

HW2: Exploring Bitcoin transactions - Report

Ramandeep Singh
8019-7991

Tool Used:-

Pandas - Pandas is a python library used for data analysis.

SQLite- An SQLite database can be read directly from pandas library and can be used to implement SQL concepts on the large CSV file for data analysis.

Installation Steps:-

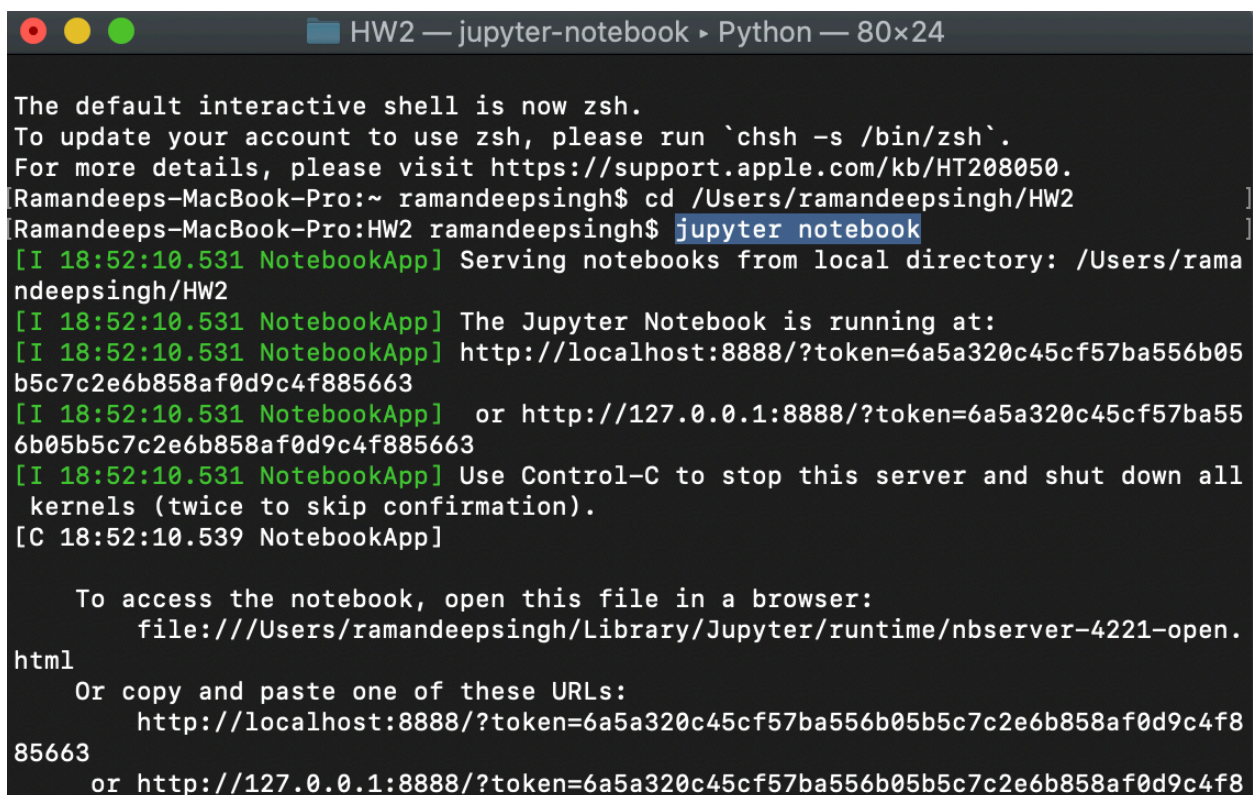
Run below commands in terminal to install pandas library, Jupyter notebook and sqlite in pandas

```
pip3 install pandas
pip3 install notebook
pip3 install -U pandasql
```

To run Jupyter notebook run below command

```
jupyter notebook
```

The above command should be run from the same path where csv files are placed.



```
HW2 — jupyter-notebook ▸ Python — 80x24

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Ramandeeps-MacBook-Pro:~ ramandeepsingh$ cd /Users/ramandeepsingh/HW2
Ramandeeps-MacBook-Pro:HW2 ramandeepsingh$ jupyter notebook
[I 18:52:10.531 NotebookApp] Serving notebooks from local directory: /Users/rama
ndeepsingh/HW2
[I 18:52:10.531 NotebookApp] The Jupyter Notebook is running at:
[I 18:52:10.531 NotebookApp] http://localhost:8888/?token=6a5a320c45cf57ba556b05
b5c7c2e6b858af0d9c4f885663
[I 18:52:10.531 NotebookApp] or http://127.0.0.1:8888/?token=6a5a320c45cf57ba55
6b05b5c7c2e6b858af0d9c4f885663
[I 18:52:10.531 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[C 18:52:10.539 NotebookApp]

To access the notebook, open this file in a browser:
file:///Users/ramandeepsingh/Library/Jupyter/runtime/nbserver-4221-open.
html
Or copy and paste one of these URLs:
http://localhost:8888/?token=6a5a320c45cf57ba556b05b5c7c2e6b858af0d9c4f8
85663
or http://127.0.0.1:8888/?token=6a5a320c45cf57ba556b05b5c7c2e6b858af0d9c4f8
```

Part 1: Transactions analysis

1. What is the number of transactions and addresses in the dataset?

Number Of Transactions - 10000055

Number of Addresses - 8385065

Code:-

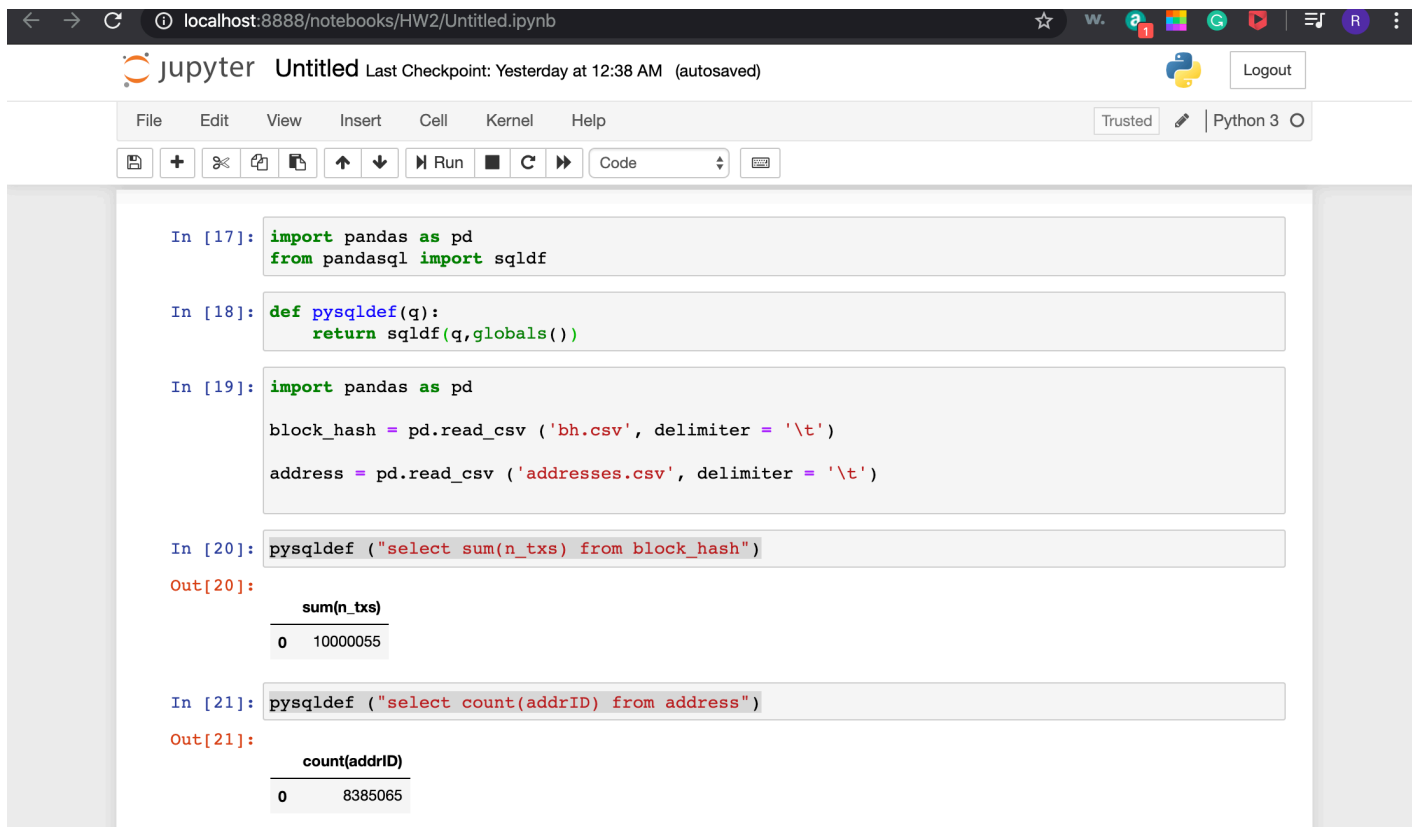
```
import pandas as pd
from pandasql import sqldf
```

```
def pysqldf(q):
    return sqldf(q,globals())
```

```
import pandas as pd
block_hash = pd.read_csv ('bh.csv', delimiter = '\t')
address = pd.read_csv ('addresses.csv', delimiter = '\t')
```

```
pysqldf ("select sum(n_txs) from block_hash")
```

```
pysqldf ("select count(addrID) from address")
```



```
In [17]: import pandas as pd
from pandasql import sqldf

In [18]: def pysqldf(q):
return sqldf(q,globals())

In [19]: import pandas as pd

block_hash = pd.read_csv ('bh.csv', delimiter = '\t')
address = pd.read_csv ('addresses.csv', delimiter = '\t')

In [20]: pysqldf ("select sum(n_txs) from block_hash")
Out[20]:
      sum(n_txs)
0    10000055

In [21]: pysqldf ("select count(addrID) from address")
Out[21]:
      count(addrID)
0         8385065
```

2. What is the Bitcoin address that is holding the greatest amount of bitcoins? How much is that exactly? Note that the address here must be a valid Bitcoin address string. To answer this, you need to calculate the balance of each address. The balance here is the total amount of bitcoins in the UTXOs of an address.

Address - 1933phfhK3ZgFQNLGSDXvqCn32k2buXY8a
Amount - 11111100000000

```
import pandas as pd
address = pd.read_csv('addresses.csv', delimiter = '\t')
transaction_in = pd.read_csv('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv('txout.csv', delimiter = '\t')
```

```
pysqldef ( "select sval, address FROM (select row_number() over (ORDER BY
(IFNULL(tout.sout,0)-IFNULL(tin.sin,0)) DESC) row_num, (IFNULL(tout.sout,0)-
IFNULL(tin.sin,0)) sval, ad.address, ad.addrid from address ad, (select
IFNULL(sum(sum),0) sout, addrid from transaction_out group by addrid) tout NATURAL
LEFT OUTER JOIN (select IFNULL(sum(sum),0) sin, addrid from transaction_in group
by addrid) tin where tout.addrid = ad.addrid) where row_num = 1" )
```

The screenshot shows a Jupyter Notebook interface with the following content:

```
In [1]: import pandas as pd
        from pandasql import sqldf

In [2]: def pysqldef(q):
        return sqldf(q,globals())

In [4]: import pandas as pd
        address = pd.read_csv('addresses.csv', delimiter = '\t')
        transaction_in = pd.read_csv('txin.csv', delimiter = '\t')
        transaction_out = pd.read_csv('txout.csv', delimiter = '\t')
```

Below the code cells, the output of the SQL query is displayed:

```
In [22]: pysqldef ( "select sval, address FROM (select row_number() over (ORDER BY (IFNULL(tout.sout,0)-
Out[22]:
```

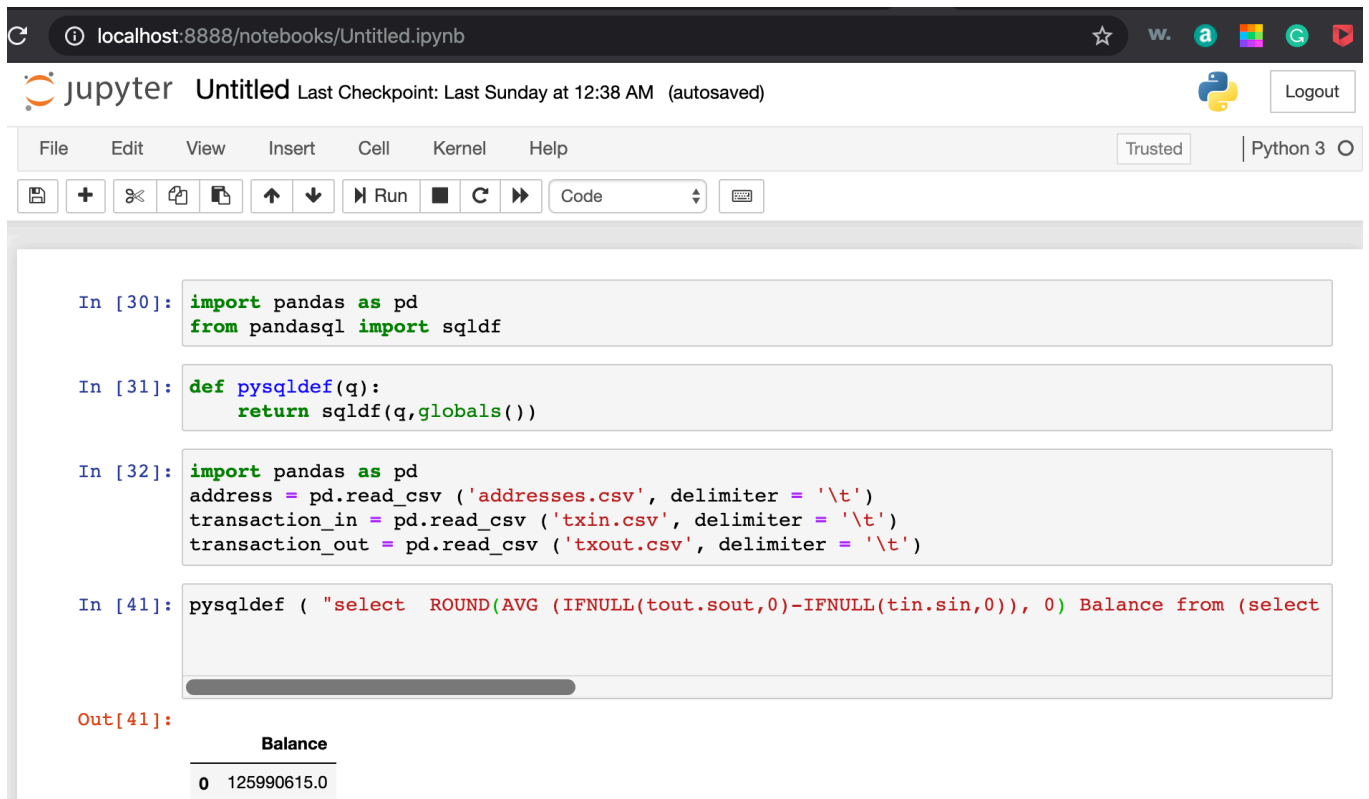
	sval	address
0	11111100000000	1933phfhK3ZgFQNLGSDXvqCn32k2buXY8a

3. What is the average balance per address?

Average Balance per address - 125990615.0

```
import pandas as pd
address = pd.read_csv ('addresses.csv', delimiter = '\t')
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```

```
pysqldef ( "select ROUND(AVG (IFNULL(tout.sout,0)-IFNULL(tin.sin,0)), 0) Balance from (select
IFNULL(sum(sum),0) sout, addrID from transaction_out group by addrID) tout NATURAL LEFT
OUTER JOIN (select IFNULL(sum(sum),0) sin, addrID from transaction_in group by addrID)
tin" )
```



The screenshot shows a Jupyter Notebook interface with the following content:

File Edit View Insert Cell Kernel Help Trusted Python 3

In [30]: `import pandas as pd`
`from pandasql import sqldf`

In [31]: `def pysqldef(q):`
 `return sqldf(q,globals())`

In [32]: `import pandas as pd`
`address = pd.read_csv ('addresses.csv', delimiter = '\t')`
`transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')`
`transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')`

In [41]: `pysqldef ("select ROUND(AVG (IFNULL(tout.sout,0)-IFNULL(tin.sin,0)), 0) Balance from (select`
`IFNULL(sum(sum),0) sout, addrID from transaction_out group by addrID) tout NATURAL LEFT`
`OUTER JOIN (select IFNULL(sum(sum),0) sin, addrID from transaction_in group by addrID)`
`tin")`

Out[41]:

	Balance
0	125990615.0

4. What is the average number of input and output transactions per address? What is the average number of transactions per address (including both inputs and outputs)? An output transaction of an address is the transaction that is originated from that address. Likewise, an input transaction of an address is the transaction that sends bitcoins to that address.

Average In - 2.17

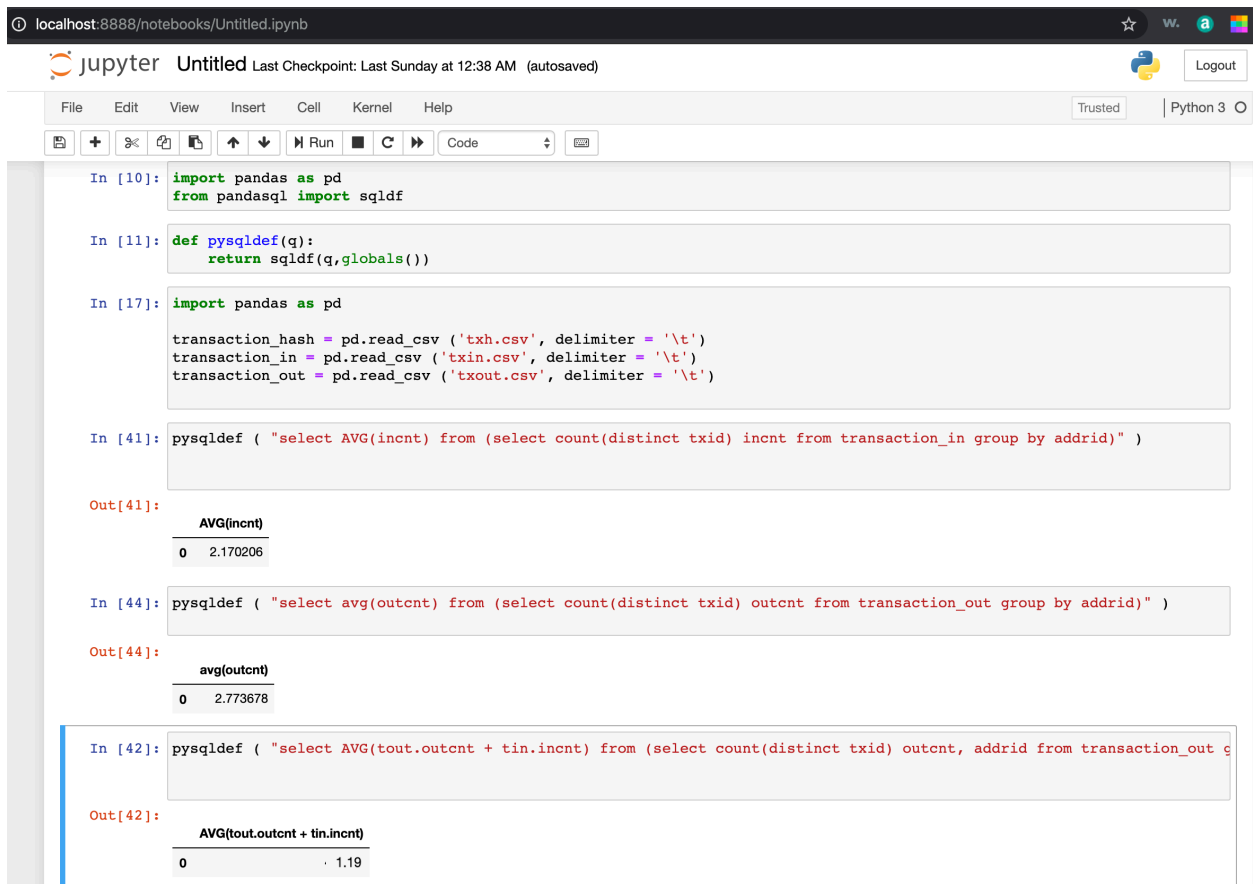
Average Out - 2.773678

Average Total - 1.19

```
pysqldef ( "select AVG(incnt) from (select count(distinct txid) incnt from transaction_in group by addrid)" )
```

```
pysqldef ( "select avg(outcnt) from (select count(distinct txid) outcnt from transaction_out group by addrid)" )
```

```
pysqldef ( "select AVG(tout.outcnt + tin.incnt) from (select count(distinct txid) outcnt, addrid from transaction_out group by addrid) tout, (select count(distinct txid) incnt , addrid from transaction_in group by addrid) tin where tout.addrid = tin.addrid" )
```



The screenshot shows a Jupyter Notebook interface with the following content:

```
In [10]: import pandas as pd
         from pandasql import sqldf

In [11]: def pysqldef(q):
         return sqldf(q,globals())

In [17]: import pandas as pd

         transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')
         transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
         transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')

In [41]: pysqldef ( "select AVG(incnt) from (select count(distinct txid) incnt from transaction_in group by addrid)" )

Out[41]:
      AVG(incnt)
0      2.170206

In [44]: pysqldef ( "select avg(outcnt) from (select count(distinct txid) outcnt from transaction_out group by addrid)" )

Out[44]:
      avg(outcnt)
0      2.773678

In [42]: pysqldef ( "select AVG(tout.outcnt + tin.incnt) from (select count(distinct txid) outcnt, addrid from transaction_out group by addrid) tout, (select count(distinct txid) incnt , addrid from transaction_in group by addrid) tin where tout.addrid = tin.addrid" )

Out[42]:
      AVG(tout.outcnt + tin.incnt)
0              1.19
```

5. What is the transaction that has the greatest number of inputs? How many inputs exactly? Show the hash of that transaction. If there are multiple transactions that have the same greatest number of inputs, show all of them.

Transaction Hash -

9621b3c67f9bddd3de65fafc488087b8f2b40b638e3a06209a904c66c0b32982

Number of transactions - 1312

```
import pandas as pd
```

```
transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')
```

```
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
```

```
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```

```
pysqldef ( " select txgrp.txcnt, txhash.hash FROM transaction_hash txhash, (select  
row_number() over (ORDER BY count(txid) DESC) row_num, count(txid) txcnt, txid from  
transaction_in group by txid) txgrp where txgrp.txid = txhash.txid and row_num = 1 ")
```

The screenshot shows a Jupyter Notebook interface with the following content:

- Toolbar:** Includes icons for file operations, code execution, and a dropdown menu set to 'Code'.
- Code Cells:**
 - In [10]:**

```
import pandas as pd  
from pandasql import sqldf
```
 - In [11]:**

```
def pysqldef(q):  
    return sqldf(q,globals())
```
 - In [14]:**

```
import pandas as pd  
  
transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')  
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')  
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```
 - In [16]:**

```
pysqldef ( " select txgrp.txcnt, txhash.hash FROM transaction_hash txhash, (select row_number()
```
- Output Cell:**
 - Out[16]:** A table with two columns: **txcnt** and **hash**.

	txcnt	hash
0	1312	9621b3c67f9bddd3de65fafc488087b8f2b40b638e3a06...

6. What is the average transaction value? Transaction value is the sum of all outputs' value.

Average Transaction Value - 12315588064.0

import pandas as pd

```
transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```

```
pysqldef ( " select substr((sout/cout) * 1.0, 1, instr((sout/cout) * 1.0, '.') + 1) avg from (select
sum(sum) sout,count(distinct txid) cout from transaction_out) " )
```

The screenshot shows a Jupyter Notebook interface with the following content:

- Browser Address Bar:** localhost:8888/notebooks/Untitled.ipynb#
- Jupyter Header:** jupyter Untitled Last Checkpoint: Last Sunday at 12:38 AM (unsaved changes) Logout
- Menu Bar:** File Edit View Insert Cell Kernel Help Trusted Python 3
- Toolbar:** Includes icons for file operations, cell navigation, and execution.
- Code Cells:**
 - In [10]:**

```
import pandas as pd
from pandasql import sqldf
```
 - In [11]:**

```
def pysqldef(q):
    return sqldf(q,globals())
```
 - In [17]:**

```
import pandas as pd

transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```
 - In [25]:**

```
pysqldef ( " select substr((sout/cout) * 1.0, 1, instr((sout/cout) * 1.0, '.') + 1) avg from (s
```
- Output [25]:** A table with one column 'avg' and one row containing the value 12315588064.0.

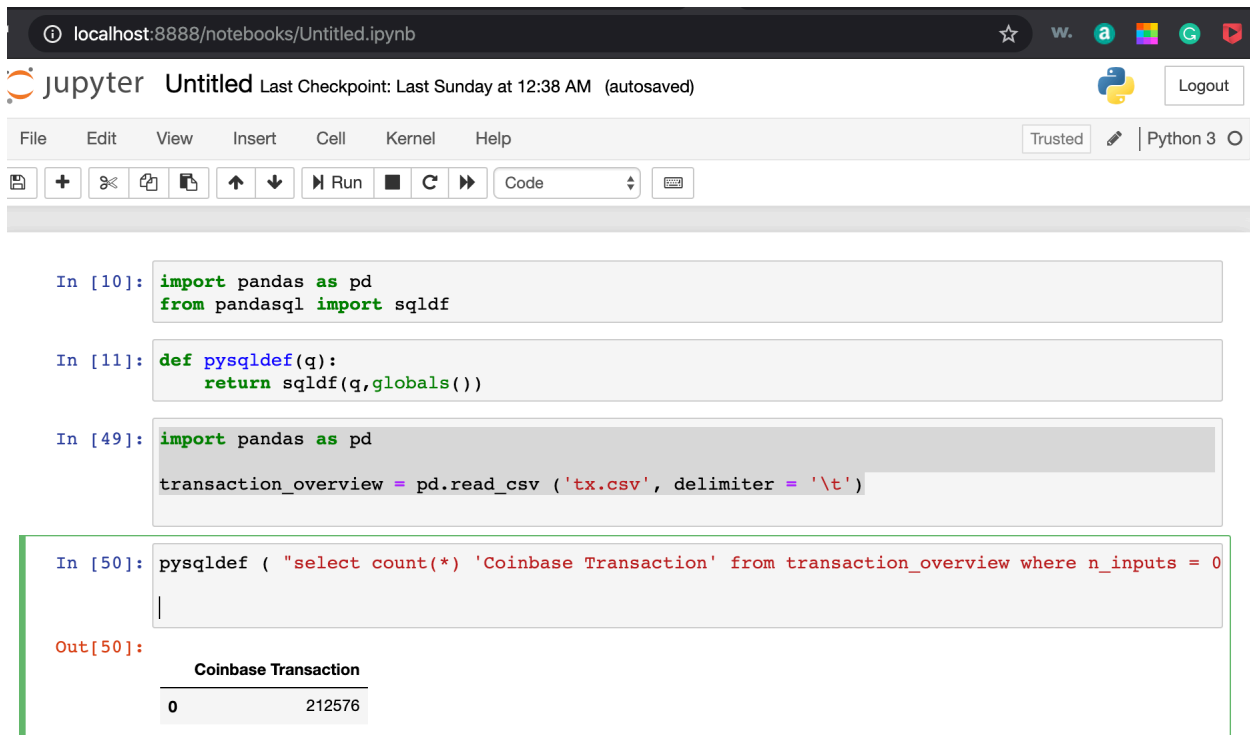
7. How many coinbase transactions are there in the dataset?

Number of Coinbase Transaction - 212576

```
import pandas as pd
```

```
transaction_overview = pd.read_csv('tx.csv', delimiter = '\t')
```

```
pysqldef ( "select count(*) 'Coinbase Transaction' from transaction_overview where n_inputs = 0" )
```



The screenshot shows a Jupyter Notebook interface with the following content:

```
In [10]: import pandas as pd
         from pandasql import sqldf

In [11]: def pysqldef(q):
         return sqldf(q,globals())

In [49]: import pandas as pd
         transaction_overview = pd.read_csv ('tx.csv', delimiter = '\t')

In [50]: pysqldef ( "select count(*) 'Coinbase Transaction' from transaction_overview where n_inputs = 0
|

Out[50]:
```

Coinbase Transaction	
0	212576

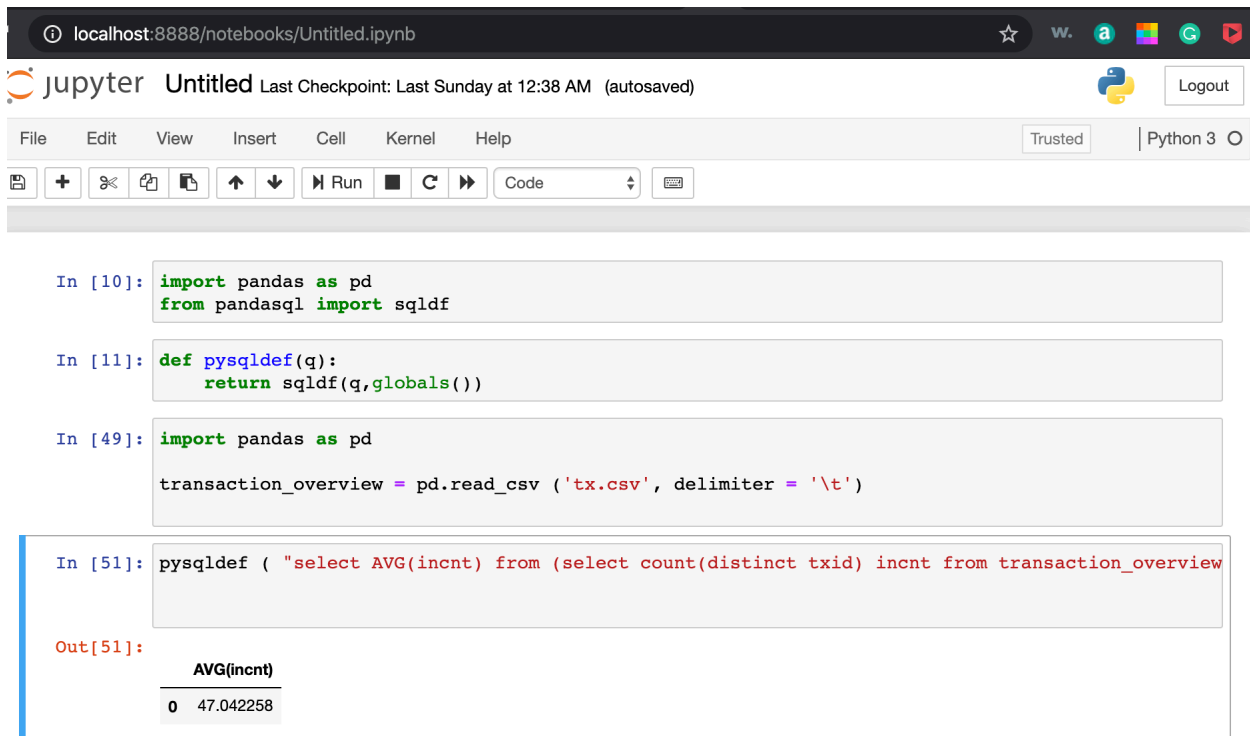
8. What is the average number of transactions per block?

Average Number of transactions per block - 47.042258

```
import pandas as pd
```

```
transaction_overview = pd.read_csv('tx.csv', delimiter = '\t')
```

```
pysqldef ( "select AVG(incnt) from (select count(distinct txid) incnt from transaction_overview  
group by blockid)" )
```



The screenshot shows a Jupyter Notebook interface with the following content:

- Browser address bar: localhost:8888/notebooks/Untitled.ipynb
- Jupyter logo and title: Untitled Last Checkpoint: Last Sunday at 12:38 AM (autosaved)
- Menu bar: File, Edit, View, Insert, Cell, Kernel, Help
- Trust status: Trusted | Python 3
- Code cells:
 - In [10]:

```
import pandas as pd  
from pandasql import sqldf
```
 - In [11]:

```
def pysqldef(q):  
    return sqldf(q,globals())
```
 - In [49]:

```
import pandas as pd  
  
transaction_overview = pd.read_csv ('tx.csv', delimiter = '\t')
```
 - In [51]:

```
pysqldef ( "select AVG(incnt) from (select count(distinct txid) incnt from transaction_overview
```
- Output cell for In [51]:

```
Out[51]:
```

	AVG(incnt)
0	47.042258

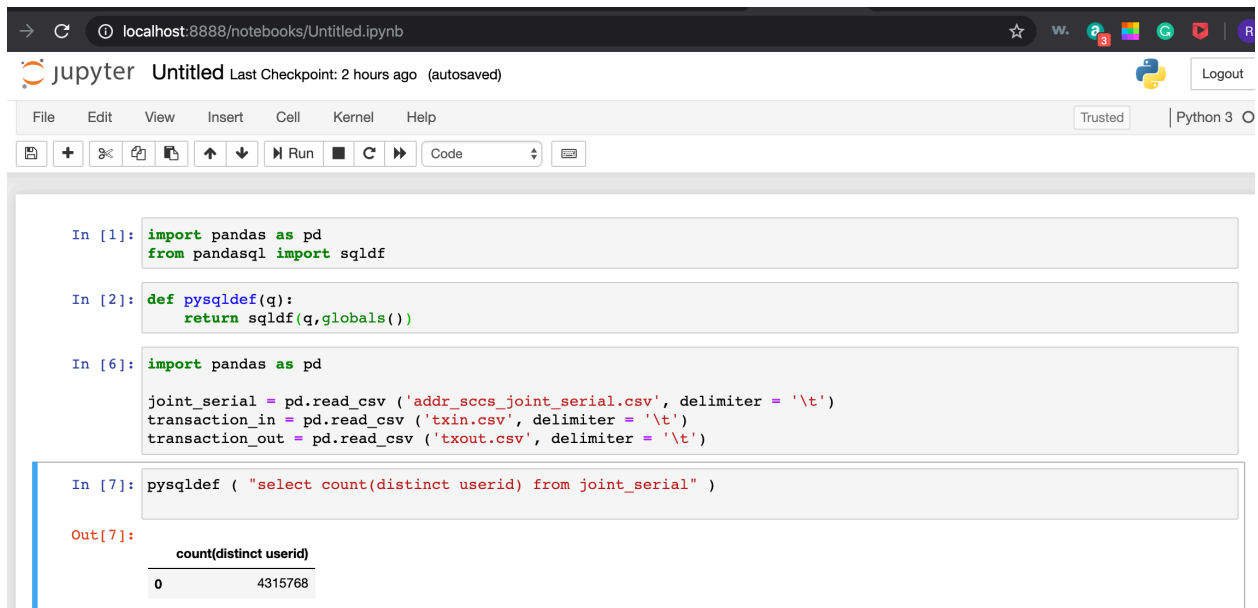
Part 2: Address de-anonymization

1. How many users are there in the dataset?

Number of distinct Users - 4315768

```
import pandas as pd
joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')

pysqldef ( "select count(distinct userid) from joint_serial" )
```



The screenshot shows a Jupyter Notebook interface with the following content:

```
In [1]: import pandas as pd
        from pandasql import sqldf

In [2]: def pysqldef(q):
        return sqldf(q,globals())

In [6]: import pandas as pd

        joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')
        transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
        transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')

In [7]: pysqldef ( "select count(distinct userid) from joint_serial" )

Out[7]:
```

	count(distinct userid)
0	4315768

2.1 What is the Bitcoin user that is holding the greatest amount of bit coins? How much is that exactly? Note that the address here must be a valid Bitcoin address string. To answer this, you need to calculate the balance of each address. The balance here is the total amount of bitcoins in the UTXOs of an address.

UserID - 12461805

Amount - 213940806047898

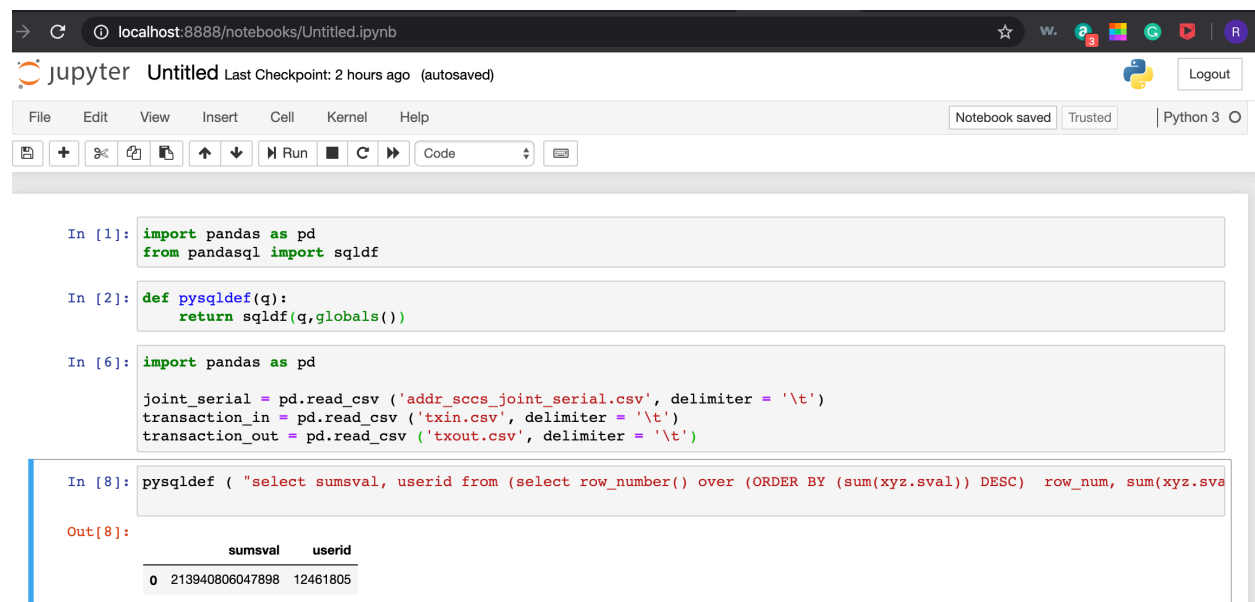
```
import pandas as pd
```

```
joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')
```

```
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
```

```
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```

```
pysqldef ( "select sum sval, userid from (select row_number() over (ORDER BY
(sum(xyz.sval)) DESC) row_num, sum(xyz.sval) sum sval, xyz.userid from (select
addr.addrId, addr.userid, abc.sval from joint_serial addr, (select (IFNULL(tout.sout,0)-
IFNULL(tin.sin,0)) sval, tout.addrId addrId from (select IFNULL(sum(sum),0) sout,
addrId from transaction_out group by addrId) tout NATURAL LEFT OUTER JOIN
(select IFNULL(sum(sum),0) sin, addrId from transaction_in group by addrId) tin ) abc
where addr.addrId = abc.addrId) xyz group by xyz.userid) where row_num = 1 " )
```



The screenshot shows a Jupyter Notebook interface with the following content:

```
In [1]: import pandas as pd
from pandasql import sqldf

In [2]: def pysqldef(q):
        return sqldf(q,globals())

In [6]: import pandas as pd

joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')

In [8]: pysqldef ( "select sum sval, userid from (select row_number() over (ORDER BY (sum(xyz.sval)) DESC) row_num, sum(xyz.sval) sum sval, xyz.userid from (select
addr.addrId, addr.userid, abc.sval from joint_serial addr, (select (IFNULL(tout.sout,0)-
IFNULL(tin.sin,0)) sval, tout.addrId addrId from (select IFNULL(sum(sum),0) sout,
addrId from transaction_out group by addrId) tout NATURAL LEFT OUTER JOIN
(select IFNULL(sum(sum),0) sin, addrId from transaction_in group by addrId) tin ) abc
where addr.addrId = abc.addrId) xyz group by xyz.userid) where row_num = 1 " )
```

Out[8]:

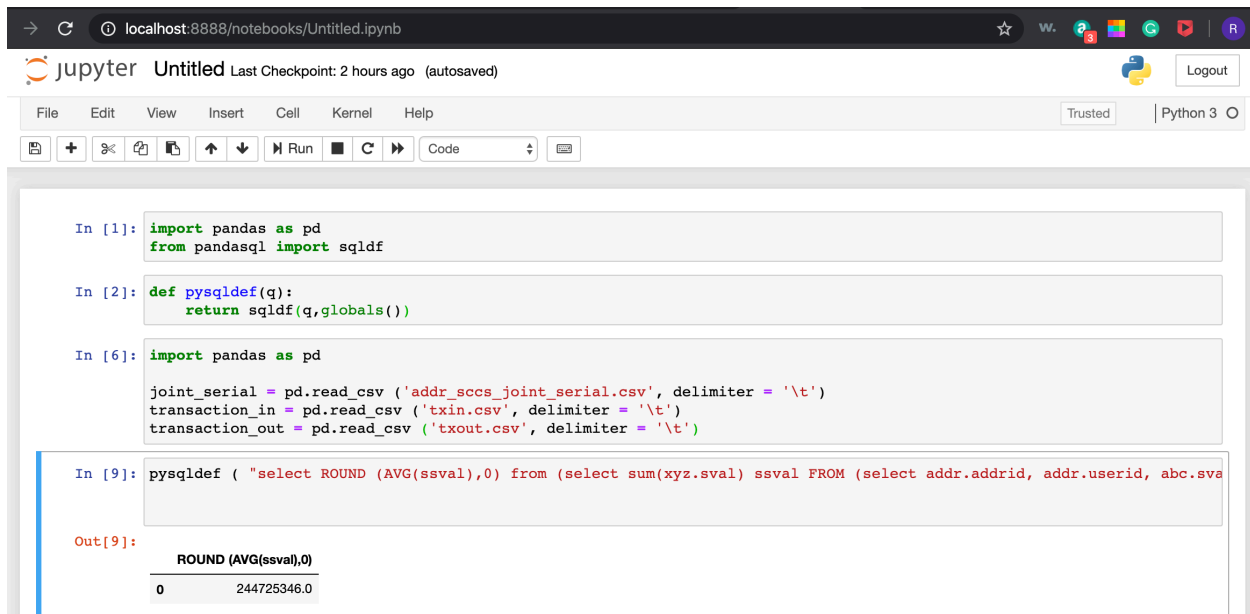
	sum sval	userid
0	213940806047898	12461805

2.2 What is the average balance per user?

Average Balance per user - 244725346.0

```
import pandas as pd
joint_serial = pd.read_csv ('joint_serial.csv', delimiter = '\t')
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```

```
pysqldef ( "select ROUND (AVG(ssval),0) from (select sum(xyz.sval) ssval FROM (select
addr.addrId, addr.userId, abc.sval from joint_serial addr, (select (IFNULL(tout.sout,0)-
IFNULL(tin.sin,0)) sval, tout.addrId addrId from (select IFNULL(sum(sum),0) sout,
addrId from transaction_out group by addrId) tout NATURAL LEFT OUTER JOIN
(select IFNULL(sum(sum),0) sin, addrId from transaction_in group by addrId) tin) abc
where addr.addrId = abc.addrId) xyz group by xyz.userId)" )
```



The screenshot shows a Jupyter Notebook interface with the following content:

- Code Cells:**
 - In [1]:** `import pandas as pd`
`from pandasql import sqldf`
 - In [2]:** `def pysqldef(q):`
 `return sqldf(q,globals())`
 - In [6]:** `import pandas as pd`
`joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')`
`transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')`
`transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')`
 - In [9]:** `pysqldef ("select ROUND (AVG(ssval),0) from (select sum(xyz.sval) ssval FROM (select addr.addrId, addr.userId, abc.sval`
- Output:**
Out[9]:

	ROUND (AVG(ssval),0)
0	244725346.0

2.3 What is the average number of input and output transactions per user? What is the average number of transactions per user (including both inputs and outputs)? An output transaction of user is the transaction that is originated from that user. Likewise, an input transaction of user is the transaction that sends bitcoins to that user.

Average In - 2.170198

Average Out - 2.773547

Average Total - 4.979047

```
import pandas as pd
```

```
joint_serial = pd.read_csv('joint_serial.csv', delimiter = '\t')
```

```
transaction_in = pd.read_csv('txin.csv', delimiter = '\t')
```

```
transaction_out = pd.read_csv('txout.csv', delimiter = '\t')
```

```
pysqldf ( "select avg(abc.incnt) from joint_serial addr, (select count(distinct tin.txid) incnt ,tin.addridd from transaction_in tin group by tin.addridd) abc where abc.addridd = addr.addridd" )
```

```
pysqldf ( "select avg(abc.outcnt) from joint_serial addr, (select count(distinct tout.txid) outcnt ,tout.addridd from transaction_out tout group by tout.addridd) abc where abc.addridd = addr.addridd" )
```

```
pysqldf ( "select avg(abc.outcnt + abc.incnt) from joint_serial addr, ((select count(distinct tout.txid) outcnt ,tout.addridd from transaction_out tout group by tout.addridd) NATURAL LEFT OUTER JOIN (select count(distinct tin.txid) incnt ,tin.addridd from transaction_in tin group by tin.addridd)) abc where abc.addridd = addr.addridd" )
```

```
In [1]: import pandas as pd
from pandasql import sqldf

In [2]: def pysqldf(q):
return sqldf(q,globals())

In [6]: import pandas as pd

joint_serial = pd.read_csv('addr_sccs_joint_serial.csv', delimiter = '\t')
transaction_in = pd.read_csv('txin.csv', delimiter = '\t')
transaction_out = pd.read_csv('txout.csv', delimiter = '\t')

In [13]: pysqldf ( "select avg(abc.incnt) from joint_serial addr, (select count(distinct tin.txid) incnt ,tin.addridd from trans

Out[13]:
      avg(abc.incnt)
0      2.170198

In [14]: pysqldf ( "select avg(abc.outcnt) from joint_serial addr, (select count(distinct tout.txid) outcnt ,tout.addridd from t

Out[14]:
      avg(abc.outcnt)
0      2.773547

In [17]: pysqldf ( "select avg(abc.outcnt + abc.incnt) from joint_serial addr, ((select count(distinct tout.txid) outcnt ,tout.

Out[17]:
      avg(abc.outcnt + abc.incnt)
0      4.979047
```

3. Give the hash of the transaction sending the greatest number of bitcoins to the user who is holding the greatest balance.

Transaction Hash -

c246c27e7bacc667d27ace253abf2bba82aa1e5fcd1d73e1b85863f6b890e1bf

```
import pandas as pd
```

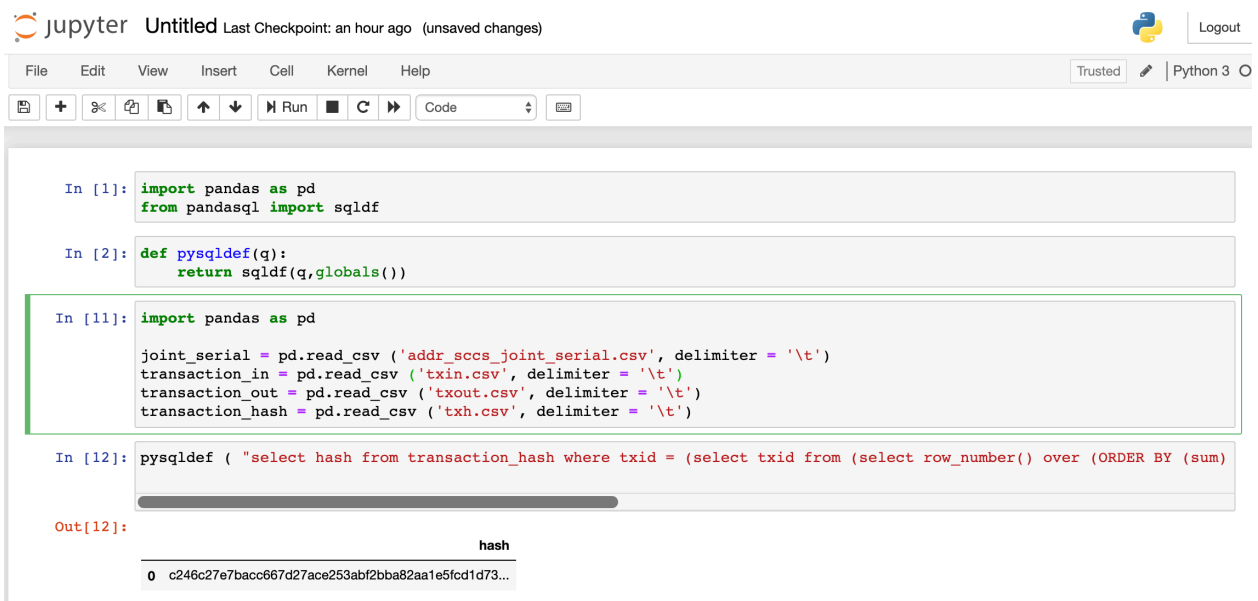
```
joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')
```

```
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')
```

```
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')
```

```
transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')
```

```
pysqldef ( "select hash from transaction_hash where txid = (select txid from (select  
row_number() over (ORDER BY (sum) DESC) row_num, txid, sum, addrid from transaction_in  
where addrid IN (select addrid from joint_serial where userid = 12461805)) where row_num =  
1)" )
```



```
In [1]: import pandas as pd  
from pandasql import sqldf  
  
In [2]: def pysqldef(q):  
        return sqldf(q,globals())  
  
In [11]: import pandas as pd  
  
joint_serial = pd.read_csv ('addr_sccs_joint_serial.csv', delimiter = '\t')  
transaction_in = pd.read_csv ('txin.csv', delimiter = '\t')  
transaction_out = pd.read_csv ('txout.csv', delimiter = '\t')  
transaction_hash = pd.read_csv ('txh.csv', delimiter = '\t')  
  
In [12]: pysqldef ( "select hash from transaction_hash where txid = (select txid from (select row_number() over (ORDER BY (sum)  
row_number() over (ORDER BY (sum) DESC) row_num, txid, sum, addrid from transaction_in  
where addrid IN (select addrid from joint_serial where userid = 12461805)) where row_num =  
1)" )  
  
Out[12]:
```

	hash
0	c246c27e7bacc667d27ace253abf2bba82aa1e5fcd1d73...