In this project, you will undertake a data engineering task on a real dataset of meter data from the University of California, Berkeley. The dataset contains data from a collection of buildings, each with various meters measuring different quantities (steam, chilled water, power, energy, etc). This dataset is stored in the `berkeley_meters` schema, which contains the following tables:

- `meters` - contains information about each meter, including the meter id, the site it is located at, the label of the meter, and the unit of measurement.
- `data` - contains the actual data points for each meter, including the meter id, the timestamp of the data point, and the value of the data point.
- `real_estate` - contains information about each site, including the site name, the net square footage of the site, and the year the site was built.

Set your search path so your schema comes first, followed by `berkeley_meters`. You can do this by running the following command in `psql`:

```
SET search_path TO your_schema, berkeley_meters;
```

Your task is to write a series of SQL queries to clean and analyze this dataset. You should write your queries in the `project.sql` file. You should also write a `README.txt` file that contains the answers to the questions that require a written response. You should submit both the `README.txt` and `project.sql` files in a zip archive to Canvas. This project is worth 50 points.

The `project.sql` file has the following structure:

```
CREATE VIEW q1 AS (
    -- YOUR CODE HERE
);
CALL check_view('q1', ARRAY['id', 'label', 'site', 'units'], 23);
```

Do not change the `CREATE VIEW qX AS (` line or the `CALL check_view` line. Your queries will go into the view definition; replace the placeholder query (`SELECT 'YOUR CODE HERE'`) with your answer. The `CALL check_view` function will check that your query is correct by comparing the columns in your view to the expected columns. The second argument to `CALL check_view` is an array of the expected column names. The third argument is the number of rows in the view. You should run the `project.sql` file in your database to check your answers using the `\i project.sql` command in `psql`.

`check_view` is a custom function that we have provided for you. You do not need to understand how it works, but you should not modify it. If your answer has the correct columns and the correct number of rows, the function will return without error and not print anything. If your answer is incorrect, the function will print an error message. When you are using the `\i project.sql` command in `psql` to check your answers, you will see error messages for all views you have not yet implemented. You can ignore these error messages until you are ready to implement the next view. *We recommend that you make sure each view works correctly before moving on to the next one.*

Q0 (3 points) Submit both the `README.txt` and `project.sql` files in a zip archive to Canvas.

Q1 (4 points) Create a view `Q1` that contains only the *main power meters* for each building. A meter is a main power meter if it has the words "demand" and "main" in its label (not necessarily in that order), and has kilowatts as its unit. Make sure your query is case-insensitive.

Q2 (3 points) Notice the duplicate building names which are similar (e.g. "Soda Hall New (as of 1/12/12)" and "Soda Hall"). Create a view `Q2` that cleans up the site names by removing the text inside parentheses. For example, "Soda Hall New (as of 1/12/12)" should be cleaned up to "Soda Hall". Your `Q2` view should query from the `Q1` view you created in the previous question.

(Hint: no need to make this fully generic! You can hard-code the cleaning for the given examples.)

(Hint: you can use methods like `regexp_replace` or hard-coded `CASE` statements to clean up the site names.)

Q3 (2 points) Create a new view `Q3` that contains the set of unique buildings in our newly cleaned-up dataset. From now on, when we refer to `site`, we mean the cleaned-up site name. (`clean_site`). Your `Q3` view should query from the `Q2` view.

Now that we have the cleaned site names, we can figure out which sites have data quality issues. The `data` table contains timeseries data for each meter. The rows in this table were created by reading each meter's value every 15 minutes. However, meters can sometimes fail to record data or break down, leading to gaps in the timeseries. The next few questions will focus on identifying missing timeseries data. We want to remove all sites and meters with missing data from our analysis. Additionally, some sites have multiple meters and we will need to combine the data from these meters into a single timeseries.

Q4 (4 points) Let's start by getting an idea of how much data we have for each site. Create a new view `Q4` that lists:

- the (cleaned) site name (as `clean_site`),
- the number of main power meters at the site (as `num_meters`),
- the total number of data points for all main power meters at the site (as `num_data_points`).

Now, we want to remove sites that (a) have no data at all, or (b) have gaps in their data. We will identify these sites in the next few questions.

Q5 (2 points) **Sites with no data:** At this point, take a look at the number of rows in `Q3` and `Q4` — they are different! This happens because 1 site does not have any entries in the `data` table. Create a new view `Q5` to return the name of the site that does not have data. You should use the `Q3` and `Q4` views for this question.

Q6 (6 points) **Sites with data gaps:** Each meter should have a data point every 15-16 minutes. Create a new view `Q6` to find the id of the main power meters that are missing data. We consider a meter to be missing data if the gap between consecutive data points is greater than 16 minutes. The query should return the site, the meter id, and the size of the gap (in a column called `time_diff`). Use your `Q2` view and the `data` table for this question.

(Hint: use the `INTERVAL` type in Postgres to represent time differences. Docs are here.)

At this point, views `Q5` and `Q6` contain sites which we want to *exclude* from future processing. Make sure you exclude these sites in the following questions.

Q7 (5 points) Our result from `Q4` tells us some sites have more than one meter, which will complicate our data analysis in future questions. To make the analysis easier, we want to aggregate all the meter data for each site into a single timeseries. Create a new view `Q7` which combines all meter data for each site into a single timeseries by taking the sum of all values *at each timestamp* for a site. You will need to aggregate values by site and timestamp. The view should contain the following columns: `site`, `time`, `value`. Your `Q7` view should only contain sites with complete data (i.e. exclude the sites you found in `Q6`). Do not include sites which have missing data; use a subquery to check if the site is in the result of `Q6` as part of your solution.

Q8 (5 points) Five of our sites have serious data quality issues with the `value` field. Create a new view `Q8` that contains the names of sites with these data quality issues. These issues should be obvious when you look at the data. Your query should compute the answer by doing some computation on the data, not by hard-coding the list of sites.

(Hint: you can use the `COUNT` function to count the number of non-zero values for each site. Most sites should have about the same number of non-zero values. If a site has a significantly different number of non-zero values, it has a data quality issue.)

Q9 (3 points) Explain the data quality issues you found in `Q8` and how your query identified them. Explain any "magic numbers" in your query, such as thresholds and limits and constants. Write your answer in the README.txt file. Include an explanation of why these data quality issues were not caught by the "data gap" check in `Q6`.

Begin the first line of your answer with the string "(Q9)", e.g. *(Q9) The data quality issue I found was...*

Q10 (5 points) Calculate the energy usage intensity (EUI) for each building using the cominbed data from `Q7`. EUI is the total *energy* usage for a building over a week divided by its net square footage. Calculate energy consumption from power data by adding up all power consumption for each site and dividing it by 4 (this will convert 4 15-minute interval kilowatt power readings to kilowatt-hours). Then divide by the net square footage of the building. The resulting value has units of kWh/sqft. Put your query in a view called `Q10`. Only compute the answer for our clean sites with complete data (i.e. exclude the sites you found in `Q8`).

(Note: We only have 1 week of data in the `data` table, so this is technically "Weekly EUI". In practice, EUI is calculated over a year.)

Q11 (5 points) We would like to see if there is a relationship between the age of a building and its energy usage. We *hypothesize* that older buildings are less energy efficient (i.e. there should be a *positive* correlation between the age of a building and its EUI).

Write a new view `Q11` that computes a simple linear regression model that predicts the EUI of a building based on the *age* of the building in years. Your view should return the slope, intercept, and correlation coefficient of the regression line. Only compute the answer for our clean sites with complete data (i.e. exclude the sites you found in `Q8`).

(Hint: use Postgres's built-in linear regression functions. See the documentation here.)

Q12 (3 points) Does our hypothesis that older buildings are less energy efficient hold? In a real data science position we would want more rigorous statistical tests, but for now, we will just look at the correlation coefficient. Provide an answer ("yes" or "no") in the README.txt file and justify your answer using the correlation coefficient values from `Q11`. How confident can we be in the model?

(Hint: the correlation coefficient is a value between -1 and 1 and tells us the strength and direction of the relationship between two variables. A positive value indicates a positive relationship, while a negative value indicates a negative relationship. The closer the value is to 1 or -1, the stronger the relationship. A value of 0 indicates no relationship. The slope and intercept allow us to predict the EUI of a building based on its age. The correlation coefficient tells us how well the model fits the data.)

Begin the first line of your answer with the string "(Q12)", e.g. *(Q12) The hypothesis that older buildings are less energy efficient...*