Table of Contents:

Data Types:

Household

Attribute	Data type	Nullable
email	String	Not Null
square_footage	Int	Not Null
thermostat_cooling	Int	Null
no_heat	Boolean	Not Null
thermostat_heating	Int	Null
no_cooling	Boolean	Not Null
type	String	Not Null

Location

Attribute	Data Type	Nullable
postal_code	String	Not Null
city	String	Not Null
state	String	Not Null
latitude	Float	Not Null
longitude	Float	Not Null

On Grid

Attribute	Data type	Nullable
public_utility_name	List<String>	Non Null

Off Grid

Attribute	Data type	Nullable

Power Generator

Attribute	Data type	Nullable
power_generator_number	Int	Not Null

avg_monthly_kwh	Int	Not Null
battery_storage_capacity	Int	Null
type	String	Not Null

Appliance Manufacturer

Attribute	Data type	Nullable
name	String	Not Null

Appliance

Attribute	Data type	Nullable
appliance_number	Int	Not Null
model_name	String	Null
btu_rating	Int	Not Null

Air Handler

Attribute	Data type	Nullable
fan_rotation	Int	Not Null

Air Conditioner

Attribute	Data type	Nullable
energy_efficiency_ratio	Float	Not Null

Heater

Attribute	Data type	Nullable
energy_source	String	Not Null

Heat Pump

Attribute	Data type	Nullable
seasonal_energy_efficiency_rating	Float	Not Null

heating_seasonal_performance_factor	Float	Not Null

Water Heater

Attribute	**Data type**	**Nullable**
tank_size	Float	Not Null
temperature_setting	Int	Null
energy_source	String	Not Null

Constraints:

- Users who are new to Alternakraft must register their household first with a unique email address.
- Users who have an existing household in Alternakraft will not be able to register.
- A user's household must be located in a postal code found in the database's postal code listing.
- The listing of postal codes will not be updated.
- Household type will be selected from a predefined list.
- Household square footage should be input as a non-negative whole number.
- Household public utilities will be selected from a predefined list.
- A household is "off-the-grid" if it has no utilities.
- Since a newly added household will not have any appliances, the user should first be shown the "add appliance" form.
- Appliance type will be selected from a predefined list.
- User should provide attributes and values that are relevant to the selected appliance type.
- Appliances will show up in the sequential order number they were entered into the system for each household.
- Each appliance will have a manufacturer that is specified in the database.
- Manufacturer list will be an updatable list retrieved from the database.
- Energy sources for heaters, heat pumps, and water heaters will be selected from a predefined list.
- Since a newly added household will not have any appliances, the user should first be shown the "add power generation" form.
- Power generator will be selected from a predefined list.
- Power generators will show up in the sequential order number they were entered into the system for each household.
- Generation type will be selected from a predefined list.
- Data should be persisted to the database when saving from the "Enter household info", "Add appliance", "Appliance listing", "Add power generation", and "Power generation listing" screens.
- If a sort order is not specified as ascending or descending, then use ascending order.
- If a number is rounded, unless otherwise specified, it should follow the "half rounds up" method.
- During report generation, the system will validate user-input parameters.
- 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 the subset, with no more than the specified number of rows returned - "tie-breaking" to determine which rows are shown is not required.
- Users will not have an option to go back and/or be able to change data they have previously entered.
- The haversine formula should be used to calculate the distances between postal codes. This calculation should happen every time it is needed instead of using a built-in or custom function. The calculation should not be performed outside of the database.

Task Decomposition with Abstract Code

Main menu

Task Decomp

Lock Types: None
Number of Locks: None
Enabling Conditions: Trigger by user loading the application, or by clicking ***Return to the main menu*** link from the **Thank You** page, or by clicking ***Back*** button on the **Reports** page.
Frequency: Around 300 times a day.
Consistency (ACID): not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Two links are shown:
    - When the ***Enter my household info*** link is clicked, go to the **Enter Household Info** page.
    - When the ***View reports/query data*** link is clicked, go to the **Reports** page.

Enter Household Info

Task Decomp

Lock Types: Write on Household, Read on Household, Read on Location
Number of Locks: Several different schema constructs are needed (at least 3).
Enabling Conditions: Trigger by user clicking ***Enter my household info*** link from the **Main menu** page.
Frequency: Around 200 new households per day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User enters *email* ('$Email'), *postal code* ('$PostalCode'), *home type* ('$HomeType'), *square footage* ('$SquareFeet'), *thermostat setting for heating* ('$ThermostatHeating') or *no heat* ('$NoHeat'), *thermostat setting for cooling* ('$ThermostatCooling') or *no cooling* ('$NoCooling'), and *public utilities* ('$PublicUtilities') input fields.
- If data validation is successful for *email, postal code, home type, square footage, thermostat setting for heating* or *no heat, thermostat setting for cooling* or *no cooling,* and *public utilities* input fields, then:
    - When ***Next*** button is clicked:
        - If '$Email' is found in Household:
            - Highlight the *email* field with an error message.
        - If '$PostalCode' does not exist in Location:
            - Highlight the *postal code* field with an error message.
        - If '$ThermostatHeating' is blank but '$NoHeat' is unchecked:
            - Highlight both *thermostat setting for heating* and *no heat* with error styling.
        - If '$ThermostatCooling' is blank but '$NoCooling' is unchecked:
            - Highlight both *thermostat setting for heating* and *no heat* with error styling.
        - Else:
            - Store all input fields (household information) as a new row in Household.
            - Go to **Add Appliance** form.

- Else *email, zip code, home type, square footage, thermostat setting for heating, no heating, thermostat setting for cooling,* and/or *no cooling* input fields are invalid, display **Enter household info** form, with the error highlighted on the input field that failed data validation and the ***Next*** button disabled until errors are cleared.

Add Appliance

Task Decomp

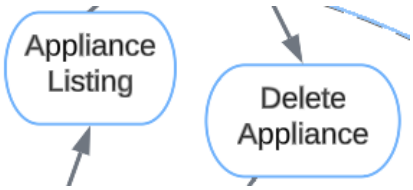
Lock Types: Read on ApplianceManufacturer, Write on Appliance
Number of Locks: Several different schema constructs are needed (at least 2).
Enabling Conditions: Trigger by successfully entering the household and clicking the ***Next*** button from the **Enter household info** page or by clicking ***Add Another Appliance*** button from the ***Appliance Listing*** page.
Frequency: High - Around 200 - 400 new appliances a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User picks the *appliance type* ('$ApplianceType') option from a dropdown input field.
- Populate the '$ApplianceManufacturer' field with allowed values from the database by reading appliance manufacturers from ApplianceManufacturer.
- Show the *BTU Rate* ('$ApplianceBTU'), *appliance manufacturer* ('$ApplianceManufacturer'), and *model name* ('$ApplianceModelName') input fields.
- Show fields specific to each appliance type:
    - If '$ApplianceType' == "Air handler", show the following input fields:
        - *Heating/cooling method* ('$ApplianceAirHandlerHeatingCoolingMethod')
            - If $ApplianceAirHandlerHeatingCoolingMethod == "Air conditioner", show the *energy efficiency ratio* ('$ApplianceAirConditionerEER') input field.
            - If $ApplianceAirHandlerHeatingCoolingMethod == "Heater", show the *energy source* ('$ApplianceHeaterEnergySource') input field.
            - If $ApplianceAirHandlerHeatingCoolingMethod == "Heat pump", show the *seasonal energy efficiency rating* ('$ApplianceHeatPumpSEER') and *heating seasonal performance factor* ('$ApplianceHeatPumpHSPF') input fields.
        - *Fan rotations per minute* ('$ApplianceAirHandlerFanRPM')
        - *Energy efficiency ratio* ('$ApplianceAirHandlerEnergyEfficiencyRatio')
    - Else, if '$ApplianceType' == "Water heater", show the following input fields:
        - *Energy source* ('$ApplianceWaterHeaterEnergySource')
        - *Tank size* ('$ApplianceWaterHeaterTankSize')
        - *BTU rating* ('$ApplianceWaterHeaterBTU')
        - *Temperature setting* ('$ApplianceWaterHeaterTemperature')
- If data validation is successful for the combination of (*appliance manufacturer, BTU, model name) AND either (Heating/cooling method, Fan rotations per minute, Energy efficiency ratio OR Energy source, Tank size, BTU rating, Temperature setting)* input fields, then:
    - When ***Add*** button is clicked:
        - Store all input fields (appliance information) as a new row in Appliance.
            - Go to the **Appliance Listing** page.
- Else the combination of *(appliance manufacturer, BTU, model name)* AND either *(Heating/cooling method, Fan rotations per minute, Energy efficiency ratio* OR *Energy source, Tank size, BTU rating, Temperature setting)* input fields are invalid, display **Add Appliance** form, with error highlighted on the input field that failed data validation.

Appliance Listing

Task Decomp



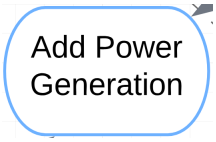
Lock Types: Read on Appliance, Write on Appliance, Read on ApplianceManufacturer
Number of Locks: Several different schema constructs are needed (at least 3).
Enabling Conditions: Trigger by clicking the ***Add*** button on the **Add Appliance** page.
Frequency: High - Around 200 - 400 - Used every time a user adds or deletes an appliance.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- We make a Read call to generate a list of *appliances* ('$Appliance') with each *appliance*'s *number* ('$ApplianceNumber'), *appliance type* ('$ApplianceType'), *appliance manufacturer* ('$ApplianceManufacturer'), *appliance model* ('$ApplianceModelName'), and a ***Delete*** button. Each ***Delete*** button makes a Write call to the database to delete an *appliance*.
- The ***Add another appliance*** button displays the **Add Appliance** form to input a new *appliance*.
- When ***Next*** button is clicked:
    - Go to the **Thank You** page.

Add Power Generation

Task Decomp



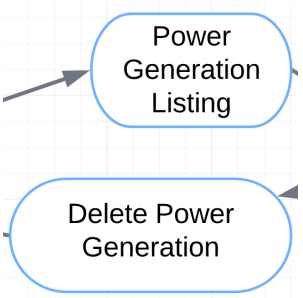
Lock Types: Write on PowerGeneration, Read on Household
Number of Locks: Several different schema constructs are needed (at least 2).
Enabling Conditions: Triggered by user clicking the ***Next*** button in **Appliance Listing** form or by clicking ***Add More Power*** button from the ***Power Generation Listing*** page.
Frequency: Low - Around 1 or 2 entries per new household.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Read Household to check if *public utilities* ('$PublicUtilities') has stored values.
    - If *public utilities* ('$PublicUtilities') has stored values, ***Skip*** button is displayed.
        - When ***Skip*** button is clicked:
            - Go to the **Thank You** page.
- User picks the *power generation type* ('$PowerGenerationType') option from a dropdown input field.
- If data validation is successful for the combination of (*power generation type, monthly kwh, storage kwh)*, then:
    - When ***Add*** button is clicked:
        - Store all input fields (power generation information) as a new row in PowerGeneration.
            - Go to the **Power Generation Listing** page.
- Else the combination of (*power generation type, monthly kwh, storage kwh)* input fields are invalid, display **Add Power Generation** form, with error highlighted on the input field that failed data validation.

Power Generation Listing

Task Decomp



Lock Types: Read on PowerGeneration, Write on PowerGeneration
Number of Locks: Several different schema constructs are needed (at least 2).
Enabling Conditions: Triggered by clicking the ***Add*** button on the **Add Power Generation** page.
Frequency: High - 100 times a day - Used every time a user views or deletes power generation.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- We make a Read call to generate a list of *power generations* ('$PowerGenerations') with each *power generation number* ('$PowerGenerationNumber'), *power generation type* ('$PowerGenerationType'), *monthly kwh* ('$MonthlykWh'), *storage kwh* ('$StoragekWh'), and a ***Delete*** button. Each ***Delete*** button makes a Write call to the database to delete an *appliance*.
- The ***Add another appliance*** button displays the **Add Appliance** form to input a new *appliance*.
- When ***Finish*** button is clicked:
    - Go to the **Thank You** page.

Thank You

Task Decomp

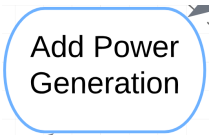
Lock Types: None
Number of Locks: None
Enabling Conditions: Triggered by user clicking the ***Skip*** button on the **Add Power Generation** page or from clicking the ***Finish*** button on the **Power Generation Listing** page.
Frequency: High - Around 200 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- One link is shown:
    - When the ***Return to the main menu*** link is clicked, go to the **Main menu** page.

Reports

Task Decomp



Lock Types: None
Number of Locks: None
Enabling Conditions: Triggered by user clicking "View reports/query data" in the main menu page.
Frequency: Medium - 180 reports a day
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- A list of report names are linked to each report.
- When the ***Top 25 popular manufacturers*** link is clicked, go to the **Top 25 popular manufacturers** page.
- When the ***Manufacturer/model search*** link is clicked, go to the **Manufacturer/model search** page.
- When the ***Heating/cooling method details*** link is clicked, go to the **Heating/cooling method details** page.
- When the ***Water heater statistics by state*** link is clicked, go to the **Water heater statistics by state** page.
- When the ***Off-the-grid household dashboard*** link is clicked, go to the **Off-the-grid household dashboard** page.
- When the ***Household averages by radius*** link is clicked, go to the **Household averages by radius** page.
- When user clicks the ***Back*** button, show the **Main menu** page.

Top 25 popular manufacturers

Task Decomp

Lock Types: Read on Appliance, Read on ApplianceManufacturer
Number of Locks: Several different schema constructs are needed (at least 2).
Enabling Conditions: Triggered by clicking the ***Top 25 popular manufacturers*** link on the **Reports** page.
Frequency: Low - around 30 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Read Appliance, Read on ApplianceManufacturer.
- The query returns a list of $ApplianceManufacturer and appliance count by $ApplianceManufacturer
    - Clicking on a link on each manufacturer opens a drill down report
        - The drill down report shows the $ApplianceType and the count of appliances by each $ApplianceType for the particular $ApplianceManufacturer.
- If the read does not return results, a message indicating “No Records Found” will be displayed.
- When user clicks the ***Back*** button, show the **Reports** page.

Manufacturer/model search

Task Decomp

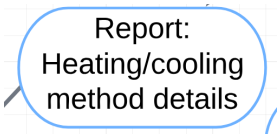
Lock Types: Read on Appliance, Read on ApplianceManufacturer.
Number of Locks: Several different schema constructs are needed (at least 2).
Enabling Conditions: Triggered by clicking the ***Manufacturer/model search*** link on the **Reports** page.
Frequency: Low - around 30 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- User enters a search *string* ('$SearchString').
- Use the *search string* ('$SearchString') to read Appliance.
- If the query returns results, there will be a list of query results ordered by *appliance manufacturer* ('$ApplianceManufacturer') ascending and *appliance model* ('$ApplianceModelName') ascending.
    - The *appliance manufacturer* ('$ApplianceManufacturer') cell and/or the *appliance model* ('$ApplianceModelName') that partially or completely matches the *search string* ('$SearchString') is highlighted with a light green background.
- If the read does not return results, a message indicating "No Records Found" will be displayed.
- When user clicks the ***Back*** button, show the **Reports** page.

Heating/cooling method details

Task Decomp



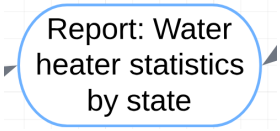
Lock Types: Read on Household, Read on Appliance
Number of Locks: Several different schema constructs are needed (at least 2).
Enabling Conditions: Triggered by clicking the ***Heating/cooling method details*** link on the **Reports** page.
Frequency: Low - around 30 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Read the Household and Appliance.
- If the query returns results, show a table grouped and ordered by *home type* ('$HomeType') that displays various statistics for '$ApplianceType' == "Air handler".
- If the read does not return results, a message indicating "No Records Found" will be displayed.
- When user clicks the ***Back*** button, show the **Reports** page.

Water heater statistics by state

Task Decomp



Lock Types: Read on Household, Read on Location, Read on Appliance
Number of Locks: Several different schema constructs are needed (at least 3).
Enabling Conditions: Triggered by clicking the ***Water heater statistics by state*** link on the **Reports** page.
Frequency: Low - around 30 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

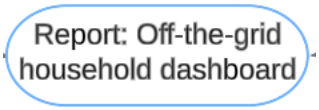
- Read Household, Location, and Appliance.
- The query returns a list of appliances with $ApplianceType' == "Water heater" and average values of *Tank size* ('$ApplianceWaterHeaterTankSize'), *BTU rating* ('$ApplianceWaterHeaterBTU'), and *Temperature setting* ('$ApplianceWaterHeaterTemperature') grouped by state.
    - If the user clicks on a link in a row, another read on Appliance is initiated and:
        - The query returns a drilldown of $ApplianceType' == "Water heater" for the selected state, listing the *Energy source* ('$ApplianceWaterHeaterEnergySource'); the

minimum, average, and maximum *Tank size* (‘$ApplianceWaterHeaterTankSize’); and the minimum, average, and maximum *Temperature setting* (‘$ApplianceWaterHeaterTemperature’).

- If the above read did not return results, display a message in place of the table, indicating “No Records Found”.
- When user clicks the ***Back*** button, show the **Reports** page.

Off-the-grid household dashboard

Task Decomp



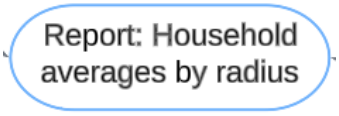
Lock Types: Read on Household, Read on PowerGeneration, Read on Appliance
Number of Locks: Several different schema constructs are needed (at least 3).
Enabling Conditions: Triggered by clicking the ***Off-the-grid household dashboard*** link on the **Reports** page.
Frequency: Low - around 30 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- There will be six tables on this page:
  - The first table requires a read on Household which returns the state which has the most households without *public utilities* ($PublicUtilities) (off-the-grid) and a count of the number of off-the-grid households in that state.
  - The second table requires a read on PowerGeneration and a read on Household which returns a list of results of all households without *public utilities* ($PublicUtilities) (off-the-grid) and the average battery *storage capacity* ($StoragekWh) per battery.
  - The third table requires a read on PowerGeneration and a read on Household, returning a list of results of the breakdown of *power generation type* ('$PowerGenerationType’) by percentage for all households without *public utilities* ($PublicUtilities) (off-the-grid).
    - If partial results are returned, use “0%” in place of missing values.
  - The fourth table requires a read on Household returning a list of results of the breakdown of *home types* (‘$HomeType’) by percentage for all households without *public utilities* ($PublicUtilities) (off-the-grid).
  - The fifth table requires a read on Appliance and a read on Household returning a list of results of the average *water heater tank size* (‘$ApplianceWaterHeaterTankSize’) for all households without *public utilities* ($PublicUtilities) (off-the-grid) and for all households with *public utilities* ($PublicUtilities) (on-the-grid).
  - The sixth table requires a read on PowerGeneration returning a list of results, grouped by *appliance type* (‘$ApplianceType’), for the minimum, maximum, and average *BTU rating* (‘$ApplianceWaterHeaterBTU’) for all households without *public utilities* ($PublicUtilities) (off-the-grid).
    - If partial results are returned, use zero in place of missing values.
- If any of the above six read locks did not return results, display a message in place of the table, indicating “No Records Found”.
- When user clicks the ***Back*** button, show the **Reports** page.

Household averages by radius

Task Decomp



Lock Types: Read on Location, Read on Household, Read on Appliance, Read on PowerGeneration
Number of Locks: Several different schema constructs are needed (at least 4).
Enabling Conditions: Triggered by clicking the ***Household averages by radius*** link on the **Reports** page.
Frequency: Low - around 30 times a day.
Consistency (ACID): Not critical, order is not critical.
Subtasks: Mother Task is not needed. No decomposition needed.

Abstract Code

- Read on Location to populate a list of allowed postal codes.
- User enters *postal code* ('$PostalCode') and *postal code search radius* ('$PostalCodeSearchRadius') input fields.
- If data validation is successful for both *postal code* and *distance of postal code* input fields, then:
    - Read on Appliance, Household, and PowerGeneration.
        - If any of the above read locks did not return results, display a message in place of the table, indicating "No Records Found".
        - Else if results exist:
            - The query returns a list of the following data:
                - $PostalCode
                - $PostalCodeSearchRadius
                - The total number of households in the $PostalCodeSearchRadius
                - For each *household type* ('$HomeType'):
                    - Count of households
                    - Average *square footage* ($SquareFeet)
                    - Average *heating temperature* ($ThermostatHeating)
                    - Average *cooling temperature* ($ThermostatCooling)
                    - List of *Public utilities* ($PublicUtilities)
                    - Count of "off-the-grid" homes
                    - Count of homes with *power generation* ($PowerGenerations)
                    - Top 1 generation method for all households with power generation ($PowerGenerations)
                    - Average monthly power generation per household ($PowerGenerations)
                    - Count of households with battery storage ($StoragekWh)
- Else *postal code* or *distance of postal* input fields are invalid, display **Household averages by radius** page, with an error message on the specific field.
- When user clicks the ***Back*** button, show the **Reports** page.