of Binary M: N Relationship Types

I do not have any M: N relationship types for my auction system so there is nothing to map for this step!

Step 6: Mapping of Multivalued attributes

I do not have any multivalued attributes present thus no need to map!

Step 7: Mapping of N-ary Relationship Types

I do not have any n-ary relationships present. The closest relationship that I have would be the Winnings entity has 3 foreign keys of BidID, ActionID, and UserID but these are linked

separately as 3 different binary relationships rather than 1 n-ary relationship. I chose to avoid using n-ary relationships because it is not necessary and would make the database more complex.

Mapping EER Model Constructs to Relations

Step 8: Options for Mapping Specialization or Generalization.

While it is possible to map specializations, the auction system that I am currently building is extremely generic rather than for a set number of specific products. If I knew what type of products this system would be auctioning out, I may be able to specialize based on product type but for the time being, there is no specialization or generalization needed.

Step 9: Mapping of Union Types (Categories).

In this auction system, there are no mappings necessary of Union Types because there are no Union types!

Normalizing the Database to 3NF:

Checking for 1NF:

Goal: Remove any multivalue attributes, composite attributes, or any combinations. Make sure that all attributes only have single atomic/indivisible values.

Based on our tables, there are no composite attributes or multivalue attributes, to begin with as we store items and bids all separately and are only joined rather than having an item store multiple bids. All auctions that have multiple items are also represented as two separate tables already as well. All in all, there are no multivalue or composite attributes and no divisible values stored.

Checking for 2NF:

Goal: Every nonprime attribute must be fully dependent on the primary key of the table.

We know that we have achieved 2NF because we already are in 1NF and I do not have any tables that have more than 1 attribute as a primary key. According to the text, “if the primary key contains a single attribute, the test does not need to be applied at all”.

Checking for 3NF:

Goal: No nonprime attribute of a table is transitively dependent on the primary key.

Based on our tables, we looked to create them so that only the necessary attributes are within each table to ensure that there are no transitive dependencies between them. For example, the bid table does not include any information about the item that they are bidding are and only refers to the bid price. It is important to note that there is no dependency here as the ITEM table only contains info about the starting price and buy-now price, this is independent of the bid price in the BIDS table.

Integrity Constraints:

Users:

Attributes:

- Primary Key: UserID
    - Domain: Unique alphanumeric value (starting with 5)
    - Data Type: VARCHAR
    - Must be Unique
    - Must be NON-NULL
- Username
    - Domain: Unique alphanumeric combination (min 3, max 12)
    - Data Type: VARCHAR
    - Must be Unique
    - Must be Non-null
- Password(Encrypted)
    - Domain: Any alphanumeric combination (min 3, max 12)
    - Data Type: VARCHAR
    - Must be Non-null
- Email
    - Domain: Must have domain + @ + _____. ____
    - Data Type: VARCHAR
- PaymentMethod
    - Domain: Credit Card Number
    - Data Type: Numeric
    - Cannot be Null
- IsAuctioneer
    - Domain: True or False
    - Data Type: BOOLEAN
    - Must be NON-NULL

Keys:

Primary Key: UserID
Foreign Key: None

Items:

Attributes:

- Primary Key: ItemID
    - Domain: Unique alphanumeric value (7+)
    - Data Type: VARCHAR
    - Must be Unique
    - Must be Non-null
- Name
    - Domain: Unique alphanumeric value (3-12 characters)
    - Data Type: VARCHAR
    - Must be Non-null
- Description
    - Domain: Within 250 characters
    - Data Type: TEXT
- StartingPrice
    - Domain: Price above $1
    - Data Type: INT
- BuyNowPrice
    - Domain: Price Above $1 and greater than the Starting Price
    - Data Type: INT
- Status (‘Sold’, ‘Taken Down’, ‘In-Auction’, ‘Pending’)
    - Domain: The above String Values
    - Data Type: String
    - Must be Non-null
- Foreign Key: Auc_item
    - Domain: Must be an existing AuctionID
    - Data Type: VARCHAR
    - Must be Non-null
- Foreign Key: Users_item
    - Domain: Must be an existing UserID
    - Data Type: VARCHAR
    - Must be Non-null

Keys:

Primary Key: ItemID

Foreign Key: Auc_item, Users_item

Auctions:

Attributes:

- Primary Key: AuctionID
    - Domain: Must be Unique Alphanumeric Value
    - Data Type: VARCHAR
    - Must be Unique
    - Must be NON-NULL
- StartDateTime
    - Domain: Must be current time or in the future
    - Data Type: TIMESTAMP
    - Must be NON-NULL
- EndDateTime
    - Domain: Must be after StartDateTime
    - Data Type: TIMESTAMP
    - Must be NON-NULL
- Status (‘Canceled’, ‘Completed’, ‘Pending’, ‘Active’)
    - Domain: Set strings above
    - Data Type: VARCHAR
    - Must be NON-NULL
- Foreign Key: Users_auc
    - Domain: Must be an existing UserID
    - Must be NON-NULL

Keys:

Primary Key: AuctionID
Foreign Key: User_auc

Bids:

Attributes:

- Primary Key: BidID
    - Domain: Identifier for Each Bid
    - Data Type: VARCHAR
    - Must be Unique
    - Must be NON-NULL
- BidPrice
    - Domain: Any value in form $_._ _ greater than previous bid
    - Data Type: NUMERIC
    - Must be NON-NULL
- BidTime
    - Domain: Within the Start and End time of Auction
    - Data Type: TIMESTAMP
    - Must be NON-NULL
- BidSuccess (Based on BidTime + BidPrice)
    - Domain: True or False
    - Data Type: Boolean
    - Must be NON-NULL

- Foreign Key: Items_bids
    - Domain: Must reference a valid ItemID
    - Must be NON-NULL
- Foreign Key: Users_bids
    - Domain: Must reference a valid UserID
    - Must be NON-NULL
- Foreign Key: Auc_bids
    - Domain: Must reference a valid AuctionID
    - Must be NON-NULL

Keys:
Primary Key: BidID
Foreign Key: Items_bids, Users_bids, Auc_bids

Credibility:

Attributes:

- Primary Key: CredibilityID
    - Domain: Unique Identifier
    - Data Type: VARCHAR
    - Must be Unique
    - Must be NON-NULL
- CredibilityScore (scored based on successful payments, auctions created, etc.)
    - Domain: Between the Scores of 0-100
    - Data Type: INT
    - Must be NON-NULL (can default as 85)
- LastUpdate
    - Domain: Only updated if the Date given is after LastUpdate
    - Data Type: TIMESTAMP
    - Must be NON-NULL
- Foreign Key: Users_cred
    - Domain: Must reference an existing UserID
    - Must be NON-NULL

Keys:

- Primary Key: CredibilityID
- Foreign Key: Users_cred

Winnings:

Attributes:

- Primary Key: WinID
    - Domain: Alphanumeric value
    - Data Type: VARCHAR
    - Must be NON-NULL
    - Must be Unique
- Price

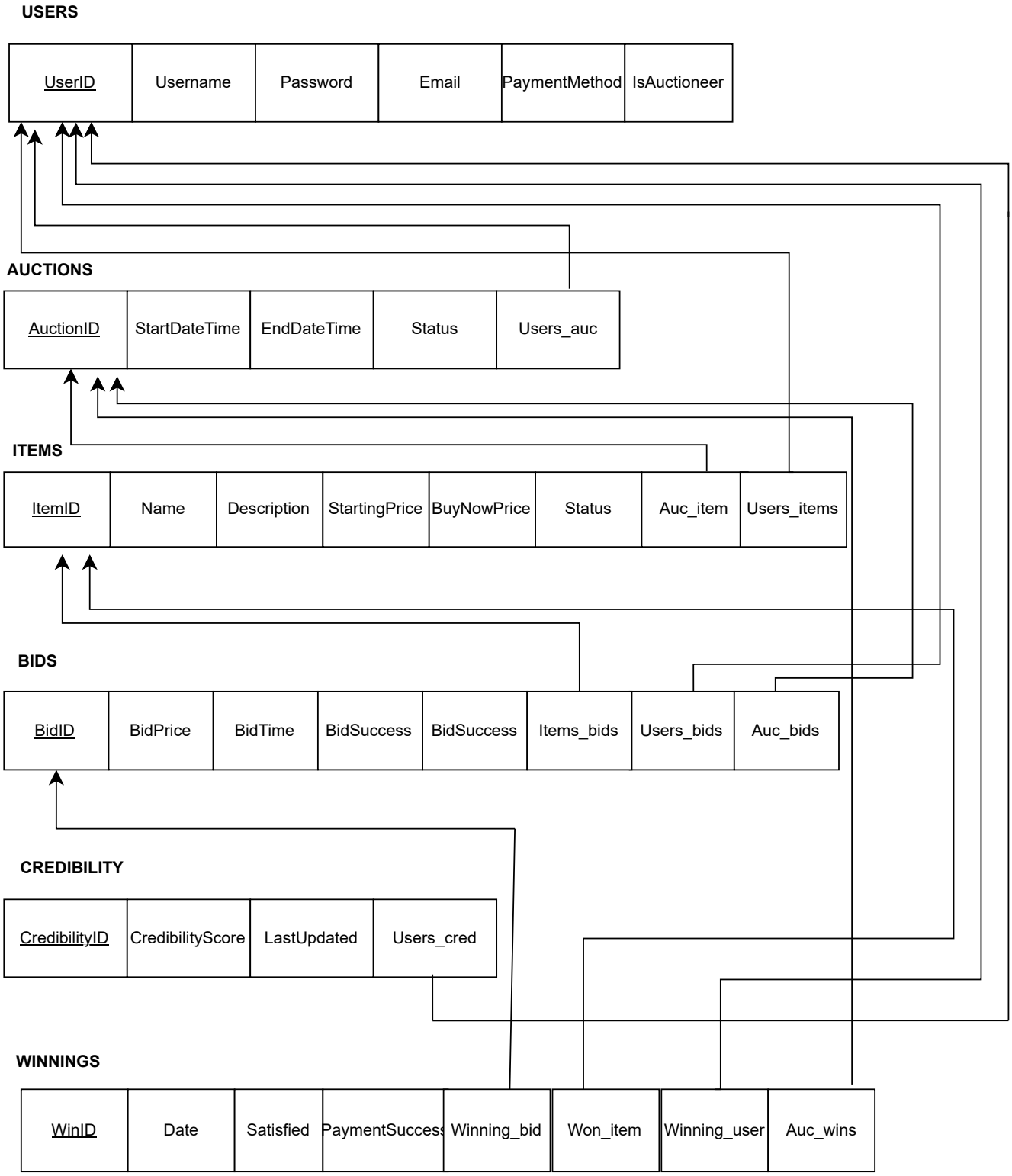
    - Domain: Buy Price or Highest Bid Price + Any Commissioned Fee
    - Data Type: NUMERIC
    - Must be NON-NULL
- Date
    - Domain: Day that Item was Sold
    - Data Type: TIMESTAMP
    - Must be NON-NULL
- PaymentSuccess (Impacts Credibility Score for Bidder)
    - Domain: True or False
    - Data Type: Boolean
    - Must be NON-NULL
- Satisfied (Impacts Credibility Score for User)
    - Domain: Yes or No
    - Data Type: Boolean
    - Must be NON-NULL
- Foreign Key: Winning_bid
    - Domain: Must reference an existing BidID
    - DataType: VARCHAR
    - Must be NON-NULL
- Foreign Key: Won_item
    - Domain: Must reference an existing ItemID
    - Data Type: VARCHAR
    - Must be NON-NULL
- Foreign Key: Winning_user
    - Domain: Must reference an existing UserID
    - Data Type: VARCHAR
    - Must be NON-NULL
- Foreign Key: Auc_win
    - Domain: Must reference an existing AuctionID
    - Data Type: VARCHAR
    - Must be NON-NULL

Keys:
Primary Key: WinID
Foreign Key: Winning_bid, Won_item, Winning_user, Auc_win

Design Choices:

Most design choices have already been explained above in terms of trying to create a 3NF database. I have chosen to not have any foreign keys in the Users table and have so many foreign keys of UserID in the other tables to ensure that we can always directly join the USER table to another for easy access and querying. Additionally, we do not have to worry that including a foreign key to the UserID will increase the number of rows/entries because all other tables must have at least as many rows as there are bidders/auctioneers who are Users. For example, I do not have a table like “Department Location” where it would not make sense to have foreign keys for the employee as it would be much larger than necessary. However, in my auction system, all

users must have an entry in the Credibility table, all Winnings must be associated with a user, there will be more bids than there are users usually (if at least each user has bid once), and every single auction must be associated with a User host. The most important change from the previous steps in the project is the opening paragraph on removing the Auctioneer entity and having an attribute in the USERS to indicate this if a User is a host for an auction or not.


USERS
UserID
Username
Password
Email
PaymentMethod
IsAuctioneer
AUCTIONS
AuctionID
StartDateTime
EndDateTime
Status
Users_auc
ITEMS
ItemID
Name
Description
StartingPrice
BuyNowPrice
Status
Auc_item
Users_items
BIDS
BidID
BidPrice
BidTime
BidSuccess
BidSuccess
Items_bids
Users_bids
Auc_bids
CREDIBILITY
CredibilityID
CredibilityScore
LastUpdated
Users_cred
WINNINGS
WinID
Date
Satisfied
PaymentSuccess
Winning_bid
Won_item
Winning_user
Auc_wins