**​​**

Table of Contents

[Enter Household Information - 2 -](#_Toc130088297)

[Add Appliance - 3 -](#_Toc130088298)

[Get Appliance Listing - 5 -](#_Toc130088299)

[Add Power Generation - 7 -](#_Toc130088300)

[Power Generation Listing - 9 -](#_Toc130088301)

[View Reports/Query Data - 10 -](#_Toc130088302)

[Get Manufacturer/Model Search - 12 -](#_Toc130088303)

[Get Heating/Cooling Details - 13 -](#_Toc130088304)

[Water Heater Statistics - 15 -](#_Toc130088305)

[Off-the-Grid household Dashboards - 17 -](#_Toc130088306)

[HouseHold Averages by radius - 21 -](#_Toc130088307)

**​**

**​**

**​**

**​**

***​***

***​***

**​**

**​**

**​​**

# Enter Household Information

**Abstract Code**

**Show Enter household info 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 addres*s format
* If *email address* already exists in the database

|  |
| --- |
| SELECT email  FROM household  WHERE email = '$email '; |

* Length of *Postal Code*
* If *Postal Cod*e 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 **Enter household info** 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 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),    heater\_stats AS (SELECT COUNT(hr.email) AS heater\_count,  AVG(a.btu\_rating) AS avg\_heater\_btu,  (SELECT source, COUNT(\*) AS source\_count FROM heater GROUP BY source ORDER BY source\_count DESC LIMIT 1) AS most\_common\_energy\_source  FROM heater hr  INNER JOIN appliance a ON hr.email = a.email AND hr.order\_of\_entry = a.order\_of\_entry),    heat\_pump\_stats AS (SELECT 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)    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),  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  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