9.5.6 Statistics Route

The investors will need to see the minimum, maximum, and average temperatures. For this we'll create a route for our summary statistics report.

Just one more route to create! Our last route will be to report on the minimum, average, and maximum temperatures. However, this route is different from the previous ones in that we will have to provide both a starting and ending date. Add the following code to create the routes:

```
@app.route("/api/v1.0/temp/<start>")
@app.route("/api/v1.0/temp/<start>/<end>")
```

Next, create a function called `stats()` to put our code in.

```
def stats():
    return
```

We need to add parameters to our `stats()` function: a `start` parameter and an `end` parameter. For now, set them both to `None`.

```
def stats(start=None, end=None):
     return
```

With the function declared, we can now create a query to select the minimum, average, and maximum temperatures from our SQLite database. We'll start by just creating a list called `sel`, with the following code:

```
def stats(start=None, end=None):
    sel = [func.min(Measurement.tobs), func.avg(Measurement.tobs), func.max(
```

Since we need to determine the starting and ending date, add an `if-not` statement to our code. This will help us accomplish a few things. We'll need to query our database using the list that we just made. Then, we'll unravel the results into a one-dimensional array and convert them to a list. Finally, we will jsonify our results and return them.

NOTE

> In the following code, take note of the asterisk in the query next to the `sel` list. Here the asterisk is used to indicate there will be multiple results for our query: minimum, average, and maximum temperatures.

```
def stats(start=None, end=None):
    sel = [func.min(Measurement.tobs), func.avg(Measurement.tobs), func.max(

    if not end:
        results = session.query(*sel).\
            filter(Measurement.date >= start).all()
```

```
        temps = list(np.ravel(results))
        return jsonify(temps=temps)
```

Now we need to calculate the temperature minimum, average, and maximum with the start and end dates. We'll use the `sel` list, which is simply the data points we need to collect. Let's create our next query, which will get our statistics data.

```
def stats(start=None, end=None):
    sel = [func.min(Measurement.tobs), func.avg(Measurement.tobs), func.max(

    if not end:
        results = session.query(*sel).\
            filter(Measurement.date >= start).all()
        temps = list(np.ravel(results))
        return jsonify(temps)

    results = session.query(*sel).\
        filter(Measurement.date >= start).\
        filter(Measurement.date <= end).all()
    temps = list(np.ravel(results))
    return jsonify(temps)
```

Finally, we need to run our code. To do this, navigate to the "surfs_up" folder in the command line, and then enter the following command to run your code:

```
flask run
```

After running this code, you'll be able to copy and paste the web address provided by Flask into a web browser. Open `/api/v1.0/temp/start/end route` and check to make sure you get the correct result, which is:

```
[null,null,null]
```

This code tells us that we have not specified a start and end date for our range. Fix this by entering any date in the dataset as a start and end date. The code will output the minimum, maximum, and average temperatures. For example, let's say we want to find the minimum, maximum, and average temperatures for June 2017. You would add the following path to the address in your web browser:

```
/api/v1.0/temp/2017-06-01/2017-06-30
```

When you run the code, it should return the following result:

```
["temps":[71.0,77.21989528795811,83.0]]
```

You've just completed your second Flask application. Great work! Flask and Python are incredibly useful tools to have in your toolbelt. But this is challenging stuff, so give yourself a pat on the back for getting through it—and successfully!

ADD/COMMIT/PUSH

The Flask app is complete, which is a good time to add, commit, and push our updates to GitHub. Remember these steps:

1. `git add < FILE NAME>`
2. `git commit -m "< ADD COMMIT MESSAGE HERE>"`
3. `git push origin main`