Table of Contents:

Datatypes

Household

Attribute	Data Type	Nullable
Email	String	Not Null
SquareFootage	Integer	Not Null
HouseholdType	String	Not Null

Heating

Attribute	Data Type	Nullable
HeatingSetting	Integer	Not Null

Cooling

Attribute	Data Type	Nullable
CoolingSetting	Integer	Not Null

On-the-Grid

Attribute	Data Type	Nullable
PublicUtility	List<String>	Not Null

Location

Attribute	Data Type	Nullable
PostalCode	String	Not Null
City	String	Not Null
State	String	Not Null
Latitude	Decimal	Not Null
Longitude	Decimal	Not Null

Appliances

Attribute	Data Type	Nullable
ApplianceID	Integer	Not Null
Appliancetype	String	Not Null
Manufacturer	String	Not Null
Appliance Model	String	Null

BTU	**Integer**	**Not Null**

Manufacturer

Attribute	Data Type	Nullable
ManufacturerName	**String**	**Not Null**

Air Conditioner

Attribute	Data Type	Nullable
RPM	**Integer**	**Not Null**
EER	**Decimal (10)**	**Not Null**

Heater

Attribute	Data Type	Nullable
RPM	**Integer**	**Not Null**
EnergySource	**String**	**Not Null**

HeatPump

Attribute	Data Type	Nullable
SEER	**Decimal (10)**	**Not Null**
HSPF	**Decimal (10)**	**Not Null**

WaterHeater

Attribute	Data Type	Nullable
TankSize	**Decimal (10)**	**Not Null**
CurrentTemperature	**Integer**	**Null**
EnergySource	**String**	**Not Null**

PowerGenerator

Attribute	Data Type	Nullable
GeneratorID	**Integer**	**Not Null**
GenerationType	**String**	**Not Null**
AverageMonthlyKilowattHoursGenerated	**Integer**	**Not Null**

BatteryStorageCapacityInKilowattHours	**Integer**	**Not Null**

Business Logic Constraints

User

- Anyone will be able to submit their data and can browse the selected set of reports available.
- One email can you be registered once in the system

Household information

- The user will not have an option to go back and/or be able to change data they have previously entered.
- If these validations fail, an appropriate error message should be displayed.
  - Entering an invalid postal code that is not on the provided list should be rejected.
  - This listing of postal codes will not change.
  - If the email address already exists in the database.
  - A heating/cooling thermostat entry is required unless the user has indicated they do not have heating or cooling.

Appliance

- The user can enter as many appliances as they want
- Appliance can be deleted by the user
- The user for a newly added household should first be shown the “add appliance” form.
- The user for a newly added household should be shown the “Add power generation” form.
- If a house is “off-the-grid”, no option for skipping “Add power generation” should be presented.
- If all generators are deleted and the household is “off-the grid”, the user cannot leave “Power generation listing” form until at least one generator has been added.

Report

- In the event a report does not return any results, an appropriate message should be displayed instead of a blank page or an empty table.
- Any instances where a NULL value might be returned should be replaced with an empty string
- If a sort order is not specified as ascending or descending, then ascending order is implied.
- If a number is rounded, unless otherwise specified, it should follow the “half rounds up” method.

- If a report definition asks to limit the number of rows returned from a larger set of sorted results, allow the DBMS to arbitrarily choose that subset, with no more than the specified number of rows returned - "tie-breaking" to determine which rows are shown is not required.
- Top 25 popular manufacturers
  - All possible appliance types should be present in the table even if the manufacturer does not have any appliances belonging to one or more appliance types.
- Heating/cooling method details
  - All household types should be displayed even if a household type does not have one or more heating/cooling method(s) associated with it.
- Water heater statistics by state
  - If there are no water heaters and/or households for a state, the state should be displayed on this report with blank values for all statistical columns.
  - All energy sources should be displayed even if the selected state has no water heaters utilizing that energy source
- Off-the-grid household dashboard
  - All household types must be displayed even if that household type has no off-the-grid households
  - All appliance types should be displayed; if there is no data for a specific appliance type then all values for that appliance type should be displayed as zeroes.
- Household averages by radius
  - The postal code input should be validated, and if invalid, an appropriate error message displayed.
  - All averages should be calculated using only households which have corresponding attributes. If an attribute is not present for all of the households within the search radius, no value should be displayed for that attribute.

Task Decomposition with Abstract Code

Main Menu

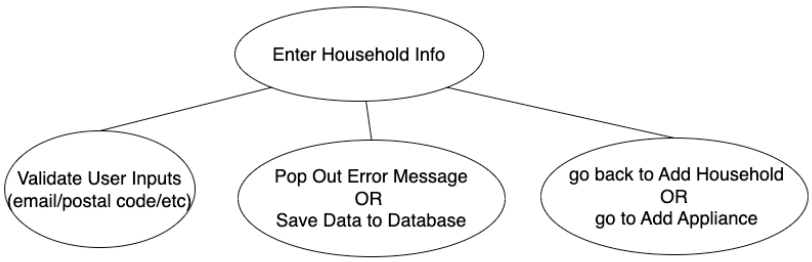
Task Decomposition

- Show "***Enter my household info***" and "***View Reports/Query Data***" buttons.
- Enabling condition: Triggered after accessing the web
- Frequency: Form has high frequency; Fields in form have the same frequency
- Database interaction: No read/write/update/delete/insert from database
- Lock Types: No lock
- Sequence of subtask: No subtask, no decomposition is needed
- Consistency: Not critical given there is no update of data

Abstract Code

- Show "***Enter my household info***" and *"**View Reports/Query Data**"* buttons.
- If "***Enter my household info***" button is pushed, perform **Enter Household Info** task
- If "***View Reports/Query Data***" button is pushed, perform **View Report** task

Enter Household Info



Task Decomposition

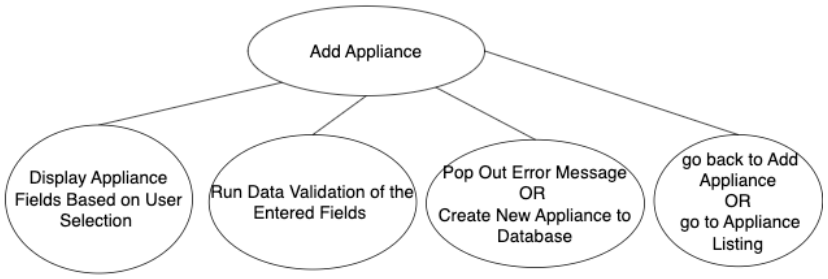
- Enabling condition: Triggered after user pressed the "*Enter my household info*" button
- Frequency: Form has high frequency; Fields in form have the same frequency
- Database interaction: Read and Insert from HOUSEHOLD table, Read from LOCATION table
- Lock Types: 2 read locks on HOUSEHOLD and LOCATION, 1 write lock on HOUSEHOLD
- Sequence of subtask: Mother task is needed. Order is necessary

    - Validate User Inputs -> Pop Out Error Message Window or Save Data To Household Table -> go back to Add Household or go to Add Appliance
- Consistency: not critical, assume there won't be another user entering info using the same address

Abstract Code

- If "***Next***" button is pushed,
    - **Validate User Inputs** as below:
        - If the *email* input already exists in HOUSEHOLD table:
            - Error message includes "Email already exists!"
        - If the *postal code* doesn't exist in LOCATION table:
            - Error message includes "Postal code is not correct!"
        - If data validation of *Home Type, Square footage* (whole number) doesn't pass:
            - Error message includes "Please enter valid Home Type/Square footage"
        - If "No Heat" box is not ticked, and data validation of *Thermostat setting for heating* (as a whole degrees Fahrenheit) doesn't pass:
            - Error message includes "Please enter valid thermostat setting for heating"
        - If "No Cooling" box is not ticked, and data validation of *Thermostat setting for cooling* (as a whole degrees Fahrenheit) doesn't pass:
            - Error message includes "Please enter valid thermostat setting for cooling"
    - If there is error message, **Pop Out Error Message Window and Go Back to Enter Household**
    - If there is no error, **Save Data To Household Table** and **Go To Add Appliance**

Add Appliance



Task Decomposition

- Enabling condition: Triggered after user pressed the "***Next***" button in Enter Household Info form
- Frequency: Form has high frequency as all household must have at least one APPLIANCE; Different fields' frequency can be different, dependent on which Appliance/Method user selects
- Database interaction: Read from HOUSEHOLD table, Multiple times of Insert into APPLIANCE and MANUFACURER table
- Lock Type: 1 read lock for HOUSEHOLD table, 2 write lock for APPLIANCE and MANUFACURER table
- Sequence of subtask: Mother task is needed since there is order of sequence in subtask, **Display Appliance Fields Based on User Selection** -> **Run Data Validation of the Entered Fields -> (Pop Out Error Message Window and Go Back to Add Appliance)** or (**Create new Appliance to APPLIANCE database** and go **to Appliance Listing)**
- Consistency: critical, data should be persisted to the database when saving from the screen

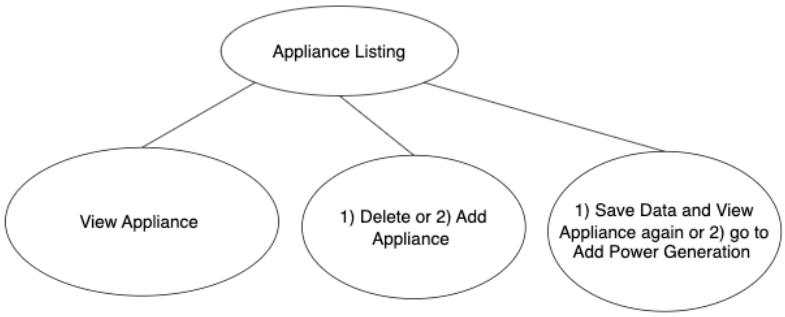
Abstract Code

- **Display Appliance Fields Based on User Selection**
    - If user selects *Air Handler*
        - Displays fields of *manufacturer, model name, BTU and Fan RPMs*
        - Displays heating/cooling method
            - If user selects *air conditioner,* displays *EER*
            - If user selects *heater*, displays *energy source*
            - If user selects *heat pup*, displays *SEER and HSPF*
    - If user selects *Water Heater*

- Displays fields of *manufacturer*, model name, BTU, tank size, current temperature setting and energy source

- If "***Add***" button is pushed,
  - **Run Data Validation of the Entered Fields**
    - If *BTU* doesn't exist or not in whole number, add to error message
    - If *Manufacturer or Model Name* is Null, add to error message
    - If *Air Handler* is selected
      - if *RPM* is Null or not in whole number, add to error message
      - If *air conditioner* is selected, *EER* is Null or not in decimal number to the tenth decimal point, add to error message
      - if *heater* is selected and *energy source* is Null, add to error message
      - if *heat pup* is selected, *SEER or HSPF is Null* or not in decimal number to the tenth decimal point, add to error message
    - If *Water heater* is selected,
      - if t*ank size* is Null *or* not in decimal number to the tenth decimal point, add to error message
      - if *current temperature setting* is not Null but not in whole number, add to error message
      - If *energy source* is Null, add to error message
    - If there is error message, **Pop Out Error Message Window** and **Go Back to Add Appliance**
    - If there is no error, based on user inputs, **Create new Appliance to APPLIANCE database** and **go to Appliance Listing**

<u>Appliance Listing</u>



Task Decomposition

- Enabling condition: Triggered after user pressed the "***Add***" button in Add Appliance form
- Frequency: Form has high frequency, appear each time after a new appliance is added; Each field has the same frequency
- Database interaction: Read/Delete from APPLIANCE table
- Lock Type: 1 read and 1 write lock for APPLIANCE table
- Sequence of subtask: Mother task is needed since there is subtask sequence, **View Appliance** -> **1) Delete or 2) Add Appliance** -> **1) Save Data and View Appliance again or 2) go to Add Power Generation**
- Consistency: critical, data should be persisted to the database when saving from the screen

Abstract Code

- **View Appliance**
    - Query the Appliance table and display a list of appliances with info of Type, Manufacturer and Model
- If **"*delete*"** button is pushed, the related record is to **Delete Appliance** from Appliance table
- If **"*Add another appliance*"** button is pushed, go to **Add Appliance**
- If **"*Next*"** button is pushed, go to **Add Power Generation**

<u>Add Power Generation</u>

Task Decomposition

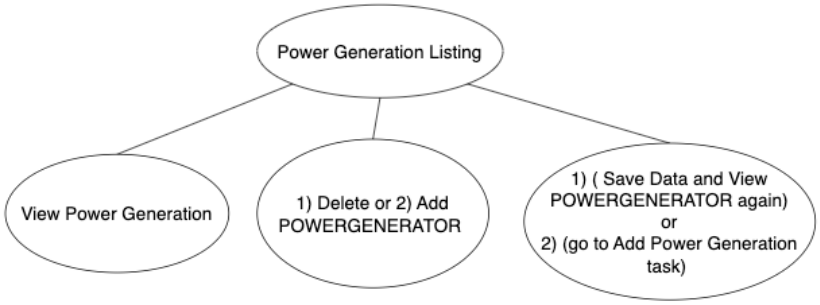
- Enabling condition: Triggered after user pressed the "***Next***" button in Appliance Listing form
- Frequency: Form has medium frequency as not all households have power generator; Fields in the form have the same frequency
- Database interaction: Read from HOUSEHOLD table, Insert into POWERGENERATOR table
- Lock Type: 1 read lock for HOUSEHOLD table, and 1 write lock for POWERGENERATOR table

- Sequence of subtask: Mother task is not needed. Decomposition is not needed
- Consistency: not critical

Abstract Code

- If user presses ***Skip*** button
    - If public utility in HOUSEHOLD table is empty, then go to Power Generation Listing form
    - Else: pop up Error Window "Please enter power generation information"
- If user presses ***Add button***
    - If *generation type* is not selected, add to error message
    - If *average monthly kilowatt hours* is Null or not in whole number, add to error message
    - If *battery storage capacity* is not Null but not in whole number, add to error message
- If there is error message, Pop Out Error Message Window and Go Back to Add Power Generation
- If there is no error, based on user inputs, Save Data to POWERGENERATOR database and go to Power Generation Listing

<u>Power Generation Listing</u>



Task Decomposition

- Enabling condition: Triggered after user pressed the "***Add***" button in Add Power Generation form
- Frequency: The form has medium frequency, appear each time after a new power generator is added but not all households have power generator; Each field has the same frequency
- Database interaction: Read/Delete from POWERGENERATOR table, Read from HOUSEHOLD table

- Lock Type: 1 read and 1 write lock for POWERGENERATOR table, 1 read lock for HOUSEHOLD table
- Sequence of subtask: Mother task is needed since there is subtask sequence, **View Power Generation** -> **1) Delete or 2) Add POWERGENERATOR** -> **1) (Save Data and View POWERGENERATOR again) or 2) (go to Add Power Generation task)**
- Consistency: critical, data should be persisted to the database when saving from the screen

Abstract Code

- **View Power Generation**
    - Query the POWERGENERATOR table and display a list of Power Generation with info of Type, Monthly Kwh and MoBattery kWh
- If **"*delete*"** button is pushed, the related record is to **Delete Power Generation** from POWERGENERATOR table
- If **"*Add more power*"** button is pushed, go to Add Power Generation form
- If **"*Finish*"** button is pushed,
    - If there is no power added and the public utility is not empty in HOUSEHOLD table
        - Pop up Window to remind adding power, then go back to Add Power Generation form
    - else:
        - Go to Wrapping Up form

<u>Wrapping Up</u>

Task Decomposition

- Enabling condition: Triggered after user pressed the "***Finish***" button in Power Generation Listing for
- Frequency: Form has high frequency, fields have the same frequency
- Database interaction: No read/write/update/delete/insert from database
- Lock Types: No lock
- Sequence of subtask: No subtask, no decomposition is needed
- Consistency: Not critical given there is no update of data

Abstract Code

- Show "***Return to the main menu"*** button.
- If "***Return to the main menu***" button is pushed, perform **Main Menu** task

View reports

Task Decomposition

- Enabling condition: Triggered after user pressed the "***View Reports/Query Data*"** button in the Main Menu page
- Frequency: Form has high frequency; Fields have the same frequency
- Database interaction: No read/write/update/delete/insert from database
- Lock Types: No lock
- Sequence of subtask: No subtask, no decomposition is needed
- Consistency: Not critical given there is no update of data

Abstract Code

- Show links for the following sections: ***Top 25 popular manufacturers, Manufacturer/model search, Heating/cooling method details, Water heater statistics by state, Off-the-grid household dashboard, Household averages by radius***
- If each of the link is pushed by the user, perform the corresponding task with the same name
    - *e.g.*, if "***Top 25 popular manufacturers****"* is pushed, perform **Top 25 popular manufacturers**
- If "***Finish***" button is pushed, perform **Main Menu** task

Top 25 popular manufacturers

Task Decomposition

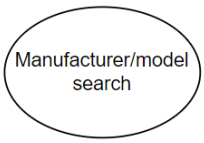
- Enabling condition: Triggered after user pressed the "***Top 25 popular manufacturers***" button in View Reports form

- Frequency: Form frequency depends on user preference, if users view each form with the same frequency, then top twenty-five manufacturers form has medium frequency, the drilldown forms have low frequency; Fields in top twenty-five manufacturers report have higher frequency than fields in drilldown report; Fields in the same report have the same frequency
- Database interaction: Read from APPLIANCE and MANUFACTURER table
- Lock Types: 1 read lock from APPLIANCE and 1 read lock from MANUFACTURER table
- Consistency: Not critical given there is no update of data
- Sequence of subtask: Only **List the top twenty-five manufacturers** task is performed initially. For **Drilldown report** of any certain manufacturer (required by users, can be multiple times), it is optional and independently re-triggered by user once at a time (by clicking at the button for the specific manufacturer). Therefore, those two sub tasks are not dependent, not simultaneously triggered and the mother task is not needed to coordinate them.

Abstract Code

- **List the top twenty-five manufacturers**
    - Query the APPLIANCE table and MANUFACTURER table (with APPLICANCE left joined by MANUFACTURER via common key ManufacturerName)
    - Display the list of top 25 manufacturers with the most appliances, with columns of manufacturer name, raw count presented
    - At each row show a button ***drilldown report*** to enable the subtask **drilldown report**
    - If ***drilldown report*** pushed by the user, perform **drilldown report** for the specific manufacturer in that row
- **Drilldown report**
    - Query the APPLIANCE table and MANUFACTURER table
    - Display manufacturer name at the top
    - Display a table, with columns of appliance type, and appliances count presented
- If "***Finish***" button is pushed, perform **View Reports** task

Manufacturer/model search



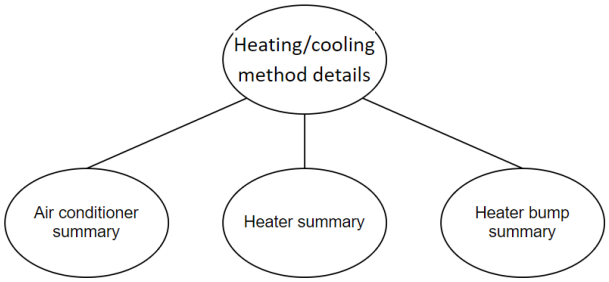
Task Decomposition

- Enabling condition: Triggered after user pressed the "***Manufacturer/model search***" button View Reports form
- Frequency: Form frequency depends on user preference, if users view each form with the same frequency, then this form has medium frequency; Fields in this form have the same frequency
- Database interaction: Read from APPLIANCE and MANUFACTURER table
- Lock Types: 1 read lock from APPLIANCE and 1 read lock from MANUFACTURER table
- Consistency: Not critical given there is no update of data
- Sequence of subtask: No subtask, no decomposition is needed

Abstract Code

- User enters a string of *keyword* in the input field
- Query the APPLIANCE table and MANUFACTURER table (with APPLIANCE left joined by MANUFACTURER via common key ManufacturerName), search for the entries where *keyword* is a substring of either the Model or ManufacturerName attribute
- Display result with column of Model and ManufacturerName , ordered by manufacturer name ascending and model name ascending
- Highlight the any substring containing *keyword* in both columns (if any) of Model and ManufacturerName
- If "***Finish***" button is pushed, perform **View Reports** task

Heating/cooling method details



Task Decomposition

- Enabling condition: Triggered after user pressed the "***Heating/cooling method details***" button in View Reports form

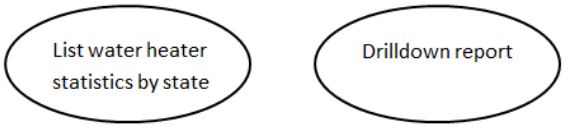
- Frequency: Form frequency depends on user preference, if users view each form with the same frequency, then this form has medium frequency; Fields in this form have the same frequency
- Database interaction: Read from HOUSEHOLD, APPLIANCE, (AIR HANDLERS, AIR CONDITIONER, HEATER, HEAT PUMP)
- Lock Types: Read-only lock for each of the table above
- Sequence of subtask: All tasks must be done but can be done in parallel. Mother task is required to coordinate subtasks. Order is not necessary. Below is the list of subtasks:
    - **Air Conditioner Summary**
    - **Heater Summary**
    - **Heat bump Summary**
- Consistency: Not critical given there is no update of data

Abstract Code

- Query the HOUSEHOLD table and APPLIANCE table (with APPLIANCE left joined by HOUSEHOLD via common key Email)
- group by the same household type, display household_type
- **Air Conditioner Summary**
    - for each household type, find all the appliances that belong to air conditioner, calculate sum of rows as count_ac, average(BTU) as average_BTU_ac, average(RPM) as average_RPM_ac, average(EER) as average_EER_ac
    - for each household type, display the corresponding count_ac, average_BTU_ac, average_RPM_ac, average_EER_ac
- **Heater Summary**
    - for each household type, find all the appliances that belong to heater, calculate sum of rows as count_ht, average(BTU) as average_BTU_ht, average(RPM) as average_RPM_ht; argmax(count_values(energy source)) as common_energy_source_ht
    - for each household type, display the corresponding count_ht, average_BTU_ht, average_RPM_ht, common_energy_source_ht
- **Heat bump Summary**
    - for each household type, find all the appliances that belong to heat bump, calculate sum of rows as count_hb, average(BTU) as average_BTU_hb, average(RPM) as average_RPM_hb, average(SEER) as average_SEER_hb, average(HSPF) as average_HSPF_hb
    - for each household type, display the corresponding count_hb, average_BTU_hb, average_RPM_hb, average_SEER_hb, average_HSPF_hb

- If "***Finish***" button is pushed, perform **View Reports** task

Water heater statistics by state



Task Decomposition

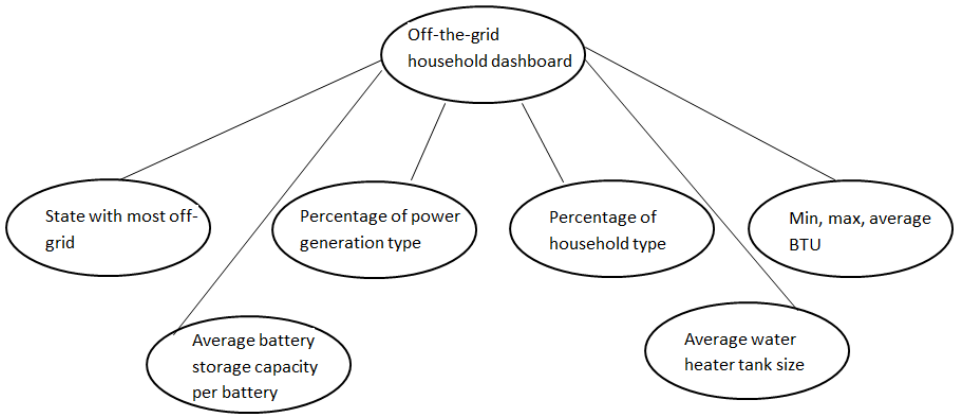
- Enabling condition: Triggered after user pressed the "***Water heater statistics by state***" button in View Reports form
- Frequency: Form frequency depends on user preference, if users view each form with the same frequency, then the water heater statistics form has medium frequency, drill down forms have low frequency; Fields in water heater statistics by state report have higher frequency than fields in drilldown report; Fields in the same report have the same frequency
- Database interaction: Read from WATERHEATER table, Read from HOUSEHOLD table, Read from LOCATION table
- Lock Types: 1 read lock for WATERHEATER table, 1 read lock for HOUSEHOLD table, 1 read lock for LOCATION table
- Sequence of subtask: Only ***List water heater statistics by state*** task is performed initially. For **Drilldown report** of any certain manufacturer (required by users, can be multiple times), it is optional and independently re-triggered by user once at a time (by clicking at the button for the specific manufacturer). Therefore, those two sub tasks are not dependent, not simultaneously triggered and the mother task is not needed to coordinate them.
- Consistency: Not critical given there is no update of data

Abstract Code

- ***List water heater statistics by state***
    - Query the WATERHEATER table, the HOUSEHOLD table and LOCATION table (with HOUSEHOLD left joined by LOCATION via common key PostalCode. Then left join this table with WATERHEATER table with household Email)
    - Display a table with the state's abbreviation, the average water heater tank size, the average water heater BTUs, the average water heater temperature setting), the count of water heaters where a temperature setting has been provided, and the count of water heaters where no temperature setting has been provided, sorted by state abbreviation ascending.
- At each row show a button ***drilldown report*** to enable the subtask **drilldown report**

- If ***drilldown report*** pushed by the user, perform **drilldown report** for the specific state in that row
- **Drilldown report**
    - Filter the main table to household located in selected state
    - Display state name at the top
    - Display a table grouped by energy source, the minimum water heater tank size, the average water heater tank size, the minimum temperature setting, the average temperature setting, the maximum temperature setting. Energy sources are ordered in ascending order.
- If "***Finish***" button is pushed, perform **View Reports** task

Off-the-grid household dashboard



Task Decomposition

- Enabling condition: Triggered after user pressed the "***Off-the-grid household dashboard***" button in View Reports form
- Frequency: Form frequency depends on user preference, if users view each form with the same frequency, then this form has medium frequency; Fields in this form have the same frequency
- Database interaction: Read from OFF-THE-GRID table, read from LOCATION table, Read from WATERHEATER table, Read from APPLIANCE table, Read from POWERGENERATOR table
- Lock Types: 1 read lock for OFF-THE-GRID table, 1 read lock for LOCATION table, 1 read lock for WATERHEATER table, 1 read lock for APPLIANCE table, 1 read lock for POWERGENERATOR table
- Sequence of subtask: All tasks must be done but can be done in parallel. Mother task is required to coordinate subtasks. Order is not necessary. Below is the list of subtasks:

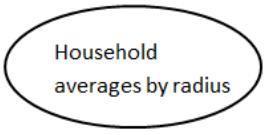
- **State with most off-grid**
- **Average battery storage capacity per battery**
- **Percentage of power generation type**
- **Percentage of household type**
- **Average water heater tank size**
- **Min, max, average BTU**

- Consistency: Not critical given there is no update of data

Abstract Code

- Query the OFF-THE-GRID table, join the table with LOCATION table via common key PostalCode
- Join table from previous step with POWERGENERATOR table via common key Email
- Join table from previous step with APPLIANCE table via common key Email
- Produce following tables：
    - **State with most off-grid**
        - Display state with the most off-the-grid households with the count (calculated using State attribute)
    - **Average battery storage capacity per battery**
        - Display average battery storage capacity per battery (calculated using BatteryStorageCapacityInKilowattHours attribute)
    - **Percentage of power generation type**
        - Display percentages for each power generation type (calculated using GeneratorType attribute)
    - **Percentage of household type**
        - Display percentages for each household type (calculated using HouseholdType attribute)
    - **Average water heater tank size**
        - Display average water heater tank size (calculated using TankSize attribute)
    - **Min, max, average BTU**
        - Display the minimum, average and maximum BTU values (calculated using BTU attribute)

Household averages by radius



Task Decomposition

- Enabling condition: Triggered after user pressed the "***Household averages by radius***" button in View Reports form
- Frequency: Form frequency depends on user preference, if users view each form with the same frequency, then this form has medium frequency; Fields in this form have the same frequency
- Database interaction: Read from HOUSEHOLD table, Read from LOCATION table
- Lock Types: 1 read lock for HOUSEHOLD table, 1 read lock for LOCATION table
- Sequence of subtask: No subtask, no decomposition is needed
- Consistency: Not critical given there is no update of data

Abstract Code

- User enters string of *postal code* and *radius* in the input field
- If postal code entered is invalid, display error message "invalid postal code"
- Query the HOUSEHOLD table and LOCATION table (with HOUSEHOLD left joined by LOCATION via common key PostalCode), calculate distance between household and postal code
- Filter to only keep household within input radius
- Display a table with the postal code, the search radius, the count of households, the count of households for each household type, the average square footage, the average heating temperature, the average cooling temperature, which public utilities are used, the count of "off-the-grid" homes, the count of homes with power generation, the most common generation method for all households with power generation, the average monthly power generation per household and the count of households with battery storage
- If "***Finish***" button is pushed, perform **View Reports** task