In [1]:

```
import json
import requests
import pandas as pd
from datetime import datetime
import time
```

In [2]:

```
# Convert timestamp to date
# Used in constructing the URL of the getDict_RvnBlocks function

def getDate(timestamp):

    return time.strftime('%Y-%m-%d', time.localtime(timestamp))
```

In [3]:

```
# Get block data from https://ravencoin.network

def getDict_RvnBlocks(timestamp):

    date = getDate(timestamp)

    url = "https://ravencoin.network/api/blocks?blockDate="+date+"&startTimestamp="+str(tim

    # Obter dados da web
    response = requests.get(url)

    # Converter resposta em Dict
    return json.loads(response.text)
```

In [4]:

```
# Get the timestamp of the next block to be extracted

def getNextTimestamp(responseDict):

    nextTimestamp = False

    if responseDict["blocks"][-1]['height'] > 1:
        nextTimestamp = responseDict["blocks"][-1]['time']
    else:
        print("Última página")
        print(responseDict["pagination"])

    return nextTimestamp
```

In [5]:

```
# Gets the relation of blocks in a Pandas data frame

def getDF_RvnBlocks(responseDict):

    return pd.DataFrame(responseDict["blocks"])
```

In [8]:

```
# Loop to extract Data from RVN Blocks
def getRvnBlocks(timestamp):

    i = 1
    j = 1
    nextBlock = timestamp
    responseDict = None
    df = pd.DataFrame(columns = ["height","size","hash","time","txlength","poolInfo","isMai

    print(f'Iteração: {i}: bloco {nextBlock}')

    while nextBlock:

        responseDict = getDict_RvnBlocks(nextBlock)

        dfTemp = getDF_RvnBlocks(responseDict)

        df = pd.concat([df, dfTemp], axis=0)

        if j == 2500:
            print(f'Pausa de 5 segundos a cada {j} iterações')
            j = 0
            df.to_csv(r'C:\temp\rvnTemp.csv')
            time.sleep(5)

        i += 1
        j += 1

        nextBlock = getNextTimestamp(responseDict)

        print(f'Iteração: {i}: bloco {nextBlock}')



    return df
```

In [9]:

```
# Extract blocks from 05/31/2021 to the first block, timestamp = 1622516262

dfOriginal = getRvnBlocks(1622516262)
dfOriginal
```

```
Iteração: 1: bloco 1622516262
Iteração: 2: bloco 1622504185
Iteração: 3: bloco 1622492305
Iteração: 4: bloco 1622480021
Iteração: 5: bloco 1622468446
Iteração: 6: bloco 1622456444
Iteração: 7: bloco 1622443866
Iteração: 8: bloco 1622431005
Iteração: 9: bloco 1622419272
Iteração: 10: bloco 1622408029
Iteração: 11: bloco 1622395205
Iteração: 12: bloco 1622383092
Iteração: 13: bloco 1622371164
Iteração: 14: bloco 1622359868
Iteração: 15: bloco 1622347067
Iteração: 16: bloco 1622334973
Iteração: 17: bloco 1622323281
Iteração: 18: bloco 1622311200
```

In [10]:

```
# Export the data
dfOriginal.to_csv('Datasets/rvnOriginal.csv')
```

In [14]:

```
dfOriginal.tail(10)
```

Out[14]:

	height	size	hash	time	txlength
182	10	262	000000574a871aca8b41b39e03cad1f00bf5e0fdcf477b...	1515015971	1
183	9	262	000000488c4ed0a6df591c62e8a2600fc1e25690ea8ca4...	1515015970	1
184	8	262	000000e2e9415eeeabe677f6f2f1df0e776ee5c5322f1e...	1515015936	1
185	7	262	0000003a6463eaf8a71dbde333e61b713ed56a37eb8cac...	1515015905	1
186	6	262	0000006d7a8490b5c4a7e41bb5549f905c75b59f902db8...	1515015847	1
187	5	262	000000a123ae742de194b6db9f3768e2a1dcd43470912d...	1515015840	1
188	4	262	0000002a5f3b73c4366e4d2614e62a166ed935fd8f85d2...	1515015833	1
189	3	262	0000003a02e3de84a46ced09886c4809ebdeca4532e002...	1515015816	1
190	2	262	000000e2aa6490a97dd26301516ce1ec3fcc5a9cea3bb3...	1515015759	1
191	1	262	00000058bcc33dea08b53691edb9e49a9eb8bac36a0db1...	1515015723	1