This document explains how to get iot data in real time from the cloud in the program realtimedata.py

* The resource that we want to work with is specified as a resource.
* The table name IMU\_Data is specified with the Table parameter to write data into it.
* The table.scan command scans the entire table and gets the data of all the attributes.

Documentation for scan : <https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/dynamodb.html#DynamoDB.Table.scan>

Api Reference:

<https://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_Scan.html>

Stack Overflow:

<https://stackoverflow.com/questions/36780856/complete-scan-of-dynamodb-with-boto3>

* The set of variables that will hold the data to be plotted are initialised with an empty list, so that values can be appended to them.
* The scan command returns a dictionary with a lot of values(mentioned in docs above). The data from the table is under the key ‘Items’ inside the dictionary. We store that in data.
* In the try block, we get the last set of data that we just retrieved. If there is no data, then there will be an index error when we try to do this which we catch with the except and exit the program.
* From the last set of data, we get the primary keys of the table which are ‘device\_id’, and ‘timestamp’ so that we can pass it in the next iteration.
* Then if there is initial data, we enter into the loop ,get the data and store them in individual variables using list comprehension.
* Here, timestamp alone uses the extend function because new timestamps from the new data will be appended to the existing list of timestamps from the old data.
* The timestamp is split and only the time part is taken using the split() which will return a list with the date at index 0 and time at index 1. So, we specify [1] to get the time. Then in the time, we don’t want to display the milliseconds as it will be too much clutter. So, [:-4], will make us get until the “end-4” index of the time, which ignores the milliseconds part.
* Each data list is sent to split\_axes function to split into x,y, and z axis data to make it easier to plot along with the individual axis variable into which it should be stored.
* All the coordinate data is in the form a string in this form [‘[0, 0, 0]’,’[1, 2, 3]’, …..]. So, we take each set from the list and then strip the brackets( ][ ). The, there is a space before the numbers, so we remove that with replace and then split them by the commas and store them in a list. So, [‘[0, 0, 0]’,’[1, 2, 3]’, …..]. will become [[‘0’,’0’,’0’], [‘1’,’2’,’3’],……]
* Now the string numbers should be converted to float in the next list comprehension.
* All the numbers in the 0th position(i.e, the x-axis) are stored in x and so on.
* These are stored in mag\_x, mag\_y etc, according to the variables which were passed.
* The extend() function joins two lists together. This is used because when the next set of data is received, that data should be appended to the already existing data.
* Coming back to the loop, the sleep is used because data is being sent at x time interval, so we have to wait x secs before the next read, otherwise we’ll get empty data and the program will terminate.
* So, after waiting x seconds, we read again but this time ExclusiveStartKey = lastevalkey is specified. What this does is, it reads data starting tight after that particular key. So, we don’t have read the old data again and only read the new data that has entered the table.
* If no new data has entered the table, then in the try block, we’ll get an IndexError when we try to get the last item. So, we catch that exception and break out of the loop.
* Otherwise, the lastevalkey is recalculated and the loop proceeds.