Table of Contents:

ASACS Data Types: 3
SQL Data Types 3
ASACS Constraints: 4
Business Logic Constraints 4
Task Decomposition with Abstract Code: 9
Login 9
Task Decomp 9
Abstract Code 9
View Sites and Services 10
Task Decomp 10
Abstract Code 10
Create/Delete/Edit Services 10
Task Decomp 10
Abstract Code 11
View Meals Remaining in Inventory 11
Task Decomp 11
Abstract Code 11
View Available Rooms/Bunks 12
Task Decomp 12
Abstract Code 12
View Request Statuses 12
Task Decomp 12
Abstract Code 13
Cancel Requests 13
Task Decomp 13
Abstract Code 13
View Outstanding Requests 14
Task Decomp 14
Abstract Code 14
Modify Outstanding Requests 14

Task Decomp....14
Abstract Code....15
View Waitlist....15
Task Decomposition:....15
Abstract Code....15
Delete/Modify Entry....15
Task Decomposition:....15
Abstract Code....16
Search and View Items....16
Task Decomposition:....16
Abstract Code....17
Request Items....17
Task Decomposition:....17
Abstract Code....18
Modify Number of Items....18
Task Decomposition:....18
Abstract Code....18
Enroll Client....19
Task Decomposition:....19
Abstract Code....19
View Full Client Report....19
Task Decomposition:....19
Abstract Code....20
Modify Client Information....20
Task Decomposition:....20
Abstract Code....21
Add New Client Log Entry....21
Task Decomposition:....21
Abstract Code....21
Place on Waitlist....22
Task Decomposition:....22
Abstract Code....22

ASACS Data Types:

SQL Data Types

Table	Data Type Constraint
User	
email varchar(50)	NOT NULL UNIQUE
firstName varchar(30)	NOT NULL
lastName varchar(50)	NOT NULL
username varchar(30)	NOT NULL
password varchar(30)	NOT NULL
Client	
description varchar(100)	
firstName varchar(30)	NOT NULL
lastName varchar(50)	NOT NULL
phoneNum varchar(10)	
ClientLog	
entryDate date	NOT NULL
site varchar(60)	NOT NULL
service varchar(50)	NOT NULL
Notes varchar(100)	
fieldModif varchar(50)	
Site	
phoneNum varchar(10)	NOT NULL UNIQUE
shortName varchar(50)	NOT NULL
street varchar(50)	NOT NULL
city varchar(50)	NOT NULL
state varchar(2)	NOT NULL
zipcode varchar(10)	NOT NULL
Donation	
name varchar(50)	NOT NULL
amount int(10) unsigned	NOT NULL
category varchar(30)	NOT NULL
subCategory varchar(30)	NOT NULL
storeageType varchar(30)	NOT NULL
expDate date	NOT NULL
Request	
source varchar(50)	NOT NULL
destination varchar(50)	NOT NULL

Item varchar(50) amtRequested int(10) unsigned Status varchar(30) amtProvided int(10) unsigned	NOT NULL NOT NULL NOT NULL NOT NULL
ClientAccessible description varchar(50) hours varchar(50) conditions varchar(50) type varchar(30)	 NOT NULL NOT NULL NOT NULL NOT NULL
SoupKitchen seatsAvailable int(10) unsigned totalSeats int(10) unsigned	 NOT NULL NOT NULL
Bunk type varchar(30) isAvailble boolean	 NOT NULL NOT NULL
Room number varchar(10) isAvailble boolean	 NOT NULL NOT NULL
Waitlist rank int(10) unsigned entryDate date endDate date endReason varchar(30)	 NOT NULL NOT NULL DEFAULT NULL DEFAULT NULL

ASACS Constraints:

Business Logic Constraints

User:

- A user is an employee/volunteer of a site.
- A user must have a login account with both a username and password.
- A user must also provide a full name and email address when creating an account.
- A user must belong to one, and only one, site.
- A user has the ability to update information only about their assigned site.
- A user cannot add, edit or delete user data.
- A user, at a site with a food pantry, shelter or soup kitchen, can request items from one or more food banks.
- A user at a food bank can approve or edit requests for requests at their assigned site.
- A user may add additional services to their site, modify existing services at their site, or delete existing services at their site.

Site:

- A site entry must contain a short name, a location (street address, city, state and zip code).
- A site short name will be displayed on forms and reports to identify the site being referenced.
- A site must provide at least one, and possibly more, different types of services.
- The site must always contain at least one service, unless the DBA removes the last remaining site service.

Service:

- A service can be one of four categories: Shelter, Soup Kitchen, Food Pantry or Food Bank
- A Shelter, Soup Kitchen and Food Pantry entry must contain a description, hours of operation, and conditions of use.
- A Soup Kitchen record with contain the total seats available.
- A Soup Kitchen's total seats available can be modified to a lower number by a user of that site.
- A Soup Kitchen's total seats available can never be less than zero.

Shelter:

- A shelter's bunk count must be updatable on demand by a site's user.
- A shelter's usage will be logged and viewable by a site's user.
- A shelter's bunk count must be viewable in real-time by all on ASACS system.
- A bunk's record must have a designated type, of either: Male only, Female only or Mixed.
- A bunk cannot be reserved.
- A shelter's available room count must be updatable by a site's user, which is independent of the shelter's waitlist.
- .When a room does become unoccupied, the top family on the waitlist becomes the occupant of the room. This process does not move the available room count from zero to one to zero, but instead, leaves the available room count at zero.
- Each shelter will have its own waitlist.
- Each waitlist record will be the "head of household", which will be a link to the requesting client's client record.
- The waitlist will track when a waitlist entry has been added, and when the waitlist entry has been closed.
- A site's user can add clients to the waitlist for their site.
- A site's user can remove clients from the waitlist for their site.
- A site's user can change a client's position on the waitlist for their site.
- If a site's user changes a client's priority, the waitlist must be reordered so that no two positions contain the same priority. (Example: If priority Five is moved up to priority Two, then the previous priority Two must drop to Three, and previous Three to Four, and previous Four to Five, as so forth).
- A client may be on a waitlist for more than one shelter.
- Waitlists may only be viewed or modified by users of that site.

- Users may run reports to view waitlist size at each site, and run reports on clients to view for which sites that client is waitlisted.

Food Bank:

- Food banks provide services to other sites, not to clients directly.
- Food banks either accept donations or grant/deny requests of items.
- Food bank donations are added to the food bank's inventory and increase the amount for that item.
- Food bank requests must have an associated status.
- Food bank requests can be accepted, modified, or denied by a user of that site.
- Accepting a food bank request decreases that food bank's inventory for that item by the accepted amount.
- Modifying a food bank request is the same as accepting the request, but for a lesser amount that originally requested. This will also decrease that food bank's inventory for that item by the modified amount.
- Denying a food bank request will end the request and will not impact the shelter's inventory.
- Food bank items can be viewed and searched by all users on the ASACS system.
- Requests can be made for one or more items, and up to the maximum amount available.
- Food bank requests are stored in a queue, which is ordered by the date that the request is made.
- Food bank requests are viewed by a site's user through the "Pending Requests" report.
- Each food bank will manage its own inventory.
- Once an item leaves the food bank, the ASACS system will no longer track the item(s).

Item:

- Item donation records must have an item name, number of units, expiration date, type ("Food" or "Supplies"), subtype and storage type when entered into the system.
- For items with a type of "Food", the only subtype options will be: "Vegetables", "nuts/grains/beans", "Meat/seafood", "Dairy/eggs", "Sauce/Condiment/Seasoning" or "Juice/Drink".
- For items with a type of "Supplies", the only subtype options will be: "Personal hygiene", "Clothing", "Shelter", or "other".
- The "Meals Remaining" report must look at the Food subtypes to determine how many meals remain based on available subtypes.
- Items can be added by users at their own food bank.

Request:

- Item request records must have a source site, a destination site, item name, requested amount, and by default, will have a status of "pending".
- A request's status can change from "pending" to "closed" only if the request was filled, either fully or partially.
- Once a request is move to "closed" status, that request's record must also capture the provided amount of that item. The provided amount may be less than or equal to the requested amount, and may be greater than or equal to zero.

- If the inventory amount of an item reaches zero, and there is currently a “pending” request for that item, the pending request is to then be closed with a status of “closed” and provided amount of zero.

Client:

- A client is a person that uses the services in the system.
- A client is added to the system by a user enrolling that client into the system.
- A client record must contain their full name (first and last), an ID or description, and an optional phone number.
- A client’s record cannot be deleted by users.
- A client’s name and description fields can be modified by users.
- If a client’s name or description is modified, the previous value of name or description will be stored in the “field modified” field of a new Client Entry record. (Example: Mary Smith is married and her last name changes to Jones. When this update is made to Mary’s Client record, a new Client Entry record will be created placing her former name of Mary Smith in the “field modified” field.)

Client Entry:

- Client entry records are added by users each time a client uses a service, and these records contain a date/time stamp, the site used, a description of the service, and any additional notes.
- Client entry records are added each time a client’s name or description field is modified.
- Client entry record userform will contain current data and time in their respective fields when the record is being created.
- Client entry record userform will contain the site name of the current site in its respective field when the record is being created.
- Client entry record userform will contain the service being used in the description field when the record is being created.

Reports:

Available Bunks/Rooms

- The “Available Bunks/Rooms” report will be publicly accessible, and will display all available bunks and rooms across ASACS.
- The “Available Bunks/Rooms” report, for sites with greater than zero available bunks or greater than zero available rooms, will display the site’s name, location, phone number, hours of operation, conditions, number of bunks available (broken down to male, female, and mixed), and rooms available.
- The “Available Bunks/Rooms” report will display “Sorry, all shelters currently at maximum capacity” is there are no rooms and no bunks available.

Meals Remaining

- The “Meals Remaining” report will be publicly accessible, and displays all meals (which is determined by the amount of Food type subtypes available) currently in inventory across all food banks.
- The “Meals Remaining” report displays what type of donations are needed to provide more meals (which is determined by the amount of Food type subtypes available).

Client Search/Report

- The “Client Search” report will be privately accessible to logged in users, and will return information on a client, but only after the user has searched for the client by either name or ID number.
- The “Client Search” report cannot be searched without any input values.
- The “Client Search” report will not return any results unless there is at least one match to a client.
- The “Client Search” report will not return more than five results in any search, and if there are more than five matches in a search, the user will be prompted to provide my information to refine the search parameters.
- If the user’s search parameters return more than zero or less than or equal to five matches, then the matching client names and descriptions will be displayed to the user.
- When the user determines and decides which client result matches with the desired client, the user can run the full client report.
- The “Client Report” will be privately accessible to logged in users, and will display the client’s name/description, list of services used ordered by date in descending order, and waitlists that the client is currently on, if any.
- The user is able to modify the client’s information through the “Client Report”.
- The user is able to add a new service log entry for the client through the “Client Report”.
- The user is able to place the client on a waitlist through the “Client Report”.

Wait-list Report

- The “Wait-list Report” will be privately accessible to logged in users, and can only be run if the logged in user’s site has a shelter.
- The “Wait-list Report” will only show the waitlist for the logged in user.
- The “Wait-list Report” will, if not empty, display the wait-list rank order.
- From the “Wait-list Report” the user can delete a wait-list entry or modify a wait-list entries order.

Item Search/Report

- The “Item Search” will be privately accessible to logged in users, and this search will allow users to search items from all or one food bank.
- The “Item Search” input will support wildcard values to search for all items.
- The “Item Search” input will support filtering (expiration date, storage type, Food/Supply type, type subcategories, or item name/description) to help guide the user’s search.
- Results from the “Item Search” can be requested by the user.
- “Item Search” results requests require a requested amount.
- “Item Search” results requests will not be allowed if the requested item is at the logged in user’s site.
- “Item Search” results requests will be modifiable if the requested item is at the logged in user’s site. (Example: The logged in user finds an item and wants to modify to amount due to items exceeding their expiration. This is permitted).
- If items in inventory have reached an amount of zero, then those items will not be displayed in the “Item Search” results.

Outstanding Requests

- The “Outstanding Requests” report will be privately accessible to logged in users, and only those users associated with a food bank.

- The “Outstanding Requests” report will display all outstanding requests, and can be sorted by storage type, category, subcategory, or quantity of items requested.
- If multiple requests for an item exceed the current inventory supply if that item, the “Outstanding Requests” report will indicate the shortfall of the items by highlighting those requested item amounts in red.
- From the “Outstanding Requests” report the user will have the ability to mark requests as fulfilled in full, or reduce the items provided in a request and mark the request as fulfilled, or mark the request as unable to be filled.

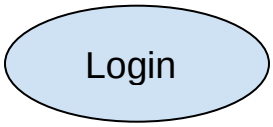
Request Status

- The “Request Status” report will be privately accessible to logged in users, and displays all requests made by the user and their statuses.
- The “Request Status” report will display “closed” requests with the actual number of items provided.
- From the “Request Status” report the user will be able to delete any outstanding requests that they previously filed.

Task Decomposition with Abstract Code:

Login

Task Decomp



Lock Types: Read-only on User table
Enabling Conditions: None
Frequency: Around 100 logins per day
Schemas: Only one, the User schema
Consistency: not critical, order is not critical
Subtasks: Mother task is not needed, no decomposition is needed.

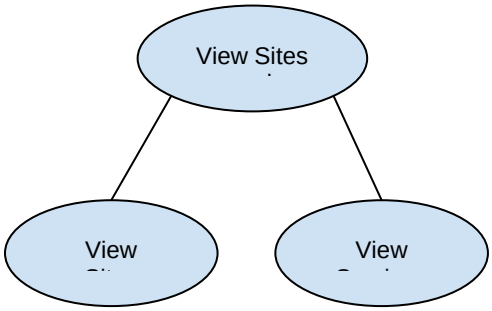
Abstract Code

- User enters *username* ($Username), *password* ($Password) into input fields
- If data validation is successful for both *username* and *password* input fields then:
    - When ***Enter*** button is clicked:
        - If User.username is found but User.password != ‘$Password’:
            - Go back to **Login** form, with error message.
        - Else:
            - Store login information as session variable $Username.

- Go to **View Sites and Services** form.
- Else *username* and password fields are invalid, display **Login** form, with error message.

View Sites and Services

Task Decomp



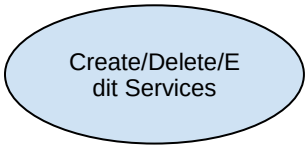
Lock Types: 2 read-only lookups of Sites and Services for a User
Enabling Conditions: Both are enabled by a user's login
Frequency: Lookups have the same frequency, moderate frequency as this is the landing page after login.
Schemas: 2 schemas are needed
Consistency: not critical, as only the user sees this information
Subtasks: Due to lookups across multiple schemas a Mother task is required. These tasks can be done in parallel.

Abstract Code

- Run the ***View Sites and Services*** task based on $Username.
- Find the current User using the User $Email; Display User's FirstName.
- Find the Site for the User; Display Site ShortName, Address and PhoneNum.
- Find each of the Services for the User; Display: Service, Description, Hours and Condition.

Create/Delete/Edit Services

Task Decomp



Lock Types: Lookups of Services of a User (use **View Sites and Services** task). Lookups of Services list. The ability to read, insert, delete and update Services list.
Enabling Conditions: A User's login and edit Services request.

Frequency: One frequency, expected to be low.
Schemas: One schema needed.
Consistency: Not critical, as no one else should be viewing page besides User.
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- **View Sites and Services** task.
- If User clicks ***Create Service*** button; Display an input text field to enter a new Service. Click ***Save*** button. Go to **View Sites and Services** form.
- If User clicks ***Delete Service*** button; Checkboxes appear to select Service(s) for deletion. Click ***Delete*** button. Click ***Save*** button. Go to **View Sites and Services** form.
- If User clicks ***Edit Service*** button; Display text box to edit Service name. Click ***Save*** button. Go to **View Sites and Services** form.

View Meals Remaining in Inventory

Task Decomp

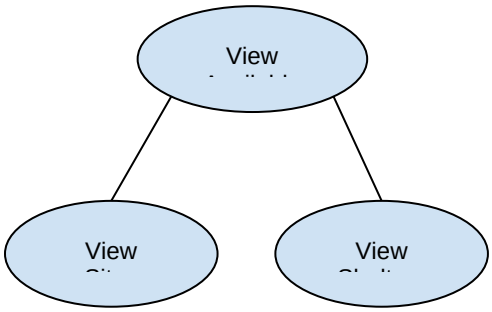
Lock Types: Read-only lookup on FoodBank table.
Enabling Conditions: Enabled by navigating to this form from the **Login** form and clicking link to the **Meals Remaining** form that does not require user login.
Frequency: High frequency, due to the fact that anyone can view this form even without being a user.
Schemas: One schema needed.
Consistency: Not critical, order is not critical.
Subtasks: Mother task is not needed. No decomposition is needed.

Abstract Code

- User or site visitor navigates to **Login** form.
- User or site visitor clicks link to **Meals Remaining** form.
- Display FoodBank inventory, because user or site visitor is not logged-in, the inventory across all food banks is displayed.

View Available Rooms/Bunks

Task Decomp



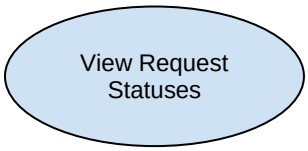
Lock Types: Read-only lookups on Sites and Shelters.
Enabling Conditions: Enabled by navigating to this page from the **Login** form and clicking link to the **Available Bunks/Rooms** form that does not require user login.
Frequency: High frequency, due to the fact that anyone can view this form, even without being a user.
Schemas: Two schemas needed.
Consistency: not critical, order is not critical.
Subtasks: Mother task is not needed. No decomposition is needed.

Abstract Code

- User or site visitor navigates to **Login** form.
- User or site visitor clicks link to **Available Bunks/Rooms** form.
- If Shelter Rooms Available and/or Bunks Available > 0
    - Display Shelter name, Site Location, Site Phone Number, Shelter Hours, Shelter Conditions, Number of Available Bunks by Bunk type, Number of Available Rooms
- Else Display message “Sorry, all shelters are currently at maximum capacity.”

View Request Statuses

Task Decomp



Lock Types: Lookup read-only Requests for a User
Enabling Conditions: User login
Frequency: Low-high, depending on user’s need for item - may check request status often to see if it changed.
Schemas: One schema needed.

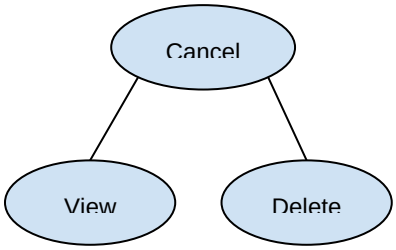
Consistency: Not critical.
Subtasks: No Mother task is needed.

Abstract Code

- Find the current User using the User Username.
- Find the Requests for the User; Display: Requests, Amount Requested, and Request Status.
- If Request Status == Closed
    - Display Amount Provided

Cancel Requests

Task Decomp



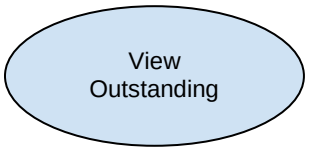
Lock Types: Lookup of Requests for a User (use **View Request Statuses** task). Read, delete, and update Requests.
Enabling Conditions: User login and request to cancel outstanding requests on **Request Status** form.
Frequency: Same frequencies.
Schemas: One schema needed.
Consistency: Not critical.
Subtasks: Mother task needed due to multiple operations.

Abstract Code

- User executes the **View Request Statuses** task.
- User clicks the ***Cancel Requests*** button.
- User selects the requests for cancellation and confirms the transaction
    - If user decides not to cancel the requests, clicks the ***Cancel*** button.
- Request is removed from system, user returns to updated **Request Status** form.

View Outstanding Requests

Task Decomp



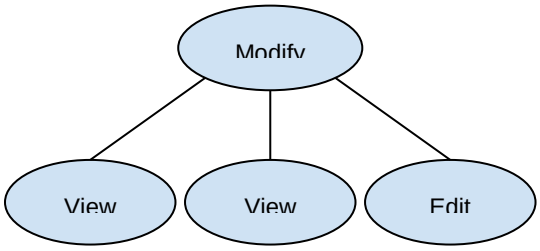
Lock Types: Lookup read-only Requests for a User associated with a Food Bank
Enabling Conditions: User login and one of the User's Sites has a Food Bank with at least one request
Frequency: Varies, but most likely a user at a Food Bank will view its requests often
Schemas: One schema needed.
Consistency: Not critical, even if someone is canceling a request for an item.
Subtasks: No Mother task is needed.

Abstract Code

- Find the current User using the User Username.
- Find the Requests associated with a User's Site that has a Food Bank; Display: Item, Storage type, Category and Amount Requested.

Modify Outstanding Requests

Task Decomp



Lock Types: Lookup of a User's requests for Items (use **View Request Statuses** task) and lookup FoodBank Inventory. Read and update Requests. Insert Request statuses.
Enabling Conditions: User login, user is associated with a Food Bank and has requests for items.
Frequency: Different frequencies.
Schemas: 2 schemas, FoodBank and Requests.
Consistency: Yes consistency matters to ensure the inventory is up to date so the User can accurately address requests.
Subtasks: Due to lookups across multiple schemas a Mother task is required.

Abstract Code

- Run **View Outstanding Requests** task.
- Find Food Bank Inventory; Display Food Bank's inventory as a table.
- User clicks ***Modify Requests*** button to edit the requests.
- If user can fulfill the request in full
    - Check the ***Fulfilled in Full*** checkbox.
- If the User can only partially fulfill the request
    - Enter the number of items provided and check the ***Partially Fulfilled*** checkbox.
- Else check the ***Unable to Fulfill*** checkbox.

View Waitlist

Task Decomposition:

Lock Types: Lookups are all read-only
Enabling Conditions: None
Frequency: Low
Schema: One schema needed
Consistency: Important
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Find relevant shelter based on ShortName and PhoneNum
- If Waitlist is empty, print that waitlist is empty.
- For each ClientID in the waitlist:
    - Print the ClientID and Rank in a list sorted by Rank
    - Add ***Delete, Move Up, Move Down*** buttons for each row that trigger **Delete/Modify Entry** task

Delete/Modify Entry

Task Decomposition:

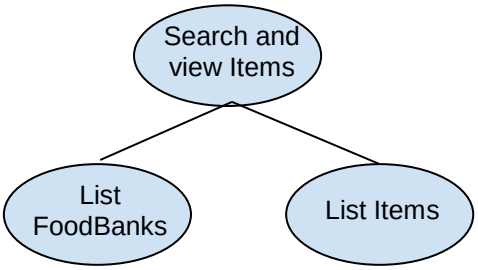
Lock Types: Locks needed on Rank for each waitlist entry
Enabling Conditions: Delete, Move up, or Move down button clicked
Frequency: Low
Schema: Two schema needed, Shelter and Waitlist
Consistency: Important
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- If ***Delete*** button clicked:
    - Remove current row from waitlist table
    - For each ClientID with Rank higher than deleted entry's Rank:
        - Decrease rank by 1
    - Call **View Waitlist** task to refresh page
- If ***Move Up*** button clicked:
    - Decrement rank for current row by 1
    - Increment rank for above row by 1
    - Call **View Waitlist** task to refresh page
- If ***Move Down*** button clicked:
    - Increment rank for current row by 1
    - Decrement rank for below row by 1
    - Call **View Waitlist** task to refresh page

Search and View Items

Task Decomposition:



Lock Types: Lookups are read-only
Enabling Conditions: Site current User is associated with
Frequency: Low

Schema: Two schema needed, FoodBank and Item
Consistency: Not essential, order of results for item lookup not important
Subtasks: Mother task is not needed. Lookup of FoodBanks must be done first

Abstract Code

- Retrieve list of all FoodBanks in $FoodBankSelection input field and display in drop down menu, with added "All" option.
- Add input fields to filter items by $ExpirationDate, $StorageType, $FoodOrSupply, $Category, and $Keyword search for name / description.
- When user clicks ***Search*** button:
    - If individual $FoodBankSelection selected, retrieve list of items from that FoodBank matching filters
    - Otherwise, retrieve list of items from all FoodBanks matching filters
    - For each item returned:
        - Display item Name, Amount, Category, StorageType, ExpirationDate
        - If Item belongs to User's FoodBank:
            - Add button to ***Modify Number of Items***
        - Otherwise, add button to ***Request Items***
- If ***Request Items*** is clicked:
    - Call **Request Items** task
- If ***Modify Number of Items*** is clicked:
    - Call **Modify Number of Items** task

Request Items

Task Decomposition:

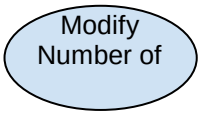
Lock Types: Inserts being performed on Requests schema
Enabling Conditions: Site current user is associated with, ***Request Items*** clicked, and the FoodBank's record for that item
Frequency: Low
Schema: Two schema needed, FoodBank and Item
Consistency: Essential, since inserts being performed, must have accurate info. Request does not have to be performed at high priority.
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Create pop-up with input form for $Quantity of item requested.
- If User clicks ***Ok***:
    - If quantity requested exceeds total quantity available at FoodBank:
        - Display error message
    - Otherwise, Insert a request entry containing Source FoodBank, Destination FoodBank, Item, Quantity Requested, Status = “Pending”, AmountProvided = 0
        - Close **Requests** pop-up
- If User clicks ***Cancel:***
    - Close **Requests** pop-up

Modify Number of Items

Task Decomposition:



Lock Types: Update being performed on Item schema
Enabling Conditions: Site current user is associated with, ***Modify Number of Items*** clicked, and the FoodBank’s record for that item
Frequency: Low
Schema: Two schema needed, FoodBank and Item
Consistency: Essential, since inserts being performed, must have accurate info. Request does not have to be performed at high priority.
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Create pop-up with input form for $NewQuantity of item being updated.
- If User clicks ***Ok***:
    - If $NewQuantity == 0:
        - Delete entry for that item from FoodBank
    - Otherwise, update Amount for that item to $NewQuantity
    - Close **Modify Number of Items** pop-up
- If User clicks ***Cancel:***
    - Close **Modify Number of Items** pop-up

Enroll Client

Task Decomposition:

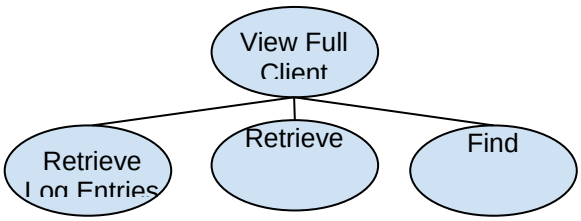
Lock Types: Insert being performed on Client
Enabling Conditions: None
Frequency: Moderate
Schema: One schema needed, Client
Consistency: No strict consistency requirement, Insert into client table not priority
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Display input fields for $ClientID, $FirstName, $LastName, $PhoneNum, $Description
- If User clicks ***Add Client*** button:
    - If not all fields filled in with valid data:
        - Display error message next to incorrect field
    - Otherwise, Insert new Client using information from fields $ClientID, $FirstName, $LastName, $PhoneNum, $Description
        - Trigger **Add New Client Log Entry** task with Service as “Enroll”

View Full Client Report

Task Decomposition:



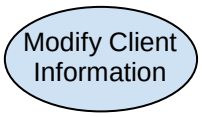
Lock Types: All lookups are read-only
Enabling Conditions: None
Frequency: Moderate
Schema: Three schema needed, Client, ClientLogE, Waitlist
Consistency: No strict consistency requirement, Insert into client table not priority
Subtasks: Mother task is not needed. Retrieve client information must be done first, other Sub-tasks can be completed in parallel.

Abstract Code

- Display input field for $NameOrIdNumber
- If ***Search*** button is clicked:
    - Perform query to find Clients with Name or ClientID LIKE $NameOrIdNumber
    - Count total number rows returned. If count == 0
        - Display error message stating no Clients found matching search string
    - Else if count > 5:
        - Display error message stating search term too general
    - Else, display similar search results, User clicks on desired Client from results :
        - Display all rows in Client table for matching ClientID
        - Perform query to find ClientLogE entries with matching ClientID
        - Perform query to find Waitlist entries with matching ClientID
        - Display all fields in each ClientLogE returned, sorted by newest first
        - Display all Shelter ShortNames for which the ClientID is waitlisted, and their Rank.
        - Display ***Modify Client's Information*** button
        - Display ***Add New Client Log Entry*** button
        - Display ***Place on Waitlist*** button
    - If ***Modify Client's Information*** button is clicked:
        - Trigger **Modify Client's Information** task
    - If ***Add New Client Log Entry*** button is clicked:
        - Trigger **Add New Client Log Entry** task
    - If ***Place on Waitlist*** button is clicked:
        - Trigger **Place on Waitlist** button
- If ***Clear*** button is clicked:
    - Redraw new **Client Search and Results Form**

Modify Client Information

Task Decomposition:



Lock Types: Update being performed on Client
Enabling Conditions: ClientID being updated passed from **Client Report Search Form**
Frequency: Low
Schema: One schema needed, Client

Consistency: Essential. Fields must be populated with current data so they are not incorrectly modified.
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Make fields editable for for $FirstName, $LastName, $PhoneNum, $Description
- If User clicks ***Save Client Information*** button:
    - If not all fields filled in with valid data:
        - Display error message next to incorrect field
    - Otherwise, Update Client matching ClientID using information from fields $FirstName, $LastName, $PhoneNum, $Description
        - Trigger **Add New Client Log Entry** task with Service as "Updated Client Information."
- If User clicks ***Cancel*** button:
    - Make fields no longer editable and display their original values

Add New Client Log Entry

Task Decomposition:

Lock Types: Insert being performed on ClientLogE
Enabling Conditions: ClientID and name being updated passed from **Client Report Search Form**
Frequency: Slightly less frequent than **View Full Client Report**
Schema: One schema needed, ClientLogE
Consistency: No strict consistency requirement, Insert into ClientLogE table not priority
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Show new window with editable fields for $Site, $Service, and $Notes.
- $EntryDate field will automatically be filled in. Display ClientID and Name at top of window.
- If User clicks ***Add Log Entry*** button:
    - If not all fields filled in with valid data:

        - Display error message next to incorrect field
    - Otherwise, Insert ClientLogE for ClientID using information from fields $Site, $Service, $Notes, $EntryDate

- If User clicks ***Cancel*** button:
    - Close window without inserting new UserLogE. Return to previous form

Place on Waitlist

Task Decomposition:

Lock Types: Insert being performed on Waitlist
Enabling Conditions: ClientID, Name, and current Shelter passed from **Client Report Search Form**
Frequency: Much less frequent than **View Full Client Report**
Schema: One schema needed, Waitlist
Consistency: No strict consistency requirement, Insert into Waitlist table not priority
Subtasks: Mother task is not needed. No decomposition needed.

Abstract Code

- Display confirmation box asking User if they would like to add the ClientID with Name to Waitlist for Shelter that User is signed in to.
- If User clicks ***Add Client To Waitlist*** button:
    - Insert a new waitlist entry with ClientID provided by **Client Report Search Form**, Rank and EntryDate automatically determined by table rules
    - Display Message that Client Name was added to Waitlist successfully
- If User clicks ***Cancel*** button:
    - Close confirmation box