CA 3

Lessons Learned

Für:

Big Data Engineering & Analysis (SS22)

Prof. Dr. Oliver Hummel

Bei:

Milad Afshar Jahanshahi

2150426

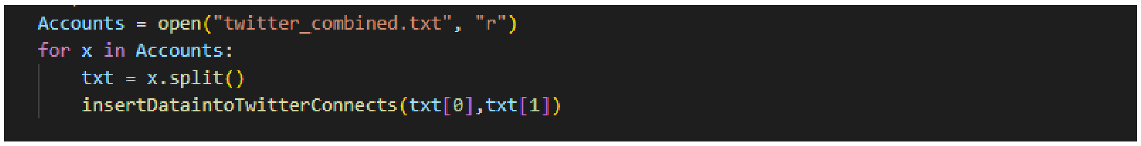
Lessons Learned :

From this task, there are several important things to learn.

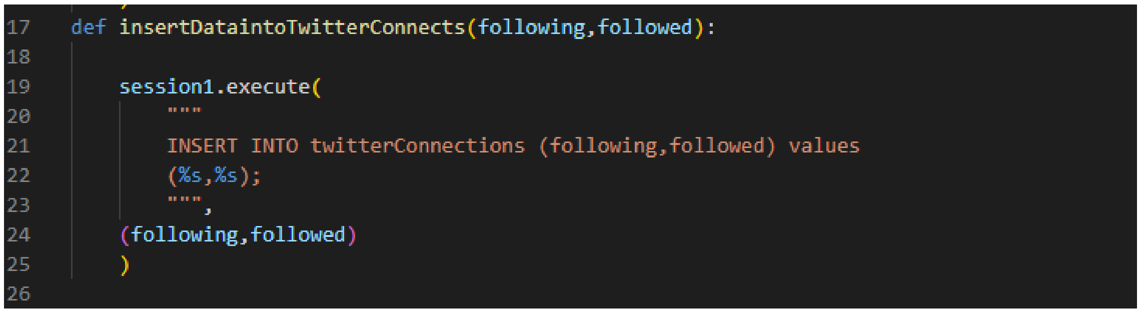
- Creating the single node of Cassandra with Docker is always easier, but creating the cluster is a difficult task in itself. The main part of this is how to map the ports from one node to the other parent node so that they could be synced externally. Another Challenge was to make this possible on the same network with several containers instead of VMs as VMs can easily have different IP addresses but containers cant.
- Making the pipeline to insert the data, specifically making the connection with the python. Several APIs are available on the internet but finding the right driver for the right API was difficult.
  I.e: for drivers there are different driver packages available on the internet like SIMBA or CDATA and letting know which one is correct is difficult.
- The main concept of Keys in PKs is easier, but when we try to make the Cluster keys with them with commands (with clustering order by) in creating table commands, was very difficult. Specifically the main hurdle was to figure out the order of the keys like first which key should be made (in case of composite keys).
- Accessing the Casandra.Yaml file for writing was also a difficult part as VIM or any other editor was not available image defined in Docker to enable the Materialised views and SASI indexes, I had to move the files from the container in local system, edited them and at last replaced them with the existing original files.

Important Tip
For this task, having the knowledge about ETL processes or pipelines can make it a lot easier. After Learning about the ETL process while I was roaming on the internet about how to load data into Cassandra, I came across different concepts of ETL using several ETL tools. Like in the case of Cassandra, it is believed that instead of making complex queries on the same table, making several tables is easier and preferred. So While Creating joins in Cassandra as it is not possible to make in Cassandra, so

I found that different ETL tools are helpful for this. Like they just read data from the sources, cache them, joins them and then give output in whichever format we like like in csv, db output or screen display. While going through these, I came across a tool called “Talend " which is free and open source. But the thing I learnt then is the driver packages they use to connect with databases specifically for Cassandra are paid while other SQL Database drivers were free. So If we have to work a lot with Cassandra, then it would be a good idea to go with the subscription of those drivers to make things easier. Another main advantage of Talend like etl tool is instead of creating all the methods on python or any other language manually and putting a lot of effort, we can just use them and in which we can drag and drop the components for each task. One more thing about this tool is that it is even more faster to use, like



```
Accounts = open("twitter_combined.txt", "r")
for x in Accounts:
    txt = x.split()
    insertDataintoTwitterConnects(txt[0],txt[1])
```



```
def insertDataintoTwitterConnects(following,followed):

    session1.execute(
        """
        INSERT INTO twitterConnections (following,followed) values
        (%s,%s);
        """,
    (following,followed)
    )

```

If we look at the code above, it is reading the data from csv line by line and invoking the insert statement one by one and every time the database connection is opened and committed or in short we can also say that we don’t have any bulk insertion option.

But at the same time, in case of ETL tools we have an advantage of bulk insertion, like in which we can insert the bulk of data together which will also initiate the commit statement at once after the batch is inserted (we can set the size of batch manually too).