1. System Requirement Specification

Abstract - An online auction is a service in which auction users or participants sell or bid for products or services via the Internet. Virtual auctions facilitate online activities between buyers and sellers in different locations or geographical areas. The goal of this project is to provide the user with a simple yet efficient backend system in the form of SQL queries to enable users to avail all basic functionalities of an online bidding system.

We have assumed two main actors in our online auction system -

1. Seller
2. Buyer

The following are the functionalities of the system -

1. **Listing an item for auction**
    - The seller provides information about the product he/she wishes to sell at the auction.
    - To be able to do so, the seller must login or must register if he/she is a first-time user of the system.
    - The seller must provide all the information about the item, along with the opening and closing time of the auction, starting bid price, and the reserve price for the item.
2. **Cancel listing or remove an item from the auction**
    - The seller withdraws an item from the auction by setting auction status to cancelled.
    - To be able to do so, the seller must be logged in.
    - A closed auction can not be cancelled.
3. **View currently active auctions**
    - The buyer views the items available for bidding by viewing a list of currently open auctions.
4. **Placing bid**
    - The buyer places a bid for a currently open auction.
    - The buyer should first log in and place the bid by mentioning the bid amount and the auction for which he/she wishes to bid.
    - The current high bid is updated accordingly in the auction.
    - The buyer can place multiple bids for the same or different items.
    - A seller cannot bid on its own product.
5. **Close auction**
    - Once the current time exceeds the closing time of the auction, status is updated to closed and the seller is notified.
    - The winner is notified to pay for the purchase.
6. **Payment for purchase**
    - The winning bidder enters billing information and pays for the purchase once the auction is closed.
    - The bidder must be logged in to be able to do so.
7. **Update item**
    - The seller changes the details of an item(only changing the reserve price, the closing time of the auction, item category, and item description are allowed to avoid any unfair practice) offered in the auction.
    - To update, the auction must still be open and the seller should be logged in.
8. **Register as a user**
    - Creating a new account with user-provided data like first name, last name, username, etc.
    - This automatically logs in the user into the system.

9. Logging out of the system

- A logged-in user logs out of the system and is unable to place bids or edit item details.

10. Auctions participated in

- The buyer views lists of all the bids placed by him/her in all the previous auctions and current auctions.
- The buyer should be logged in to be able to view the list.

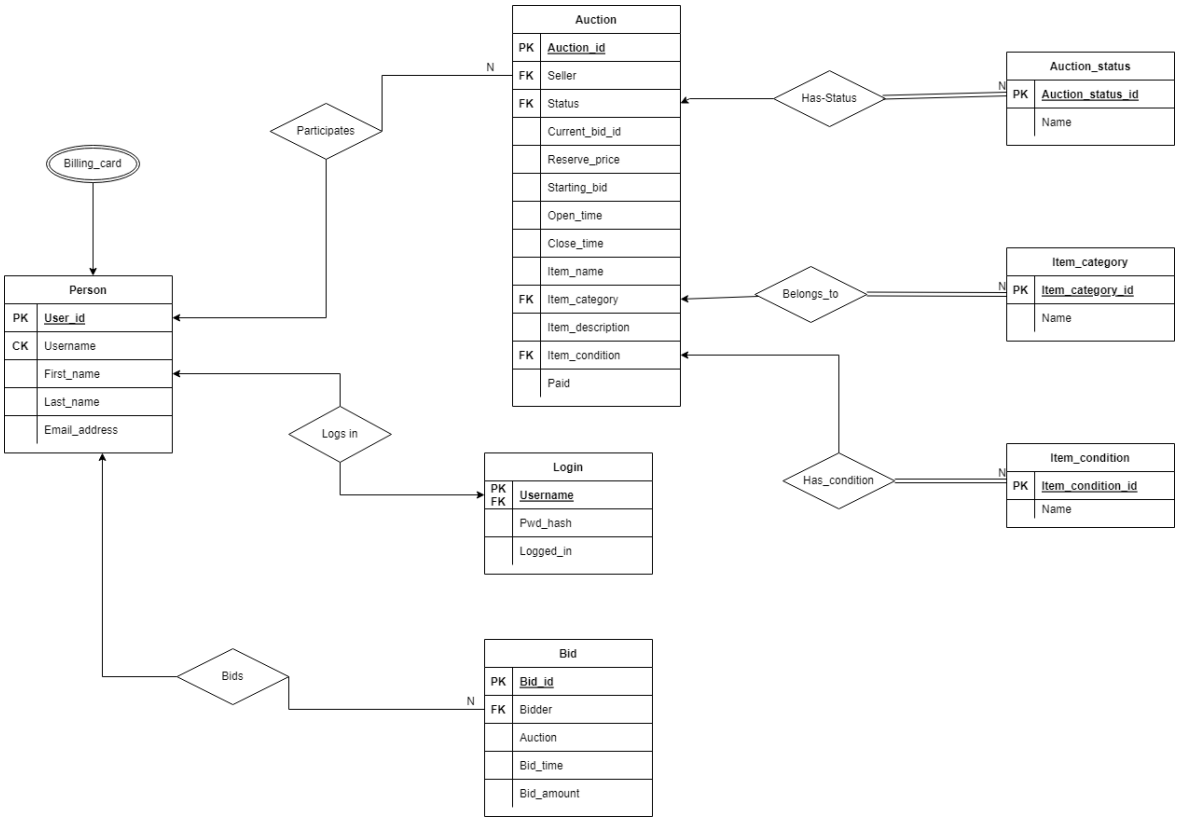
11. View all items bought

- The buyer can view the list of items bought by him/her till now.
- The buyer should be logged in to be able to view the list.

12. Show highest bidder in auction

- The user can query to get current highest bidder's username in auction.

2. System Modelling
2.1. Entity Relationship Diagram



2.2. Schema design

We initially started with having a Person entity that will have user_id, username, first_name, last_name, email_address, billing_card as attributes, however after realizing that billing_card can be multivalued attributes we decided to make separate tables for it to ensure our tables are in 1NF.
Hence we came up with the following 3 tables -
PERSON(USER_ID, USERNAME, FIRST_NAME, LAST_NAME, EMAIL_ADDRESS)
BILLING_INFO(BILLING_INFO_ID, USER_ID, LAST_NAME, FIRST_NAME, CARD_TYPE, CARD_NUMBER, EXP_MONTH, EXP_DAY, SECURITY CODE)

To store type of card(visa, mastercard, etc) , separate table was made. Primary key of this table is foreign key in PERSON table.
ADDRESS_TYPE(ADDRESS_TYPE_ID,NAME)

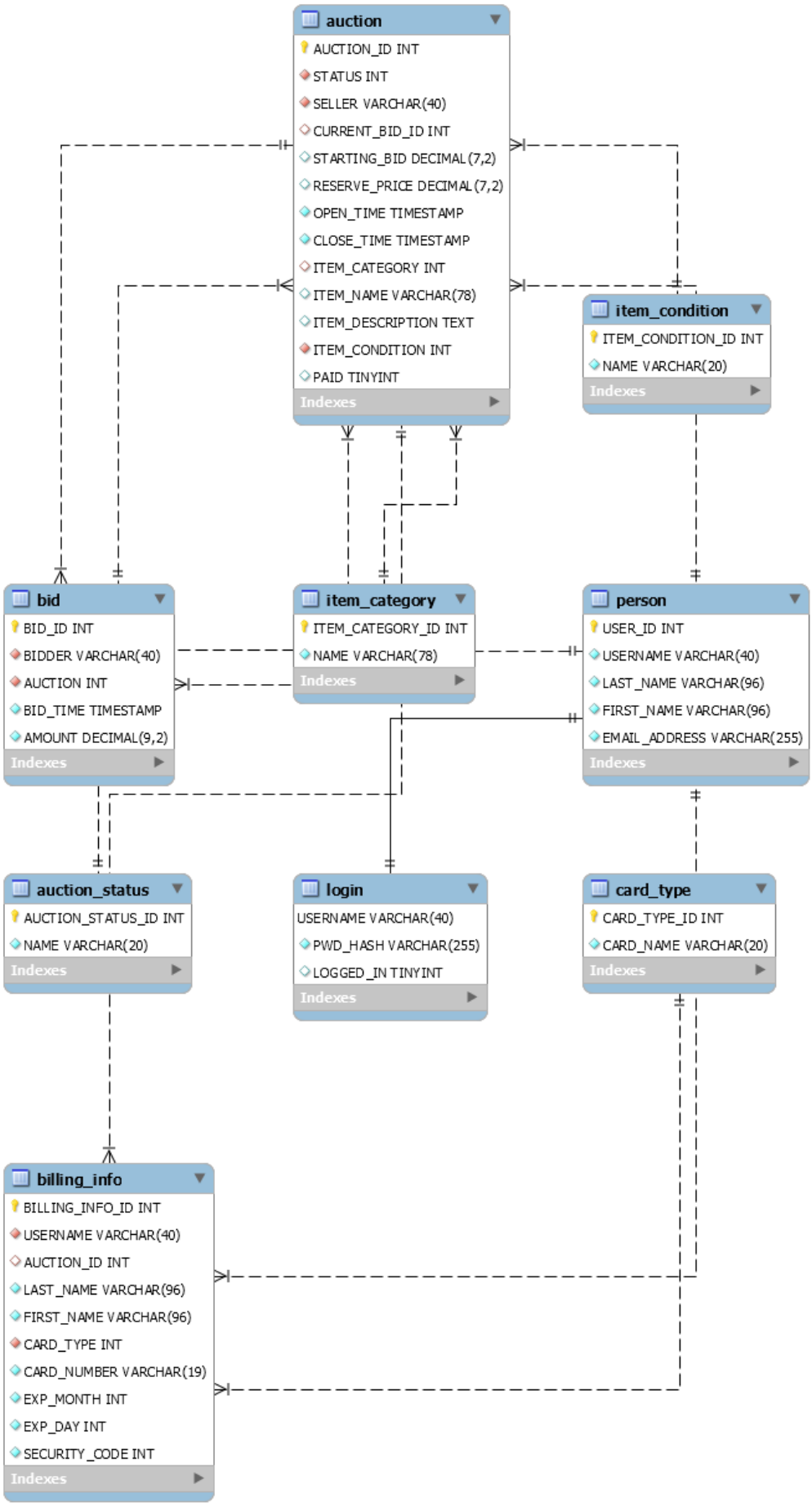
Similarly, the card_type can be made into a separate table -
CARD_TYPE(CARD_TYPE_ID, CARD_NAME)

A user needs to log in to be able to bid/sell items, hence a login table was created with attributes user_id, password_hash, and logged_in.
LOGIN(USERNAME, PWD_HASH,LOGGED_IN)

Next, we created a table for auction as follows -
AUCTION(AUCTION_ID, STATUS, SELLER, CURRENT_BID_ID, RESERVE_PRICE, OPEN_TIME, CLOSE_TIME, ITEM_CATEGORY, ITEM_NAME, ITEM_DESCRIPTION, ITEM_CONDITION, PAID)
Here status indicates whether the auction is currently open, closed, failed, or canceled. To add an item for sell in auction, the seller needs to provide details like reserve_price, open_time, close_time, item_category, item_name, item_condition, item_description and item_condition.
We made separate tables for auction status, item category and item condition. Primary key of these tables are foreign keys in auction table.

Finally, a buyer can place a bid for any currently open auction by providing auction_id and bidding_amount. Hence the bid table is as follows -
BID(BID_ID, BIDDER, AUCTION, BID_TIME, AMOUNT)

Following is the self explanatory schema design of the online auction system -


auction
AUCTION_ID INT
STATUS INT
SELLER VARCHAR(40)
CURRENT_BID_ID INT
STARTING_BID DECIMAL(7,2)
RESERVE_PRICE DECIMAL(7,2)
OPEN_TIME TIMESTAMP
CLOSE_TIME TIMESTAMP
ITEM_CATEGORY INT
ITEM_NAME VARCHAR(78)
ITEM_DESCRIPTION TEXT
ITEM_CONDITION INT
PAID TINYINT
Indexes
item_condition
ITEM_CONDITION_ID INT
NAME VARCHAR(20)
Indexes
bid
BID_ID INT
BIDDER VARCHAR(40)
AUCTION INT
BID_TIME TIMESTAMP
AMOUNT DECIMAL(9,2)
Indexes
item_category
ITEM_CATEGORY_ID INT
NAME VARCHAR(78)
Indexes
person
USER_ID INT
USERNAME VARCHAR(40)
LAST_NAME VARCHAR(96)
FIRST_NAME VARCHAR(96)
EMAIL_ADDRESS VARCHAR(255)
Indexes
auction_status
AUCTION_STATUS_ID INT
NAME VARCHAR(20)
Indexes
login
USERNAME VARCHAR(40)
PWD_HASH VARCHAR(255)
LOGGED_IN TINYINT
Indexes
card_type
CARD_TYPE_ID INT
CARD_NAME VARCHAR(20)
Indexes
billing_info
BILLING_INFO_ID INT
USERNAME VARCHAR(40)
AUCTION_ID INT
LAST_NAME VARCHAR(96)
FIRST_NAME VARCHAR(96)
CARD_TYPE INT
CARD_NUMBER VARCHAR(19)
EXP_MONTH INT
EXP_DAY INT
SECURITY_CODE INT
Indexes

2.3. Normalization

It is clearly visible that all tables are in 1NF as there are no tables with multivalued attributes.
We have ensured that in all the tables, all attributes depend upon the candidate key only and not on any part of the candidate key hence there are no partial dependencies. Hence all tables as in 2NF.
Also, we have ensured that no non-prime attributes are able to identify other non-prime attributes in the tables. Hence the schema is in 3NF.
The tables have been designed in such a way that the primary key of every table is also a superkey. Hence by definition of BCNF, the proposed schema is in BCNF.

Hence highest normalization of the proposed schema is BCNF.

2.4. List of tables required

PERSON(USER_ID, USERNAME, FIRST_NAME, LAST_NAME, EMAIL_ADDRESS)
BILLING_INFO(BILLING_INFO_ID, USERNAME, LAST_NAME, FIRST_NAME, CARD_TYPE, CARD_NUMBER, EXP_MONTH, EXP_DAY, SECURITY CODE)
CARD_TYPE(CARD_TYPE_ID, CARD_NAME)
LOGIN(USERNAME, PWD_HASH,LOGGED_IN)
AUCTION(AUCTION_ID, STATUS, SELLER, CURRENT_BID_ID, RESERVE_PRICE, OPEN_TIME, CLOSE_TIME, ITEM_CATEGORY, ITEM_NAME, ITEM_DESCRIPTION, ITEM_CONDITION, PAID)
AUCTION_STATUS(AUCTION_STATUS_ID, NAME)
ITEM_CATEGORY(ITEM_CATEGORY_ID, NAME)
ITEM_CONDITION(ITEM_CONDITION_ID, NAME)
BID(BID_ID, BIDDER, AUCTION, BID_TIME, AMOUNT)

2.5. Procedures required for smooth functioning of the system

1. register(in fname varchar(96),in lname varchar(96),in uname varchar(40),in eadd varchar(255),in passwd varchar(255)) to register a new user into the system.
2. loggingin(in username varchar(40), in passwd varchar(255)) - to login a registered user into the system.
3. auctioninfo(in auction_id INTEGER) to display all information about given auction.
4. currenthighbid(auctionId integer) - returns varchar(40) to return current highest bidder in a given auction.
5. addItem(in auction_id INTEGER, in status INTEGER, in seller VARCHAR(40), in startingBid NUMERIC(7,2), in reservePrice NUMERIC(7,2), in close_time TIMESTAMP, in item_category INTEGER, in item_name VARCHAR(78), in item_description VARCHAR(998), in item_condition INTEGER) - to add a new item for sale in auction.
6. closeAuction(inout auctionId integer) - to close a currently open auction.
7. payment(in bidder varchar(40), in auctionId integer, in card_type integer, in card_number varchar (19), in exp_month integer, in exp_day integer, in security_code integer) - to handle payments made by the buyer.
8. removeItem(in auctionId integer) - to remove an item from auction.

9. editItem(in auctionId integer, in reservePrice NUMERIC(7,2), in close_time TIMESTAMP, in item_category INTEGER, in item_description VARCHAR(998)) - to edit details of item while the auction is still open.
10. add_bid(in bidder varchar(40), in auctionId integer, in bidTime timestamp, in amt numeric(9,2)) - for placing a bid for an item.
11. logOut(in username varchar(40)) - to logout user from the system.
12. currentActive() - to display currently active auctions.
13. userInfo(in username varchar(40)) - to display infromation about a registered user along with auctions participated in.
14. itemBought(in username varchar(40)) - to display list of items bought by the user.