def precipitation(start\_date, end\_date):

#Docstring for function 'calc\_temps'

Args:

start\_date(string): A date string in the format %Y-%m-%d

end\_date (string): Adate strig in the format %Y-%m-%d

Returns:

rain = [Measurement.station,

Station.name

Station.latitude,

Station.longitutde,

Station.elevation,

Measurment.prcp]

return session.query(\*sel).\

filter(Measurement.station == Station.station).filter(Measurement.date>=start\_date).filter(Measurement.date <= end\_date).group\_by(Measurement.station).order\_by(Measurement.prcp.desc().all()

def daily\_normals(date):

"""Daily Normals.

Args:

date (str): A date string in the format '%m-%d'

Returns:

A list of tuples containing the daily normals, tmin, tavg, and tmax

"""

sel = [func.min(Measurement.tobs), func.avg(Measurement.tobs), func.max(Measurement.tobs)]

return session.query(\*sel).filter(func.strftime("%m-%d", Measurement.date) == date).all()

daily\_normals("01-01")

@app.route("/api/v1.0/precipitation/<date\_input>")

def precip(date\_input):

precip\_dict={}

results=session.query(Measurement.prcp).filter(func.strftime("%Y-%m-%d", date\_input)==Measurement.date).all()

results\_list= list(np.ravel(results))

precip\_dict["date"] = date\_input

precip\_dict["prcp"] = results\_list

return jsonify(precip\_dict)

precip\_data=[]

for prcp\_data in precip\_year:

prcp\_dict ={}

date\_dict={}

prcp\_dict["Precipitation"]=prcp\_data.prcp

date\_dict["Date"]=prcp\_data.date

precip\_data.append(prcp\_dict, date\_dict)

return jsonify(precip\_data)

precip\_data=[]

for prcp\_data in precip\_year:

prcp\_dict={}

prcp\_dict["date"]= prcp\_data.date

prcp\_dict["prcp"]= prcp\_data.prcp

precip\_data.append(prcp\_dict)

return jsonify(precip\_data)

def stations\_1():

station\_data=session.query(Station.station).group\_by(Station.station).all()

stations\_list=list(np.ravel(station\_data))

return jsonify(stations\_list)

station\_data=session.query(Measurement.station func.count(Measurement.station)).group\_by(Measurement.station).order\_by(func.count(Measurement.station).desc()).all()

data\_list\_ave=[]

for t in data\_avg:

avg\_dict={}

avg\_dict["Date Year Ago"]=date\_year\_ago

avg\_dict["Ending Date"]=end\_date

avg\_dict["Ave Temp."] = float(t[1])

avg\_dict["Top Temp."] = float(t[2])

avg\_dict["Top Temp."] = float(t[3])

data\_list\_ave.append(avg\_dict)