Mohammadamin Sheikhtaheri
0930853

Jan 27, 2020

Cloud Computing Assignment 1 - Task 3

Task 1: Implementation & Documentation

For Task 1 of this project, it was super easy to set up boto3 for AWS in python in order to get the required buckets up and running. The only inconvenience I personally had was copying and pasting the required tokens in the credentials file every few hours due to them expiring. The implementation of uploading files to specific buckets and also displaying them was very straight-forward due to the amount of helpful documentation that exists on the web, mainly on the AWS site itself. Implementing the Azure containers was definitely more complicated and time consuming than it was with the AWS buckets, which is mainly due to the lack of helpful documentation/tutorials on the web of the Azure Python SDK. Overall, Azure's Python API is significantly more difficult to understand than AWS when considering storage.

Task 1: Performance

In terms of performance, the time it took to complete identical tasks in both AWS and Azure were recorded using the Python *time* library. Looking at the comparisons, it is very clear that the Azure SDK performs tasks much quicker than the AWS boto3 SDK, which is also visually apparent. Times taken to accomplish identical tasks by both AWS and Azure are listed below:

Task	AWS (s)	Azure (s)
Uploading objects	17.5	6.6
Displaying all objects	1.2	0.4
Displaying objects in specific containers	0.5	0.05
Displaying a specific object in any container	1.4	0.2
Downloading object	6.1	2.1

Task 2: Implementation & Documentation

For task 2 of this project, the AWS implementation was definitely more complex than it was for task 1. The documentation that was available on the web was helpful, but it was still quite difficult to manipulate the various functions to accomplish the tasks we were given for the project. The documentation provided by Amazon for the DynamoDB resource was not very in depth, only providing a couple of examples of basic queries and scans. Azure CosmosDB on the other hand had decent online documentation. Creating a table using CosmosDB was simpler than DynamoDB, however populating the table was troublesome due to the restrictions of the Azure Table. CosmosDB tables do not allow the partition and row keys (year and title) to have certain characters which were present in the moviedata.json we were provided. Azure Table also converted any integers to base64 before entering them to the table, which made querying certain attributes very difficult. Overall, both DynamoDB and CosmosDB have their difficulties, however, although writing queries was simpler with CosmosDB, it is much more finicky with data than DynamoDB.

Task 2: Performance

In terms of performance, once again Azure CosmosDB was much faster compared to AWS DynamoDB when performing identical tasks. The cause of this might be due to the AWS SDK instantiating more complex objects than necessary. Below is a list of times it took for both AWS and Azure to perform identical tasks:

Task	AWS (s)	Azure (s)
Creating a table	0.4	0.9
Populating the table with movies	221	128
Performing a query based on year of movie released being 1997	0.5	0.2
Performing a query based on year of movie released after 1950		3.7

References

[1] Amazon AWS, “Quickstart - Boto 3 Docs 1.11.9 documentation,” https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html. [Accessed: 23 Jan 2020].

[2] Mhopkins-Msft, “Quickstart: Azure Blob storage library v12 - Python,” https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-python. [Accessed: 23 Jan. 2020].

[3] Amazon AWS, “DynamoDB Getting Started Python,” https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GettingStarted.Python.04.html [Accessed 24 Jan. 2020].

[4] Amazon AWS, “Using Sort Keys to Organize Data in Amazon DynamoDB,” https://aws.amazon.com/blogs/database/using-sort-keys-to-organize-data-in-amazon-dynamodb/ [Accessed 24 Jan. 2020].

[5] W3 Schools, “Python File Write,” https://www.w3schools.com/python/python_file_write.asp [Accessed 24 Jan. 2020].

[6] Zet Code, “Pretty Table Documentation,” http://zetcode.com/python/prettytable/ [Accessed 24 Jan. 2020]

[7] Microsoft, “Table Storage How to Use Python,” https://docs.microsoft.com/en-us/azure/cosmos-db/table-storage-how-to-use-python [Accessed 25 Jan. 2020]

[8] Geeks For Geeks, “Python String Replace,” https://www.geeksforgeeks.org/python-string-replace/ [Accessed 25 Jan. 2020]