Crypto Market Pool
Blockchain Engineer Resource

Use The Graph to query Ethereum data in Python

This is a step by step guide to help you use The Graph to query Ethereum data in Python. The Graph project is an indexing system for querying decentralized networks. You can use The Graph to query systems like Ethereum, IPFS, etc.

Before we get started lets go over a few definitions:

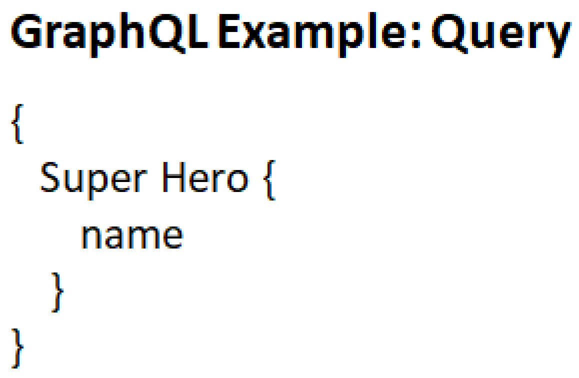
- **GraphQL** is a query syntax language
- **The Graph** is a blockchain project that uses the GraphQL query language. The project allows anyone to build and publish an open API called a **subgraph**

The Graph (project) uses GraphQL which is a syntax language that describes how to ask for data. This syntax isn't tied to a specific type of database or storage engine and is instead backed by your existing code and data. GraphQL is about asking for specific fields from an object.

GraphQL

Let's start by looking at a very simple GraphQL query structure and the results we get when we run it. Once a GraphQL service is running, it can receive GraphQL queries to execute. The service checks the query to ensure it only refers to the types and fields defined, and then runs the function to produce a result.

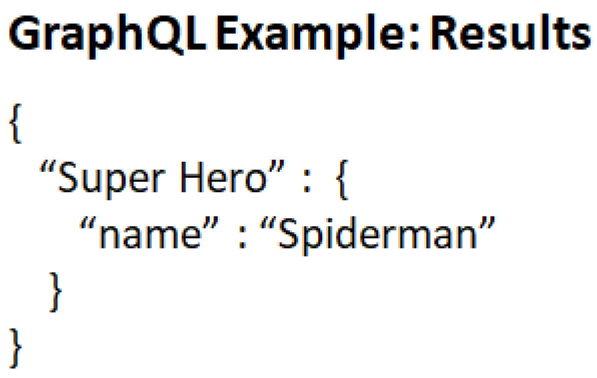
As an example view the structure of the query below:



GraphQL Example: Query

```
{
  Super Hero {
     name
   }
}
```

The GraphQL query above could produce the results below:



GraphQL Example: Results

```
{
  “Super Hero” : {
     “name” : “Spiderman”
   }
}
```

You can see that the query has the same structure as the result. This is essential to GraphQL and as a result the server knows exactly what fields the client is asking for.

Use GraphQL to:

- Search for data
- Pass arguments in your request
- Use aliases
- Fragments and more

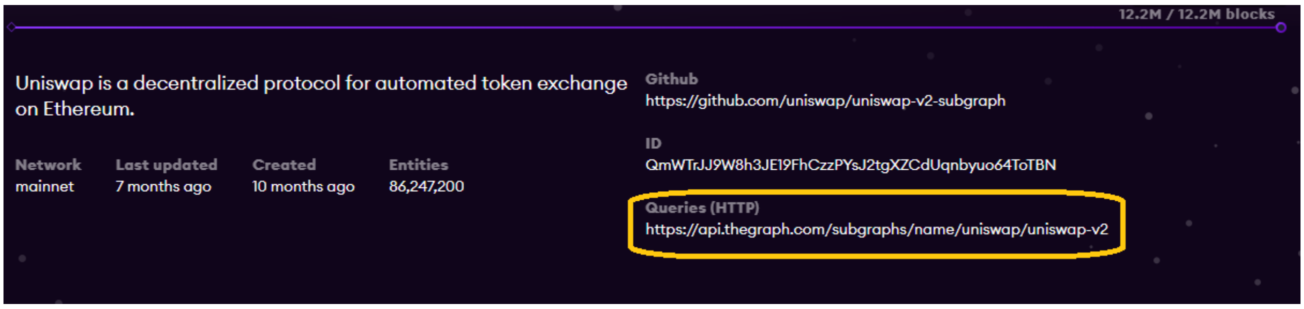
Visit GraphQL to learn more about how to write complex GraphQL queries.

The Graph

To get a better understanding of what The Graph project is and how it works visit thegraph.com/docs. It explains how to deploy a subgraph and how to query the subgraph for data. A subgraph defines which data The Graph will index from Ethereum, and how it will store it. Once a subgraph is deployed it can be queried using the GraphQL syntax.

For this tutorial we will focus on querying data from a subgraph.

1. Visit The Graph Explorer to view all of the hosted subgraphs that exist for the Ethereum blockchain. Each one of these hosted services (subgraphs) can be queried for data.
2. Select a subgraph page and take notice of the http query address and the playground section of the page
3. The http query address is needed in your Python code and is the endpoint that contains the blockchain data. This service will execute your GraphQL queries.


12.2M / 12.2M blocks

Uniswap is a decentralized protocol for automated token exchange on Ethereum.

Network	Last updated	Created	Entities
mainnet	7 months ago	10 months ago	86,247,200

Github
https://github.com/uniswap/uniswap-v2-subgraph

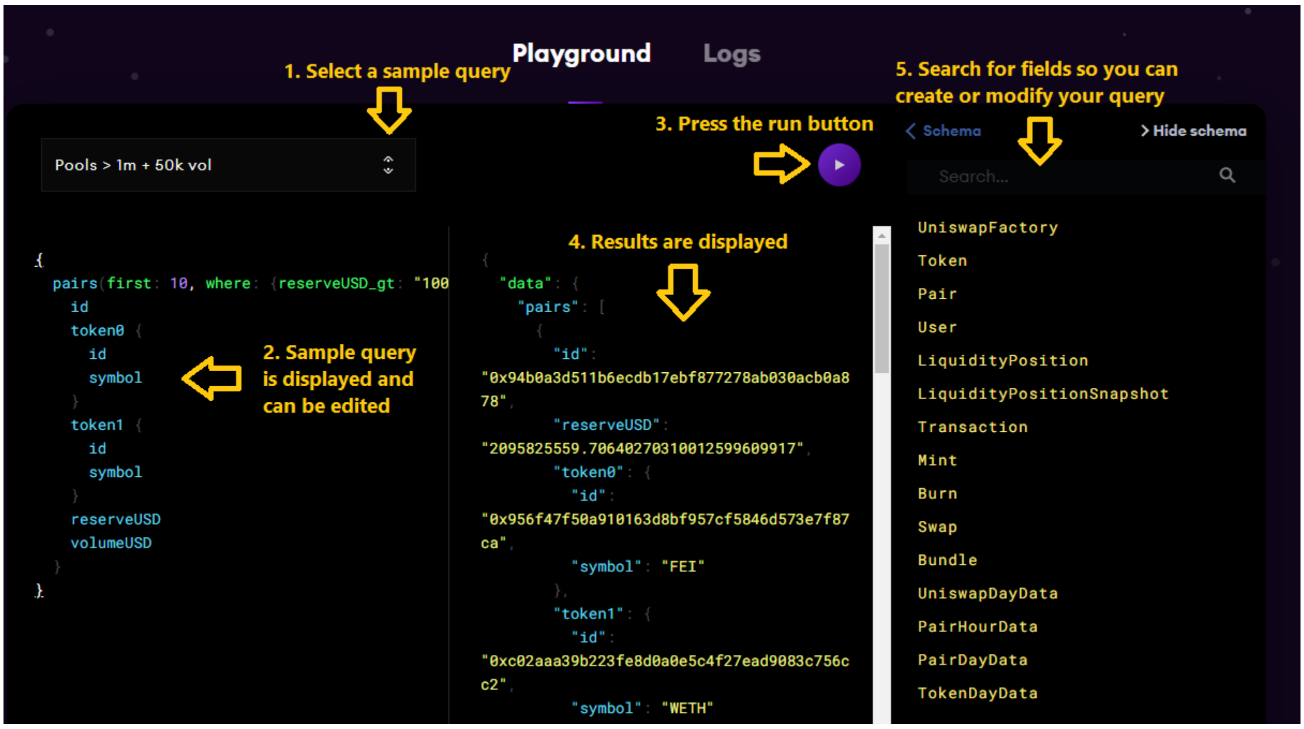
ID
QmWTrJJ9W8h3JE19FhCzzPYsJ2tgXZCdUqnbyuo64ToTBN

Queries (HTTP)
https://api.thegraph.com/subgraphs/name/uniswap/uniswap-v2

4. Make sure you experiment with the Playground. This section of the website will allow you to construct and test out your GraphQL Ethereum blockchain queries.

- Select a sample query
- The sample query is displayed and can be edited
- Press the run button
- Results are displayed

- The far right side of the screen displays a list of fields that you can add to your query



```
{
  pairs(first: 10, where: {reserveUSD_gt: "100
    id
    token0 {
      id
      symbol
    }
    token1 {
      id
      symbol
    }
    reserveUSD
    volumeUSD
  }
}
```

```
{
  "data": {
    "pairs": [
      {
        "id":
"0x94b0a3d511b6ecdb17ebf877278ab030acb0a8
78",
        "reserveUSD":
"2095825559.70640270310012599609917",
        "token0": {
          "id":
"0x956f47f50a910163d8bf957cf5846d573e7f87
ca",
          "symbol": "FEI"
        },
        "token1": {
          "id":
"0xc02aaa39b223fe8d0a0e5c4f27ead9083c756c
c2",
          "symbol": "WETH"
```

Using The Graph in Python

Next lets use a few of the queries that we constructed in The Graph's Playground and use it in our Python code to request different data from the Ethereum blockchain.

The sample Python code below contains a generic function that is used to make a post request to a subgraph. To use different subgraphs you need to change the url end point and GraphQL syntax. I included a print statement and pretty print statement (easier to read) at the end of the program so the results from the Ethereum blockchain will print out in your console.

Example 1: Query Aave on the Ethereum blockchain for a list of the last 10 flash loans by time stamp using GraphQL in Python

```
import requests
# pretty print is used to print the output in the console in an easy to read
from pprint import pprint
```

```
# function to use requests.post to make an API call to the subgraph url
def run_query(q):

    # endpoint where you are making the request
    request = requests.post('https://api.thegraph.com/subgraphs/name/aave/pr
                            '',
                            json={'query': query})
    if request.status_code == 200:
        return request.json()
    else:
        raise Exception('Query failed. return code is {}.      {}'.format(re


# The Graph query - Query aave for a list of the last 10 flash loans by time
query = """

{
flashLoans (first: 10, orderBy: timestamp, orderDirection: desc,){
  id
  reserve {
    name
    symbol
  }
  amount
  timestamp
}
}
"""
result = run_query(query)

# print the results
print('Print Result - {}'.format(result))
print('#############')
# pretty print the results to make it easier to read
pprint(result)
```

Example 2: Query Uniswap on the Ethereum blockchain for a list of the top 10 pairs using GraphQL in Python

The query below is a leaderboard for Uniswap that details the top ETH liquidity providers in descending order of ETH deposited. This can help you better analyze user behavior, like keeping track of top market movers and observing the relationship between liquidity providers of ETH vs. other tokens. Other fields that can be queried about users include their address, assets bought and sold historically and total fees paid by that user.

```
import requests
# pretty print is used to print the output in the console in an easy to read
from pprint import pprint


# function to use requests.post to make an API call to the subgraph url
def run_query(q):

    # endpoint where you are making the request
    request = requests.post('https://api.thegraph.com/subgraphs/name/uniswap
                            '',
                            json={'query': query})
    if request.status_code == 200:
        return request.json()
    else:
        raise Exception('Query failed. return code is {}.      {}'.format(re


# The Graph query - Query Uniswap for a list of the top 10 pairs where the r
query = """

{
  pairs(first: 10, where: {reserveUSD_gt: "1000000", volumeUSD_gt: "50000"},
    id
    token0 {
      id
      symbol
    }
    token1 {
      id
      symbol
    }
    reserveUSD
    volumeUSD
  }
}
"""
result = run_query(query)

# print the results
print('Print Result - {}'.format(result))
print('#############')
# pretty print the results
pprint(result)
```

Using The Graph to query Ethereum data in Python is powerful. There is a lot of data that can be queried for report generation and analysis.

This code is for learning and entertainment purposes only. The code has not been audited and use at your own risk. Remember smart contracts are experimental and could contain bugs.

Next – Build a snipe bot to monitor and trade liquidity pairs

Ledger Nano X - The secure hardware wallet

RESOURCES

How to add an image / logo to your crypto token

If you are the owner of a token on the Ethereum network or the Binance Smart Chain (BSC) you can...

PYTHON & WEB3 BASICS

How to airdrop crypto to multiple accounts using Python

An airdrop is a marketing method that involves sending crypto tokens to many accounts programmatically in order to promote awareness...

Leave a Reply

You must be logged in to post a comment.

RECENT POSTS

Vaults and the ERC-4626 token contract

What is an ERC-1404 token contract

Dynamic NFTs on Ethereum

ERC721 contract that supports sales royalties

Mint a Scary Ghost NFT on Polygon

Create a crypto faucet using a Solidity smart contract

ADVERTISEMENT

CRYPTO CURRENCY REDDIT

After months of research im beginning to think crypto is our only shot at surviving the upcoming financial collapse

Defi scammers be like

The Porn Star Banging Men Into Not Buying Crypto and NFTs

3 African countries Are Considering To Adopt Cryptocurrency

Kraken Rant

SUBSCRIBE

Get notified when new articles are added to Crypto Market Pool

Email *

Submit

test