Contents

DATA TYPES

Auction

Attribute	Data type	Nullable
description	String	Not Null
returnable	Boolean	Not Null
starting_bid	Decimal	Not Null
start_date_time	Date	Not Null
min_sale_price	Decimal	Not Null
get_it_now_price	Decimal	NULL
winner	Integer	NULL
auction_end	Date	Not Null
average_rating	Integer	NULL

Bid

Attribute	Data type	Nullable
amount	Decimal	Not Null
username	String	Not Null
date_time	Date	Not Null

User

Attribute	Data type	Nullable
username	String	Not Null
password	String	Not Null
position	String	Not Null
first_name	String	Not Null
last_name	String	Not Null

AuctionLength

Attribute	Data type	Nullable
auction_length_name	Integer	Not Null

Category

Attribute	Data type	Nullable
category_name	String	Not Null

Condition

Attribute	Data type	Nullable
condition_name	String	Not Null

Rating

Attribute	Data type	Nullable
username	String	Not Null
comment	String	NULL
date_time	Date	Not Null
stars	Integer	Not Null

BUSINESS LOGIC CONSTRAINTS

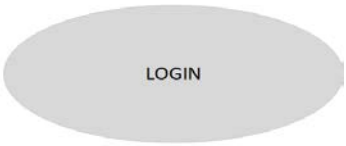
- New users must register an account with the system first.
- Users with an existing account cannot register a new one.
- Administrators cannot register; they must be added directly by a database administrator.
- If a user is an administrator, they must have a position.
- Minimum sale price of an auction must be greater than 0.
- Get It Now price of an auction must be greater than 0.
- Starting bid of an auction must be greater than minimum sale price.
- All bids made must be at least $1 higher than the last.
- Auction length must be one of [1, 3, 5, 7].
- Category must be one of [New, Very Good, Good, Fair, Poor].

ASSUMPTIONS

- An expected ending time is calculated for each auction as a derived attribute based on the auction length, but the actual auction end time is stored in the database.
- All bids are stored.
- The winner of an auction can be set in the database one of two ways:
    - A user purchases the item at the Get It Now price – the winner is set in the database immediately.
    - An auction is accessed by the application (returned for an admin report, etc.) after its ending time. All such database readings will perform a check for a NULL winner field on auctions past their end time. If the winner is NULL, the system will set the winner to the user with the highest bid.
- Average rating is always calculated at the time it is viewed, and is never stored.
- Admin reports are never stored, but always read anew from the database.
- User must be logged in to access any application features.

LOGIN

Task Decomposition



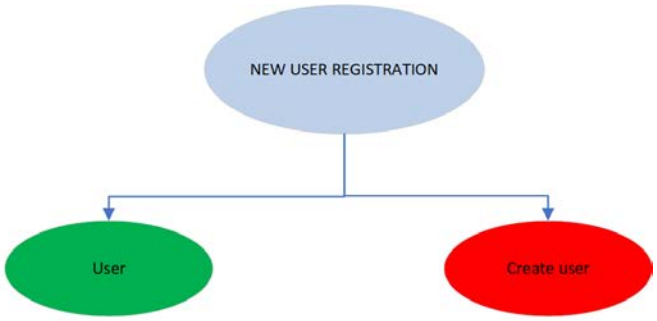
Lock-types: Read-only on Users table
Number of Locks: Single
Enabling Conditions: None
Frequency: 100 logins / day
Consistency: Not critical
Subtasks: Mother task not needed

Abstract Code

- User enters *username* ('$UNAME'), *password* ('$PWD') input fields
- If data validation is successful for both username and password input fields, then:
    - When ***Login*** button is clicked
        - If User record is found, but User.password != '$PWD':
            - Go back to **Login** form with incorrect password error message
        - Else
            - Store the login information in the session variable '$USERID'
            - Redirect to **Menu** form
    - When ***Register*** button is clicked
        - Redirect to **Registration** page

NEW USER REGISTRATION

Task Decomposition



Lock-types: Lookup Users, Insert Users
Number of Locks: Single
Enabling Conditions: None
Frequency: 100 registrations / day
Consistency: Not critical
Subtasks: All tasks must be done. Mother task required to coordinate subtasks. User completes (validating user exists/doesn't exist), then create user. In order to create user, must validate fields and passwords.

Abstract Code

- User enters *First Name*('$FNAME'), *Last Name*('$LNAME'), *username*('$UNAME'), *password*('$PWD'), and *confirm password*('$TEMP_PWD')
- When ***Register*** button is clicked,
    - If all fields != NULL
        - If ('$UNAME') record is not found
            - If *password*('$PWD') == *confirm password*('$TEMP_PWD')
                - User will be created and will be redirected to **Menu** page
            - Else print "passwords must match"
        - Else print "username is already in use, please choose another username
    - Else print "please make sure that all fields have been filled out accurately"
- When ***Cancel*** button is clicked,
    - Redirect to **Login** page

LIST ITEMS

Task Decomposition

Lock-types: Read-only on Category, Condition and Insert on Item table
Number of Locks: Several different schema constructs are needed.
Enabling Conditions: Enabled by user's login
Frequency: High Frequency
Consistency: Not critical
Subtasks: All tasks must be completed

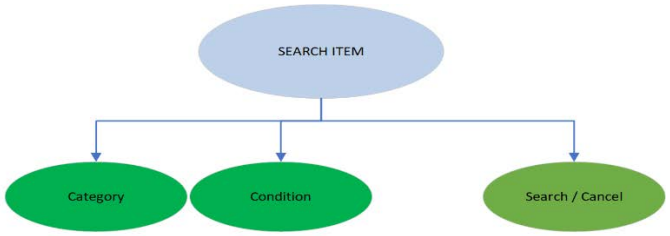
Abstract Code

- User clicked on ***List Items for Sale*** button from the **Menu** :
- User enters *Item Name*, *Description* fields
- User clicks *Category* list and selects only one of the following
    - Art
    - Books
    - Electronics
    - Home and Garden
    - Sporting Goods
    - Toys
    - Other
- User clicks *Condition* list and selects only one of the following
    - New
    - Very Good
    - Good
    - Fair
    - Poor
- User enters only numbers in the *Start Auction Bidding*, *Minimum Sale Price* fields

- User clicks *Auction Ends In* list and selects only one of the following
    - 1 day
    - 3 days
    - 5 days
    - 7 days
- User may or may not enter the *Get it Now Price* field which accepts only numbers
- User chooses whether or not to check the *Returns Accepted* checkbox
- When ***List My Item*** button is clicked
    - If all fields != NULL && data entered is valid
        - **Generate** Unique && NOT NULL ItemID for entered *Item Name*
        - Perform INSERT query on the entered data
        - Redirect to **Menu** page
    - Else
        - NO database action is taken
        - Print “Please enter valid data and make sure all required fields are not empty”
- When ***Cancel*** button is clicked
    - Redirect to **Menu** page

SEARCHING FOR ITEMS

Task Decomposition



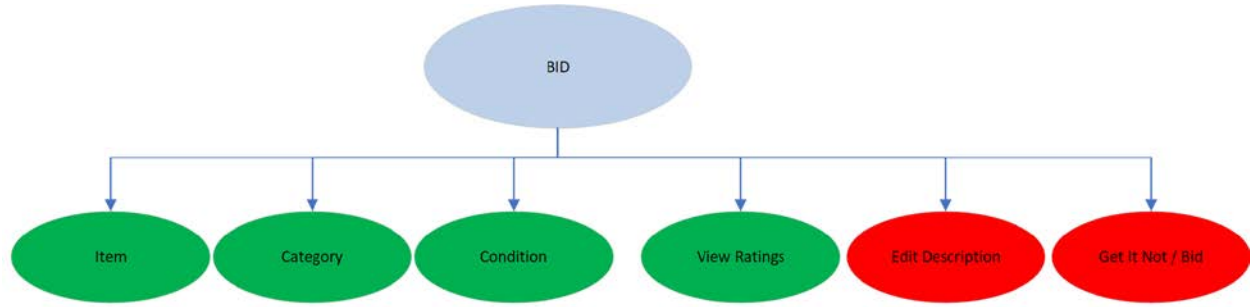
Lock-types: 3 read-only of Category, Condition, Item
Number of Locks: Several constructs needed
Enabling Conditions: User's login
Frequency: All have same high frequency
Consistency: Not critical
Subtasks: Mother task needed for coordination of subtasks. Category/Condition can be completed in parallel prior to search/cancel

Abstract Code

- User clicked on ***Search for items*** button from the **Menu**
- User can enter *Keyword* field
- User can select the *Category* field from the dropdown list
- User can input the *Minimum Price/Maximum Price* field using only numbers
- User can select the *Condition* field
- User clicks on ***Search*** button
    - If at-least 1 field != NULL
        - Execute a SELECT query from the available data which matches ALL the criteria entered
    - Else
        - Execute a SELECT * FROM ……
    - Opens up the **Search Results** page and user can see items matching with entered criteria
        - User clicks on *Item Name*
            - Redirect to **Item Description** page
        - User clicks on ***Back to Search***
            - Redirect to the **Search for Items** page
- User clicks on ***Cancel*** button
    - Redirects to the **Menu** page

BID

Task Decomposition



Lock-types: Multiple read-only and insert/update for Item and Bid.
Number of locks: multiple schema constructs needed
Enabling conditions: User's login. For the edit description task, the user must be the item owner to enable the edit description prompt to be available.
Frequency: High
Consistency: Not critical
Subtasks: Mother task needed to coordinate all subtasks. Item/Category/Condition to be completed in parallel prior to any other options to be available. Edit description task does not need to be completed and is only available to the item owner.

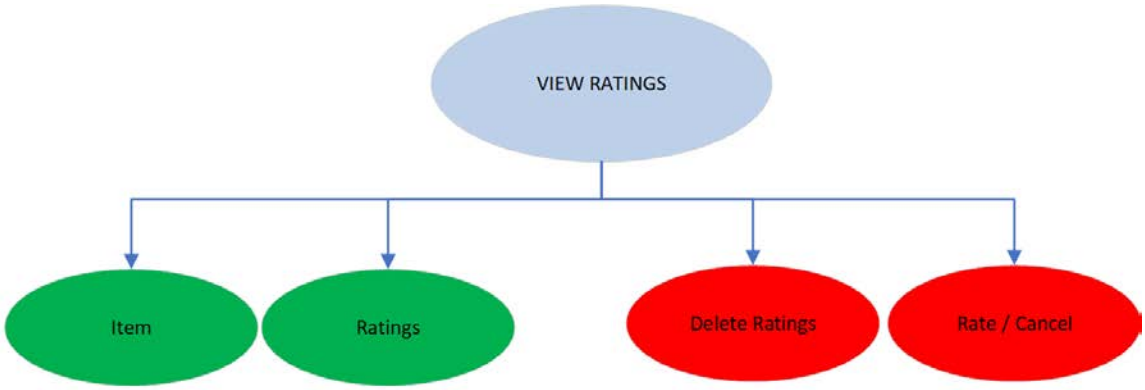
Abstract Code

- User clicked on *Item Name* in **Search Results** page
- Lists all information fields about item including the auto generated *ItemID*
- User clicks on ***View Ratings*** link
    - Redirects to **View Ratings** page
- If user is listing_user
    - User clicks on ***Edit Description***
        - Can edit description
- User clicks on ***Get It Now*** (if button has been enabled by *Get It Now* field in **List Items for Sale** page)
    - Item has been purchased
    - Auction has ended
    - Sets the user as Winner

- User can see Latest Bids – showing the latest 4 bids with the highest (and most recent) on top.
- User enters *Your Bid*
    - If amount entered is greater than last bid amount
        - No action
    - If amount entered is greater than ***Get It Now*** price
        - Print “You can choose to get the item now without bidding for the price shown in Get It Now”
    - Else
        - Print “your bid must be at least $1 higher than the last bid”
- User clicks on ***Bid On This Item***
    - User’s bid will be stored and added to the latest bids
- User clicks on ***Cancel***
    - Redirects to **Search Results**

VIEW RATINGS

Task Decomposition



Lock-types: Read-only for Item and Ratings. Insert/Update/Delete for Ratings.
Number of locks: Multiple schema constructs needed.
Enabling Conditions: Enabled by user's login. The delete ratings button is available only if the rating belongs to the logged in user.
Frequency: Moderate
Consistency: Not Critical
Subtasks: Mother task to coordinate subtasks. Delete ratings doesn't need to be completed and is only available to logged in user, if user has created a rating for a specific item.

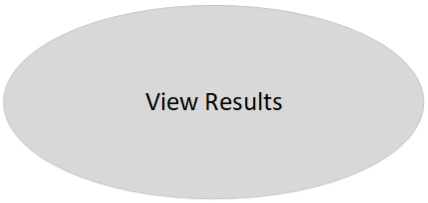
Abstract Code

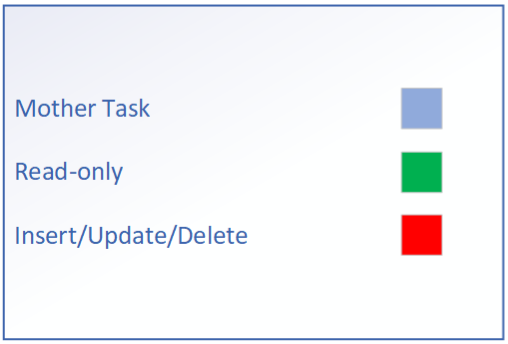
- User clicked on ***View Ratings*** in **Item Description** page
- Lists ItemId, Item Name and Average Rating for Item.
- All ratings are listed as most recent
- If user is rating_user
    - User clicks on ***Delete My Rating***
    - User's rating is deleted and Average Rating is recalculated
- User can rate with number of stars
- User can add comments in textbox

- User clicks on ***Rate this Item***
    - Rating is submitted
    - Average Rating is recalculated
    - Redirects to **Item Description** page
- User clicks on Cancel
    - No change
    - Redirects to **Item Description** page

AUCTION RESULTS

Task Decomposition





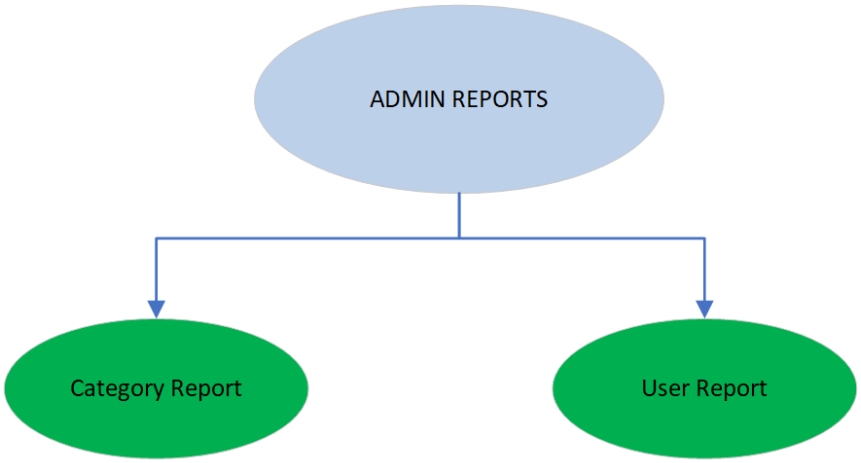
Lock-types: read-only on item/bids
Number of locks: 2
Enabling conditions: user's login
Frequency: High
Consistency: Not critical
Subtasks: mother task not needed

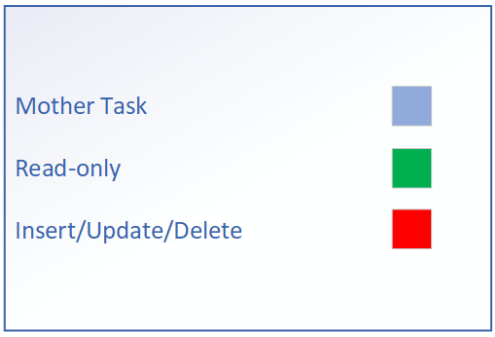
Abstract Code

- User clicked on ***View Auction Results*** in **Menu** page
- List *ItemID*, *Item Name*, Sale Price, Winner and Auction Ended fields in the screen
- If Sale Price && Winner == NULL
    - The item *Item Name*, *ItemID* did not sell
- User clicks on *Item Name*
    - Redirect to **Item Description**
- User clicks on ***Done***
    - Redirects to **Menu** page

ADMIN REPORTS

Task Decomposition





Lock-types: read-only on Item, Users, Bids, Category
Number of locks: multiple schema constructs needed
Enabling Conditions: User's login and user must be an admin.
Frequency: Moderate
Consistency: Not critical
Subtasks: mother task not needed to complete coordinate. Both tasks do not need to be completed. Admin user logged in can view either report or both.

Abstract Code

- If User = '$ADMINUSER':
    - User's **Menu** page has **"View Category Report"**, **"View User Report"** tabs
    - User clicks on ***View Category Report*** button
        - Redirects to **View Category Report**
            - User clicked on ***View Category Report*** button from **Admin Reports** page
                - List out all *Categories* of items which were already sold AND items which are still listed for sale
                    - For each *Category* list out Total Items, and the Min Price, Max Price, Average Price obtained from the ***Get It Now*** price in **List Items for Sale**
            - User clicks on ***Done*** button
                - Redirects to the **Menu** page
    - User clicks on ***View User Report*** button
        - Redirects to **View User Report**
            - User clicked on ***View User Report*** button from **Admin Reports** page
                - List out all users based on *username*
                    - Include count of : items listed for sale, items sold, items purchased, items rated
                    - Sorted on number of items listed for sale in descending order
            - User clicks ***Done*** button
                - Redirects to **Menu** page