Junior Analytics Engineer

Take Home Quiz

Overview

At Metrika Inc. we're building the world's premier Operational Intelligence and Risk Management platform for Blockchain networks. This involves consuming a large number of transactions/blocks from various blockchain networks and analyzing them in order to draw operational insights and assess their overall health and reliability.

In this exercise you are asked to perform three tasks (2 required and 1 bonus): (1) write a data ingestion script in Python that consumes blocks from a REST API [details below], (2) analyze the output data of your ingestion script to calculate some key operational KPIs, and (3 - bonus) given the community-heavy aspect of blockchain, explain how you would engage with different members of a blockchain community to collect feedback for the Metrika platform.

You should use Python 3.9+ for tasks (1) and (2).

The Exercise

Task 1: Data Ingestion Script

Metrika is closely involved with a number of next-generation blockchain protocol creators, one of whom is Algorand. Algorand has built a tool called the "V2 Indexer" which allows developers to search the Algorand Blockchain (i.e., searching all blocks committed to the ledger, along with their transactions). We are providing you with a V2 Indexer REST API with the following credentials:

- API token: `""`
- Indexer address: `"https://mainnet-algorand.api.purestake.io/idx2"`
- Headers: `{"X-API-Key": "8bIrGK8Wrm8r54ngNWgAI4ECnaZQNZ1x1km9YVQb"}`

Your task is to create a data ingestion script in Python that consumes the provided API using the `/v2/blocks/{round-number}` endpoint. Your ingestion script should:

- Accept as input a starting and an ending block, and process all blocks in between. For the purposes of this exercise use `20,923,000` and `20,926,100` as starting and ending blocks respectively. The API rate limit is 10 requests/second.
- For each block your script will process, fetch the block header as well as all transactions of either type `payment-transaction` or `asset-transfer-transaction` (you can ignore all other transaction types) and store them in a medium of your choice (flat file or DB). We recommend reading through Task 2 before deciding how you will structure/store your data.
- Use poetry to manage Python packages
- Be accompanied by unit tests
- Have proper error handling and logging

Task 2: Operational KPIs

Analyzing the volume of transactions that are being handled by a blockchain network as well as the time it takes for new blocks to be committed to the ledger are critical when evaluating the network's performance and scalability. Some frequently used metrics in the blockchain space include Transactions Per Second (TPS), number of transactions per block, and block interval time (i.e., time between two subsequent blocks).

- What is the average TPS for this network? Does it vary over time? Provide visualizations showing the spread of the TPS metric.
- Create a time series plot showing the block interval time for the entire data set.
- Is there a relationship between block interval time and number of transactions per block? Explain your approach, including appropriate visualizations and/or metrics.
- Do you observe any anomalies in the block interval time? If yes, please provide visualizations and or/metrics to support your observation. Please also provide

an explanation of how you would quantify anomalies in this dataset, a simple model or a description of possible techniques in the report will suffice.

Technical Interview: What to Expect?

After you submit your solution to the exercise, our Analytics team will carefully review it. If we decide to move forward, we will schedule a technical interview with you. During the technical interview we will discuss your submission in detail. Furthermore, be prepared to discuss topics such as:

- How you would improve your code given more time
- Best practices for coding, and considerations when taking your code to production (e.g., monitoring, scalability)
- Data wrangling and visualizations

Deliverables:

- Python 3.9+ code that performs the requested tasks (no jupyter notebooks!)
- Accompanying unit tests
- A short report (10 pages maximum):
    - Describe your approach to tasks (1) and (2) describing your approach as well as your findings for (2).
    - Highlight ways in which your solution/analysis could be improved.
    - Your submission for task (3)

You should use git, locally, and make well thought structured commits there. Please send the working directory, including the .git subdirectory, via email to alex@metrika.co.

Please note that this test, including your solution to it, are confidential and should not be published publicly, in any form, electronic or otherwise.

Good luck!