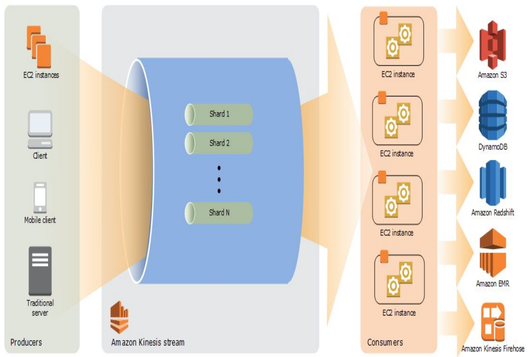
Amazon Kinesis Data Stream

Tife Olatunji and Sharon Mbaegbu
Department of Big Data Analytics, AMOD Program, Trent University at Peterborough Ontario

INTRODUCTION

This project examines the capabilities of Amazon Kinesis Data Streams within the AWS ecosystem, particularly focusing on the 'put_record' and 'get_record' APIs. Our exploration is aimed at uncovering the practical uses of these APIs in streamlining data management processes.



Kineses Data Stream High Level Architecture

https://docs.aws.amazon.com/streams/latest/dev/key-concepts.html

AIM

Our objective is to conduct a comprehensive evaluation of Amazon Kinesis. By converting CSV data into JSON and utilizing AWS CLI for stream processing, we seek to assess the scalability and performance of Kinesis in various data handling scenarios.

METHOD

- **Set up AWS CLI:** We initiate the command line interface to interact with AWS services directly.
- **Prepare JSON Datasets:** Actively converting CSV data into JSON format to ensure compatibility with Kinesis.
- **Stream Data Using APIs:** Employ 'put_record' and 'get_record' APIs for seamless data streaming.
- **Process Data:** We engage in real-time data processing, emphasizing efficiency and speed.
- **Performance Evaluation:** We test and document Kinesis performance across various data loads.

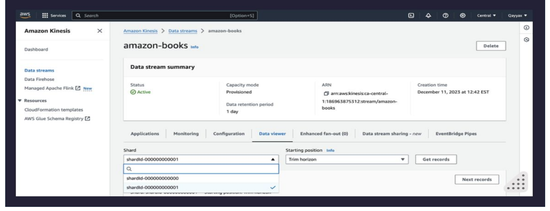
```
aws configure

# > accesskey id
# > secret accesskey
# > region name (default one you picked)
```

Image of AWS Command Line Setup

```
bash
aws kinesis create-stream --stream-name amazon-books --shard-count
2
```

This creates an Amazon Kinesis stream named amazon-books with 2 shards



Creation of Kinesis Stream and Shard Counts

```
# Put record into the Kinesis stream
response = kinesis_client.put_record(
    StreamName=stream_name,
    Data=record_data,
    PartitionKey=partition_key
)
```

Application of Put_record API to the Kinesis stream

RESULTS

The performance evaluation of Amazon Kinesis indicates its effectiveness in real-time data workflow management. Our data shows Kinesis can process 212 records in just over 10 seconds and 8786 records in 276 seconds, evidencing its capacity for scalability and operational efficiency.

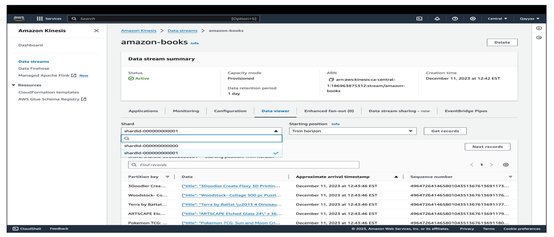
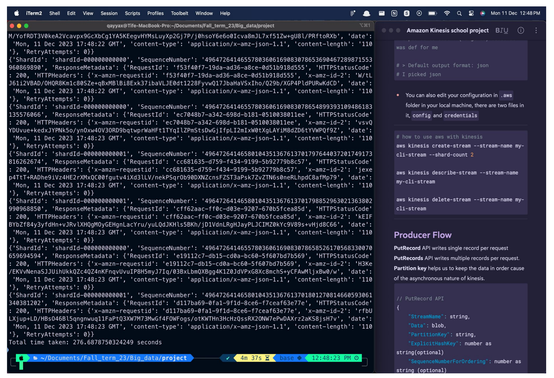


Image of Amazon Books Data Stream and Shard Counts

```
~/Documents/Fall_term_23/Big_data/project  1m 46s  base  02:45:52 PM
python process.py
Total number of books: 8786
Total price of books: 357758.99599999085
Average price of books: 40.71921192806739
~/Documents/Fall_term_23/Big_data/project  6s  base  02:46:01 PM
```

Queries processed from Amazon Books Kinesis Stream



Total time taken to upload the records into the Kinesis stream

CONCLUSIONS

The project concludes with the affirmation of Amazon Kinesis's adept handling of large data volumes and its potential as a cost-efficient solution for large-scale data streaming. These results solidify Kinesis as a reliable choice for real-time data processing within cloud-based infrastructures.

ACKNOWLEDGEMENTS

We express our thanks to our colleagues and professors of this study. Your support and contributions have been pivotal to our project's achievements.