Table of Contents

Enter Household Information

Abstract Code

Show <u>Enter household info</u> form

User enters *email address* (Input = String)
User enters five-digit *postal code* (Input = Integer)
User selects *Home type* from provided drop down menu

- Drop down menu contains following options: House, apartment, townhome, condominium, or mobile home

User enters *home Square footage* (Input = Integer)
Thermostat settings
User inputs *Thermostat setting for heating* and *Thermostat setting for cooling*

- If User choose *No heat* check box, *thermostat setting for heating* should be disabled.
- If User choose *No cooling* check box *thermostat setting for cooling* should be disabled.

User selects *Public utilities* from the provided list of checkboxes with the following options: Electric, Gas, Steam, Fuel Oil, multiple options are allowed.
Validate all household info form:

- *Email address* format
- If *email address* already exists in the database

```
SELECT email
FROM household
WHERE email = '$email ';
```

- Length of *Postal Code*
- If *Postal Code* matches one that is listed in the database, run task **get postal code** from database, check if it exists in look database.

```
SELECT p_code
FROM postal_code
WHERE p_code = 'p_code';
```

- If *Square footage* > 0
- *Thermostat setting for heating* is non null, if *No heat* is false
- *Thermostat setting for cooling* is non null, if *No cooling* is false

If any of these validations fail

- Show an Error message and display User prompt to provide valid response for the name of field
- If email address exists, error prompt will display, “Email address already exists”, if postal code is not in table, “No postal code exists in system”, similarly for thermostat settings
- Show **<u>Enter household info</u>** form

User selects ***Next*** Button

Add Appliance

Abstract Code

Show **Add Appliance** form, User with no appliances added cannot skip add appliance form.

User selects *Appliance Type* from dropdown menu which shows two options: Air handler, Water heater

Look up Manufacturer and populate Manufacturer list

If User selects Air handler

- User shown information specific to Air handler
- User inputs *BTU rating* (Input = Float)
- User selects *Manufacturer* from drop down menu of options in database
    - Manufacturer is populated from the manufacturer table.

```
SELECT name
FROM manufacturer;
```

- User can input *Model name* (Input = String)
- User selects *heating/cooling method*
    - If User selects Air Conditioner
- Show *Energy Efficiency Ratio* (EER)
- User inputs *EER* (Input = Float)
    - If User selects Heater
- Show *Energy source*
- User inputs *Energy Source* from drop down menu with options: Electric, Gas, Fuel Oil
    - If User selects Heat pump
- Show *Seasonal Energy Efficiency Rating* (SEER) and *Heating Seasonal Performance Facts* (HSPF) (Input = Float)
- User inputs *SEER* and *HSPF*

User clicks ***Add*** button

If User selects Water heater

- User shown information specific to Water heater
- User selects *Manufacturer* from drop down menu of options in database
    - Manufacturer is populated from the manufacturer table.

```
SELECT name
FROM manufacturer;
```

- User can input *Model name* (Input = String)
- User selects *Energy Source* from drop down with options: Electric, gas, thermosolar, else heat pump
- User inputs *Capacity* (Input = Float)
- User inputs *BTU rating*
- User can input *Temperature* (Input = Integer)

User clicks ***Add*** button
Run the **Save appliance** task with appliance listing record for User with appliance number in the ascending order it exists in the system

- Get order_of_entry

```
SELECT MAX(order_of_entry) + 1 noe
FROM appliance
WHERE email = '$email';
```

- Insert Values into Table

```
INSERT INTO appliance (
   email,
   order_of_entry,
   btu_rating,
   manufacturer,
   model_name
)
VALUES (
  '$email',
  '$order_of_entry',
  '$btu_rating',
  '$manufacturer',
  '$model_name '
) ;
```

Show next page **Appliance Listing** page

Get Appliance Listing

Abstract Code

Show **Appliance Listing** Form

- Application Listing task

```
SELECT
appliance.email,
appliance.order_of_entry,
appliancechild.type
FROM appliance
JOIN (
        (
            SELECT
            email,
            order_of_entry,
            "air_conditioner" as type
            FROM air_conditioner )
          UNION
          (
           SELECT
           email,
           order_of_entry,
            "heater" as type
            FROM heater)
          UNION
          (
           SELECT
           email,
           order_of_entry,
          "heat_pump" as type
           FROM heat_pump )
          UNION
          (
          SELECT
          email,
          order_of_entry,
         "water_heater" as type
          FROM water_heater )
        ) AS appliancechild
ON
appliance.email = appliancechild.email
AND
appliance.order_of_entry = appliancechild.order_of_entry
WHERE appliance.email = '$email '
```

```
ORDER BY appliance.order_of_entry;
```

- Show appliance listing in order
    - When displaying the result of the query, the view logic will take the “appliancechild.type” column and convert it either Air Handler or Water Heater.
- Appliance added first starts at 1.

If User clicks ***Add another appliance*** button

- Show User **Add Appliance** Form

If User clicks ***delete*** button

- Remove Appliance entry

```
DELETE FROM appliance
WHERE
email = '$email'
AND
order_of_entry = '$order_of_entry';
```

    - Appliance child will also delete due to CASCADE ON DELETE constraint on appliance table.
- Validate that at least one appliance is added
- Validate if list appliance is zero, after deletion, show User **Add Appliance** Form, User is not allowed to go to next button with no appliance added in system.

If User clicks ***Next*** button

- Show next page, before next page run validation appliance count is more than 0, else show **Add appliance** page.

Add Power Generation

Abstract Code

Show **Add Power Generation** Form
Check Household information and derive if Household is off-the-grid or not

```
SELECT COUNT(email)
FROM household_utility
WHERE email = '$email';
```

If Household is off-the-grid

- Does not have existing power generation
- ***Skip*** button should be disabled
- ***Add*** button should be enabled
- User inputs *Type* from dropdown menu which includes: Solar-electric or wind
- User inputs *Average Monthly kWh generate*d (Input = Integer)
- User can input *Storage* in kWh (Input = Integer)

User selects ***Add*** button which stores information into database
If Household is NOT off-the-grid

- ***Skip*** button should be enabled
    - Allows User to skip adding power generation to finish submitting later
- ***Add*** button should be enabled

If User selects ***Skip***

- Show **Submission complete!** Form
    - With option to ***Return to the main menu***

If User selects ***Add*** button

- User inputs *Type* from dropdown menu which includes: Solar-electric or wind
- User inputs *Average Monthly kWh generated* (Input = Integer)
- User can input *Storage* in kWh (Input = Integer)
- Call to validate data types
- If valid data types, true, call update method
    - Update method: updates power generation record, calculates battery capacity based on kwh, if its null or empty there is no storage)
- Get Order of Entry

```
SELECT MAX(order_of_entry) + 1
FROM power_generator
WHERE email = '$email';
```

- Insert Values to the database

```
INSERT INTO power_generator (
email,
generation_type,
 avg_monthly_kwh_generated,
order_of_entry, battery_storage_capacity
)
VALUES (
'$email',
'$power_generator_type',
 '$avg_monthly_kwh_generated',
 '$order_of_entry',
'$calculated_battery_storage_capacity'
);
```

Power Generation Listing

Abstract Code

Show **Power Generation Listing** Form

```
SELECT * FROM power_generation
WHERE email = '$email'
AND order_of_entry = '$order_of_entry';
```

- *Power Generation Listing* task
- Show Power Generation listing in order of User input
- Power Generation added first starts at 1.

If User clicks ***Add more power*** button

- Show User **Add Power Generation** Form

If User clicks ***delete*** button

- Remove Power Generation entry
- Validate that at least one Power Generation is added
- Validate if list Power Generation is zero, after deletion show User **Add Power Generation** Form, User is not allowed to go to ***Finish*** button with no Power Generation added in system.

```
DELETE FROM power_generation
WHERE email = '$email'
AND order_of_entry = '$order_of_entry';
```

If count of listing is more than 0 and User is NOT off-the-grid enable ***Finish*** button

If User clicks ***Finish*** button

- Show **Submission complete!** Form
    - With option to ***Return to the main menu***

View Reports/Query Data

Abstract Code

Show **View Reports** Page
Show links provided for following reports.

- ***Top 25 popular manufacturers*** Report
- ***Manufacturer/Model search*** Report
- ***Heating cooling method details*** Report
- ***Water heater statistics by state*** Report
- ***Off-grid household*** Dashboard
- ***Household average by radius*** Report

Detailed abstract code for each report.
Each report at the end has a ***Next*** and ***Previous*** button, to take back to the report dashboard where hyperlinks for all 6 reports are displayed.
View Reports Page has link to take to Main Menu where User can either ***Enter Household Information*** or ***View Reports/Query data***

Get Top 25 Popular Manufacturers

Abstract Code

Top 25 popular manufacturers Report is selected:

- No input needed, User can only view report
- Page will be displayed with top 25 manufacturers
- Page calls get the top 25 manufactures based on total appliances for the manufacturer

```
SELECT manufacturer AS manufacturer_name, COUNT(email) AS manufacturer_count
FROM appliance
GROUP BY manufacturer_name
ORDER BY manufacturer_count DESC
LIMIT 25;
```

Results are listed based on count descending of total appliances used.

- When ***drilldown report by manufacturer*** is clicked
  - Show drilldown report by select manufacturer

```
WITH appliance_union AS (
SELECT email, 'water_heater' AS appliance_type
FROM water_heater
UNION
SELECT email, 'heater' AS appliance_type
FROM heater
```

```
UNION
SELECT email, 'heat_pump' AS appliance_type
FROM heat_pump
UNION
SELECT email, 'air_conditioner' AS appliance_type
FROM air_conditioner
)
SELECT DISTINCT au.appliance_type, COUNT(email) AS appliance_count
FROM appliance_union au
GROUP BY appliance_type
ORDER BY appliance_count DESC;
```

Get Manufacturer/Model Search

Abstract Code

Manufacturer/Model search Report is selected:

- Show search field for User to enter the query of string (String $search_criteria). (Form validation, no white space, or symbol)
- User enters search keywords, Hits ***Search*** button, input is validated for whitespaces and special characters
- Report shows ordered ascending for Manufacturer and Model name

```
SELECT manufacturer AS manufacturer_name, model_name
FROM appliance
WHERE manufacturer LIKE LOWER('%$search_criteria%')
OR model_name LIKE LOWER('%$search_criteria%')
ORDER BY manufacturer, model_name ASC;
```

- Results are pulled from the database to display the matching part for manufacturers and models. With matched string highlighted in green
- Since match is not string sensitive, hence User input is changed to lower case, back-end system in database stores in lower case string but description string which has proper camel casing. Search is performed against lowercased columns

Get Heating/Cooling Details

Abstract Code

Heating/Cooling details Report is selected:

- Display heating and cooling details
- Grouping is done based on household types.
    - Show count air conditioners, average A/C BTU, average EER in column
    - Show count heaters, average heater BTU, most common energy source in column
    - Show count of heat pumps, average heat pump BTU, average SEER, average HSPF in column
- Aggregation operation is done to collect the data from database.

```
WITH air_conditioning_stats AS (SELECT hh.email,
COUNT(ac.email) AS ac_count,
AVG(a.btu_rating) AS avg_ac_btu,
AVG(ac.energy_efficiency_ratio) AS avg_EER
FROM air_conditioner ac
INNER JOIN appliance a ON ac.email = a.email AND ac.order_of_entry = a.order_of_entry
INNER JOIN household hh ON a.email = hh.email
GROUP BY hh.email),
heater_stats AS (SELECT hh.email,
COUNT(hr.email) AS heater_count,
AVG(a.btu_rating) AS avg_heater_btu
FROM heater hr
INNER JOIN appliance a ON hr.email = a.email AND hr.order_of_entry = a.order_of_entry
INNER JOIN household hh ON a.email = hh.email
GROUP BY hh.email),
heat_pump_stats AS (SELECT hh.email,
COUNT(hp.email) AS heat_pump_count,
AVG(a.btu_rating) AS avg_heat_pump_btu,
AVG(seasonal_energy_efficiency_rating) AS avg_seer,
AVG(heating_seasonal_performance_factor) AS avg_hspf
FROM heat_pump hp
INNER JOIN appliance a ON hp.email = a.email AND hp.order_of_entry = a.order_of_entry
INNER JOIN household hh ON a.email = hh.email
GROUP BY hh.email),
most_common_energy_source AS (SELECT source,
COUNT(*) AS count_source
FROM heater
GROUP BY source
ORDER BY count_source DESC
LIMIT 1)
```

```
SELECT h_type,
SUM(ac_count) AS air_conditioning_count,
ROUND(AVG(avg_ac_btu), 0) AS avg_air_conditioner_btu,
ROUND(AVG(avg_EER),2) AS avg_eer,
SUM(heater_count) AS heater_count,
ROUND(AVG(avg_heater_btu), 0) AS avg_heater_btu,
(SELECT source FROM most_common_energy_source) AS most_common_energy_source,
SUM(heat_pump_count) AS heat_pump_count,
ROUND(AVG(avg_heat_pump_btu), 0) AS avg_heat_pump_btu,
ROUND(AVG(avg_seer), 2) AS avg_seer,
ROUND(AVG(avg_hspf), 2) AS avg_hspf
FROM household hh
LEFT JOIN air_conditioning_stats acs ON hh.email = acs.email
LEFT JOIN heater_stats hs ON hh.email = hs.email
LEFT JOIN heat_pump_stats hps ON hh.email = hps.email
GROUP BY h_type
ORDER BY h_type ASC;
```

Water Heater Statistics

Abstract Code

Water Heater Statistics Report is selected

- No input needed, User can only view report
- Page Will display water heater statistics on the first **Table**
- Declare variable called **$recordSet**

$recordSet =

```
WITH water_heaters_with_no_setting AS
( SELECT email,order_of_entry,count(email)
AS tally
FROM water_heater
WHERE current_temperature_Setting IS NULL),
water_heaters_with_setting AS
  ( SELECT email,order_of_entry,count(email) AS tally FROM water_heater
    WHERE  current_temperature_Setting
     IS NOT NULL),
households AS
  (SELECT p_code,email,count(email)
   AS houses
   FROM household)
SELECT state,
IFNULL(ROUND(AVG(wh.capacity), 0),0) AS avg_water_heater_capacity,
IFNULL(ROUND(AVG(app.btu_rating), 0),0) AS avg_water_heater_btu,
IFNULL(ROUND(AVG(wh.current_temperature_setting), 2),0) AS
water_heater_temp_setting,
IFNULL(hh1.houses,0) Households,
COUNT(wh.email) AS water_heaters,
IFNULL(water_heaters_with_no_setting.tally,0) AS water_heaters_with_no_temp_setting,
IFNULL(water_heaters_with_setting.tally,0) AS water_heaters_with_temp_setting
FROM postal_code pc
LEFT JOIN household hh ON pc.p_code = hh.p_code
LEFT JOIN households hh1 ON pc.p_code = hh1.p_code
LEFT JOIN water_heater wh ON wh.email = hh.email
LEFT JOIN appliance app ON wh.email = app.email
AND app.order_of_entry = wh.order_of_entry
LEFT JOIN water_heaters_with_no_setting ON
water_heaters_with_no_setting.email=hh.email
AND water_heaters_with_no_setting.order_of_entry=app.order_of_entry
LEFT JOIN water_heaters_with_setting ON water_heaters_with_setting.email=hh.email
AND water_heaters_with_setting.order_of_entry=app.order_of_entry
GROUP BY state, Households
ORDER BY state ASC;
```

For each record in $recordSet extract and add as row to first table
state,average_water_heater_capacity,average_water_heater_btu,average_water_temperature_setting,water_heaters_with_temperature_setting,water_heaters_without_temperature_setting

IF User selects a row in the first **Report** then

- Retrieve the corresponding state selected

If recordset.energy_source is 'Electric'display on 'Electric'column
If recordset.energy_source is 'Gas'display on 'Gas'column
If recordset.energy_source is 'thermosolar'display on 'thermosolar'column
Else display energy_source on 'Heat pump'Column.

$state=state Selected
Display Drill Down **Report**

$drillDown =

```
SELECT    energy_source,
ROUND(MIN(capacity)) AS Min_capacity,
ROUND(AVG(capacity)) AS Avg_Capacity,
ROUND(MAX(capacity)) AS Max_capacity,
MIN(current_temperature_setting) AS Min_temp_setting,
ROUND(MAX(current_temperature_setting), 2) AS Max_temp_setting
FROM postal_code pc
INNER JOIN
household hh
ON hh.p_code = pc.p_code
JOIN appliance app ON app.email = hh.email
JOIN water_heater wh ON wh.email = app.email
AND wh.order_of_entry = app.order_of_entry
WHERE state = '$state'
GROUP BY energy_source
ORDER BY energy_source ASC;
```

For every record in $drillDown extract and add row to drill down table
Energy_source,
Min_capacity,Avg_capacity,Maximum_capacity,Min_temp_setting,Minimum_temp_setting,
Average_temperature_setting, Maximum_temp_setting

Off-the-Grid household Dashboards

Abstract Code

- *Off the grid House hold* Report is selected
- No input Needed, User can only view the report
- Display state with the most off-the-grid-household

$recordSet =

```
SELECT pc.state,count(hh.email)
AS num_house_holds
FROM postal_code pc
JOIN household hh
ON pc.p_code=hh.p_code
LEFT JOIN household_utility hh_ut
ON hh_ut.email = hh.email
WHERE hh_ut.email IS NULL
GROUP BY state
ORDER BY count(hh_ut.email)  DESC
LIMIT 1;
```

IF $recordSet is not empty THEN

Display state, number_of_households on the first Report Page

$offTheGrid =

```
SELECT
AVG(battery_storage_capacity) storage_capacity
FROM postal_code pc
INNER JOIN household hh
ON hh.p_code=pc.p_code
INNER JOIN power_generator pg
ON pg.email = hh.email
INNER JOIN
(SELECT hh.email
  FROM postal_code pc
  JOIN household hh
  ON pc.p_code=hh.p_code
  LEFT JOIN household_utility hh_ut
  ON hh_ut.email = hh.email
  WHERE hh_ut.email IS NULL
  GROUP BY email)
AS off_the_grid
ON hh.email=off_the_grid.email
GROUP BY state;
```

Display battery storage capacity

- Get data for all off-the-grid households, the percentages (as decimal numbers, rounded to tenths) for each power generation type (solar-electric, wind, or mixed) and display

$recordSet =

```
WITH capacity AS
  (SELECT sum(battery_storage_capacity) AS total_capacity
   FROM power_generator)
SELECT ROUND((battery_storage_capacity/(SELECT total_capacity FROM capacity)*100),2) AS
percentage,generation_type,pc.state
FROM postal_code pc
INNER JOIN household hh
ON hh.p_code=pc.p_code
INNER JOIN power_generator pg
ON pg.email = hh.email
INNER JOIN
(SELECT hh.email
 FROM household hh
 JOIN postal_code pc
 ON pc.p_code=hh.p_code
 LEFT JOIN household_utility hh_ut
 ON hh_ut.email = hh.email
 WHERE hh_ut.email IS NULL
 GROUP BY email) AS off_the_grid
ON hh.email=off_the_grid.email
GROUP BY state,generation_type;
```

For record in $recordSet
  Display $recordSet.state, $recordSet.generationType,$recordSet.percentage

− Get average water heater gallon capacity for off-grid house, and on-grid houses for all state and display it to a table

```
WITH off_grid
AS (SELECT ROUND(AVG(wh.capacity), 2) AS gallon_capacity_off_grid,pc.state
      FROM   postal_code pc
      INNER JOIN household hh
      ON hh.p_code = pc.p_code
      INNER JOIN water_heater wh
      ON wh.email = hh.email
       INNER JOIN
          (SELECT hh.email
            FROM   postal_code pc
            INNER JOIN household hh
            ON pc.p_code = hh.p_code
            LEFT JOIN household_utility hh_ut
            ON hh_ut.email = hh.email
            WHERE  hh_ut.email IS NULL
            GROUP  BY email) AS off_the_grid
       ON hh.email = off_the_grid.email
       GROUP  BY state),
    on_grid
    AS (SELECT Round(AVG(wh.capacity), 2) AS gallon_capacity_on_the_grid,
          pc.state  AS state2
          FROM   postal_code pc
          INNER JOIN household hh
          ON hh.p_code = pc.p_code
          INNER JOIN water_heater wh
          ON wh.email = hh.email
          INNER JOIN household_utility hh_util
          ON hh_util.email = wh.email
           GROUP  BY state)
  SELECT off_grid.gallon_capacity_off_grid,
           on_grid.gallon_capacity_on_the_grid
FROM   off_grid
      INNER JOIN on_grid
       ON on_grid.state2 = off_grid.state;
```

− Get the minimum, average and maximum (as whole numbers, rounded) BTUs for all off-the-grid households'appliances and display along with the appliance type

```
SELECT
   ROUND(MIN(btu_rating)) as Min_btu,
   ROUND(AVG(btu_rating)) as Average_btu,
   ROUND(MAX(btu_rating)) as Max_btu,
   appliance_type
FROM appliance app
INNER JOIN
   (SELECT
     email, order_of_entry, 'Water_heater' AS appliance_type
     FROM
     water_heater UNION SELECT
     email, order_of_entry, 'heater' AS appliance_type
     FROM
     heater
     UNION
     SELECT email, order_of_entry, 'heat_pump' AS appliance_type
     FROM heat_pump
     UNION
     SELECT email, order_of_entry, 'air conditioner' AS appliance_type
     FROM air_conditioner) app_type ON app_type.email = app.email
     AND app.order_of_entry = app_type.order_of_entry
      INNER JOIN
        (SELECT hh.email
          FROM postal_code pc
          JOIN household hh ON pc.p_code = hh.p_code
          LEFT JOIN household_utility hh_ut ON hh_ut.email = hh.email
          WHERE
          hh_ut.email IS NULL
          GROUP BY email) off_grid ON off_grid.email = app.email
GROUP BY appliance_type;
```

HouseHold Averages by radius

Abstract Code

Household averages by Radius Report is selected

- This report takes two inputs
- User inputs the postal code (input = Integers)
- User Selects Search Distance from *List*
- User Clicks on the *display Button.*

System validates postal code inputted is valid

$postalCode =

```
SELECT p_code from postal_code where p_code = '$postalCode';
```

If $postalCode is empty

Display message to user that post $postalCode does not exist.

Else

Query Database to fetch the report

- $searchRadius = Search Distance From List

$recordSet =

```
SELECT
    u.p_code AS postal_code,
   '$searchRadius ' AS search_radius,
    COUNT(hh.email) AS households_within_radius,
    hh.h_type as household_type,
    COUNT(hh.h_type) AS count_of_household_types,
    IFNULL(ROUND(AVG(hh.square_footage), 0), 0) AS Avg_square_footage,
    IFNULL(ROUND(AVG(hh.heat_setting), 2), 0) AS Avg_heating_temperature,
    IFNULL(ROUND(AVG(hh.cool_setting), 2), 0) AS Avg_Cooling_temperature,
    GROUP_CONCAT(hh_util.utility) AS public_utilities,
    IFNULL(off_the_grid.off_the_grid_homes, 0) AS off_the_grid_homes,
    COUNT(pg.email) AS houses_with_power_generators,
    generation_type.generation_type AS Common_generation_type,
    IFNULL(ROUND(AVG(pg.avg_monthly_kwh_generated), 2),
        0) AS Avg_Monthly_Power_Generation,
    IFNULL(COUNT(pg_battery_storage.battery_storage_capacity),
        0) AS houses_with_battery_storage
FROM
    (SELECT p1.p_code,
RADIANS(p1.latitude) AS lat1,
        RADIANS(center.latitude) AS lat2,
        RADIANS(p1.longitude) AS lon1,
        RADIANS(center.longitude) AS lon2,
        (SELECT lat2) - (SELECT lat1) AS deltaLat,
```

```
        (SELECT lon2) - (SELECT lon1) AS deltaLon,
        POWER(SIN((SELECT deltaLat / 2)), 2) + COS((SELECT lat1)) * COS((SELECT lat2)) *
         POWER(SIN((SELECT deltaLon / 2)), 2) AS a,
        2 * ATAN2(SQRT((SELECT a)), SQRT(1 - (SELECT a))) AS c,
        (SELECT c) * 3958.75 AS distance_in_mil
   FROM
      postal_code p1
   JOIN
(SELECT latitude, longitude
   FROM
      postal_code
   WHERE
      p_code = '$postalCode') AS center) u
      LEFT JOIN household hh ON hh.p_code = u.p_code
      LEFT JOIN household_utility hh_util ON hh_util.email = hh.email
      LEFT JOIN power_generator pg ON pg.email = hh.email
      LEFT JOIN
   (SELECT
      pg.email, generation_type, COUNT(generation_type)
   FROM power_generator pg
   INNER JOIN household hh2 ON pg.email = hh2.email
   GROUP BY generation_type
   ORDER BY COUNT(generation_type) DESC
   LIMIT 1) AS generation_type ON generation_type.email = hh.email
     LEFT JOIN
   (SELECT hh.email, COUNT(hh.email) off_the_grid_homes
   FROM postal_code pc
   INNER JOIN household hh ON pc.p_code = hh.p_code
   LEFT JOIN household_utility hh_ut ON hh_ut.email = hh.email
   WHERE hh_ut.email IS NULL
   GROUP BY hh.email) off_the_grid ON off_the_grid.email = hh.email
      LEFT JOIN
   power_generator pg_battery_storage ON hh.email = pg_battery_storage.email
WHERE
   distance_in_mil <= '$searchRadius'
GROUP BY u.p_code , hh.h_type;
```

Display the report in tabular format
For each record in $recordSet
Display

- Postal Code
- Search Radius
- Households within Radius
- HouseHold Type
- Count of HouseHold Types
- Average Square footage
- Average Heating Temperature
- Average Cooling Temperature
- Public Utilities
- Count of Houses with power generators
- Count of Off-the-grid homes
- Count of Houses with Power Generators
- Most Common Generation Type
- Average Monthly Power Generation
- Number of households with battery storage