**Steps to extract Ethereum account addresses:**

1. Run the ‘Ethereum\_addresses\_extractor.py’ code to extract the addresses for Ethereum accounts and save them in a CSV file. The ‘Ethereum\_addresses\_extractor.py’ code performs the following operations: (Note: There are more than 268 million Ethereum addresses as of 8 May 2024. Extracting these addresses would take an enormous amount of time, considering the limit of 100,000 API calls/per day for a free Etherscan account. Consequently, the only finalized last 100,000 blocks as of 1714521873 timestamp (19673900 – 19773900) are queried to extract the addresses involved in transactions within these blocks. 100,000 is selected based on the daily API call limit.)
   1. Defines the function to extract transactions for a block
   2. Defines the function to extract addresses for the transactions
   3. Defines the function to save addresses to a CSV file

(Note: To speed up the addresses extraction process, the ‘Ethereum\_addresses\_extractor.py’ code is run in parallel by creating five copies of the code, and then each code extracts addresses from 20,000 blocks. Five is selected as partition size based on the 5 API calls/second limit of Etherscan. The five copied codes, the code to parallelize the five codes, and the output files are in the temp folder within this folder. The five codes will generate ‘output\_1.csv’, ‘output\_2.csv’, …, and ‘output\_5.csv’ files. The number of addresses in each file is in millions. However, Excel has a limit of 1,048,576 rows and 16,384 columns per sheet. Opening these files in Windows Excel software will truncate the rows after 1,048,576. Consequently, these files are open using cloud Excel software: <https://rowzero.io/>. Each file is opened using the rowzero application, then the file is split into different Excel files such that each file contains 1,000,000 rows. The split files are ‘output\_1\_part1.csv’, ‘output\_1\_part2.csv’, …, ‘output\_5\_part1’, and ‘output\_5\_part2.csv’.)

**Steps to preprocess Ethereum account addresses:**

1. Run the ‘Address\_preprocessor\_concatenator.py’ code to extract the preprocess and concatenate the addresses stored in the CSV files (obtained from the previous step). The final list of addresses is saved in a text file. Text file is chosen over CSV file for saving the data to avoid the excel limit of 1,048,576 rows. The ‘Address\_preprocessor\_concatenator.py’ performs the following operations:
   1. Defines the function to preprocess an excel file. First it converts the addresses into lowercase to remove duplicates. Then, the rows with missing values are dropped; the duplicates are removed. The addresses that do not begin with ‘0x’ are removed as they do not represent a valid Ethereum address. The addresses are stored in a list.
   2. Define the function to process all files in the folder
   3. The list containing final addresses is saved in the ‘ethereum\_addresses.txt’ file

**Steps to extract transactions for Ethereum addresses:**

1. Run the ‘Ethereum\_transactions\_extractor\_json.py’ code to extract the transactions and save them to a text file. The ‘Ethereum\_transactions\_extractor\_json.py’ code performs the following operations:
   1. Reads the Ethereum addresses from the ‘ethereum\_addresses.txt’ file
   2. Extracts transactions for each address and saves the transactions to a csv file.

**Steps to convert JSON transactions data into text format:**

1. Run the ‘Ethereum\_transactions\_dataset.py’ code to extract the transactions and save them to a text file. The ‘Ethereum\_transactions\_dataset.py’ code performs the following operations:
   1. Reads the Ethereum transactions data saved in the json format
   2. Saves the transaction data in the text format