Class Exercises - Hack Week 10

Experimenting with Data Analysis on AWS

Build a “stock exchange”

- Generates “trades”
- Monitors trading activity for anomalies
- Halts trading when anomalies are detected

Here is a script that simulates trading activity:
https://s3.amazonaws.com/mpcs-resources/mpcs_ticker.py

Events look like this:

```
{
      "symbol": "SNA",
      "price": 148.5,
      "size": 56900,
      "trade_time": "1496533444",
      "id": "aed6b2bd-174a-45e2-82af-68feb48cc1a4"
}
```

To run the ticker: `python mpcs_ticker.py produce 1.5` (generates a quote every 1.5 seconds)
If you want to test that your events are being put into the stream you can consume them by running:
`python ticker.py produce 0.5` (pulls data from stream every 0.5 second)

Our system needs to do the following:

- Create a Kinesis stream and put trade events into the stream

- Persist the trade information to DynamoDB (call the table `<username>-trades`)
  - boto3 complains about storing floats in DynamoDB so to store `price` you must do:
    ```
    from decimal import Decimal
    data = {..."price": Decimal(str(trade['price'])), ...}
    trades_table.put_item(Item=data)
    ```

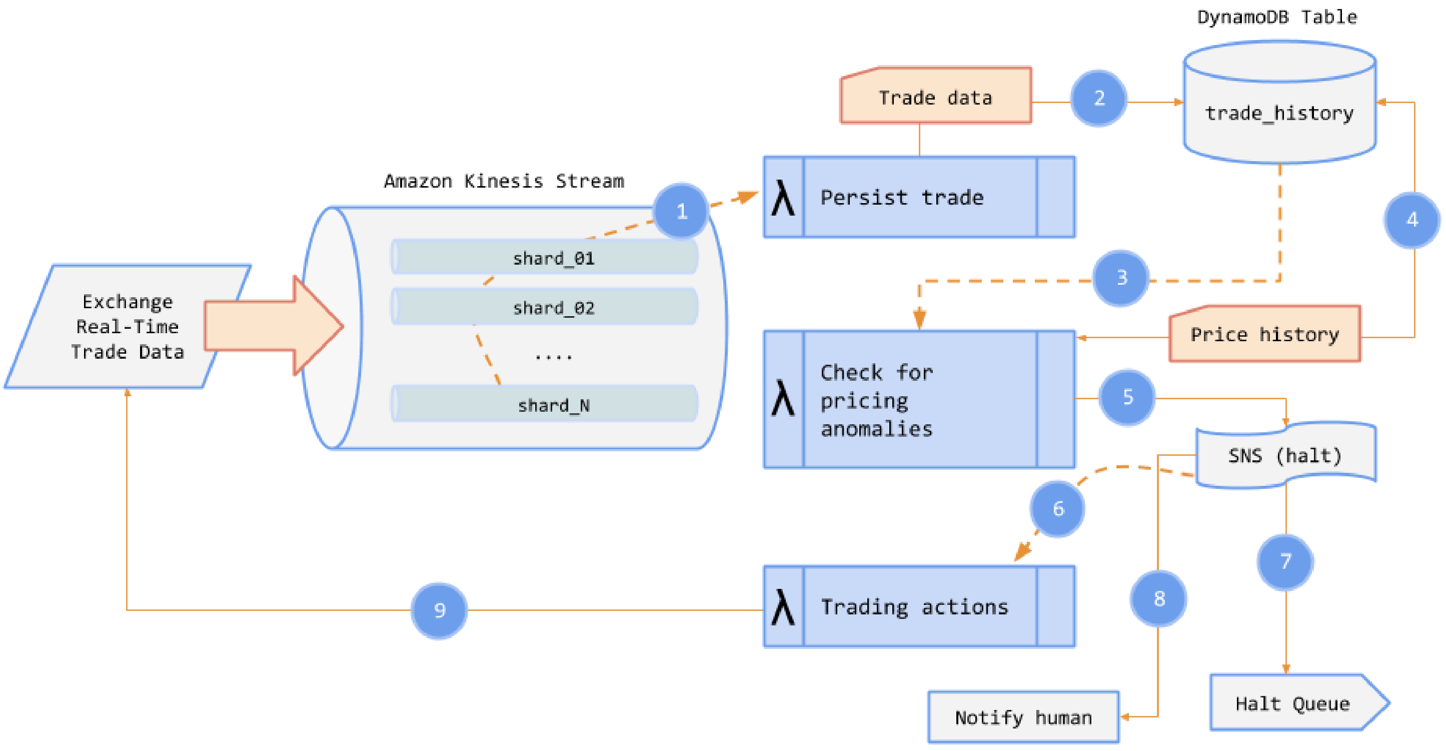
- Check if a trade is an anomaly. Our rules for this are:
  ```
  trade_price > average of trade prices in last N seconds * 1.5 OR
  trade_price < average of trade prices in last N seconds * 0.4
  ```

- If it detects an anomaly it must halt trading for the stock for 10 seconds; then everything goes back to normal.

Useful link: http://docs.aws.amazon.com/lambda/latest/dg/with-kinesis.html

When creating lambdas you can use the "`Access_for_Lambda_MPCS`" IAM role to allow them access to our environment.

You can add more provisioned capacity to the DynamoDB trades table if needed (default is 5 read/write units; you can go up to 100).



DynamoDB Table
Trade data
2
trade_history
Amazon Kinesis Stream
1
λ Persist trade
4
shard_01
Exchange Real-Time Trade Data
shard_02
3
Price history
....
λ Check for pricing anomalies
shard_N
5
SNS (halt)
6
7
9
λ Trading actions
8
Notify human
Halt Queue