Congratulations! You've decided to treat yourself to a long holiday vacation in Honolulu, Hawaii. To help with your trip planning, you decided to do a climate analysis about the area.

**Part 1: Analyze and Explore the Climate Data**

In this section, you’ll use Python and SQLAlchemy to do a basic climate analysis and data exploration of your climate database. Specifically, you’ll use SQLAlchemy ORM queries, Pandas, and Matplotlib. To do so, complete the following steps:

1. Note that you’ll use the provided files (climate\_starter.ipynb and hawaii.sqlite) to complete your climate analysis and data exploration.
2. Use the SQLAlchemy create\_engine() function to connect to your SQLite database.
3. Use the SQLAlchemy automap\_base() function to reflect your tables into classes, and then save references to the classes named station and measurement.
4. Link Python to the database by creating a SQLAlchemy session.
5. Perform a precipitation analysis and then a station analysis by completing the steps in the following two subsections.

**Precipitation Analysis**

1. Find the most recent date in the dataset.
2. Using that date, get the previous 12 months of precipitation data by querying the previous 12 months of data.

**hint**

Don’t pass the date as a variable to your query.

1. Select only the "date" and "prcp" values.
2. Load the query results into a Pandas DataFrame. Explicitly set the column names.
3. Sort the DataFrame values by "date".
4. Plot the results by using the DataFrame plot method, as the following image shows:

The AutomapBase.prepare.reflect parameter is deprecated and will be removed in a future release. Reflection is enabled when AutomapBase.prepare.autoload\_with is passed.

**Station Analysis**

1. Design a query to calculate the total number of stations in the dataset.
2. Design a query to find the most-active stations (that is, the stations that have the most rows). To do so, complete the following steps:
   * List the stations and observation counts in descending order.
   * Answer the following question: which station id has the greatest number of observations?
3. Design a query that calculates the lowest, highest, and average temperatures that filters on the most-active station id found in the previous query.
4. Design a query to get the previous 12 months of temperature observation (TOBS) data. To do so, complete the following steps:
   * Filter by the station that has the greatest number of observations.
   * Query the previous 12 months of TOBS data for that station.
   * Plot the results as a histogram with bins=12, as the following image shows:
5. Close your session.

**Part 2: Design Your Climate App**

Now that you’ve completed your initial analysis, you’ll design a Flask API based on the queries that you just developed. To do so, use Flask to create your routes as follows:

1. /
   * Start at the homepage.
   * List all the available routes.
2. /api/v1.0/precipitation
   * Convert the query results from your precipitation analysis (i.e. retrieve only the last 12 months of data) to a dictionary using date as the key and prcp as the value.
   * Return the JSON representation of your dictionary.
3. /api/v1.0/stations
   * Return a JSON list of stations from the dataset.
4. /api/v1.0/tobs
   * Query the dates and temperature observations of the most-active station for the previous year of data.
   * Return a JSON list of temperature observations for the previous year.
5. /api/v1.0/<start> and /api/v1.0/<start>/<end>
   * Return a JSON list of the minimum temperature, the average temperature, and the maximum temperature for a specified start or start-end range.
   * For a specified start, calculate TMIN, TAVG, and TMAX for all the dates greater than or equal to the start date.
   * For a specified start date and end date, calculate TMIN, TAVG, and TMAX for the dates from the start date to the end date, inclusive.

**Hints**

* Join the station and measurement tables for some of the queries.
* Use the Flask jsonify function to convert your API data to a valid JSON response object.