SQL Queries

Commands used for table creation:

- *create table addresses (ADDR_ID int, ADDRESS varchar(100));*
- *create table tx(TX_ID int, BLOCK_ID int, N_INPUTS int, N_OUTPUTS int);*
- *create table txin(TX_ID int, INPUT_SEQ int, PREVIOUS_TX_ID int, PREV_OUTPUT_SEQ int, ADDR_ID int, SUM_IN bigint);*
- *create table txout(TX_ID int, OUTPUT_SEQ int, ADDR_ID int, SUM_OUT bigint);*
- *create table multiple(TX_ID int, OUTPUT_SEQ int, ADDR_ID int);*
- *create table nonstandard(TX_ID int, OUTPUT_SEQ int);*
- *create table blockhash (BLOCK_ID int, HASH_VALUE varchar(100), BLOCK_TIMESTAMP varchar(100), N_TXS int);*
- *create table txh (TX_ID int, HASH_VALUE varchar(100));*
- *create table addr_sccs(ADDR_ID int, USER_IDG int);*
- *create table addrSccsSerial(ADDR_ID int, USER_IDG int);*

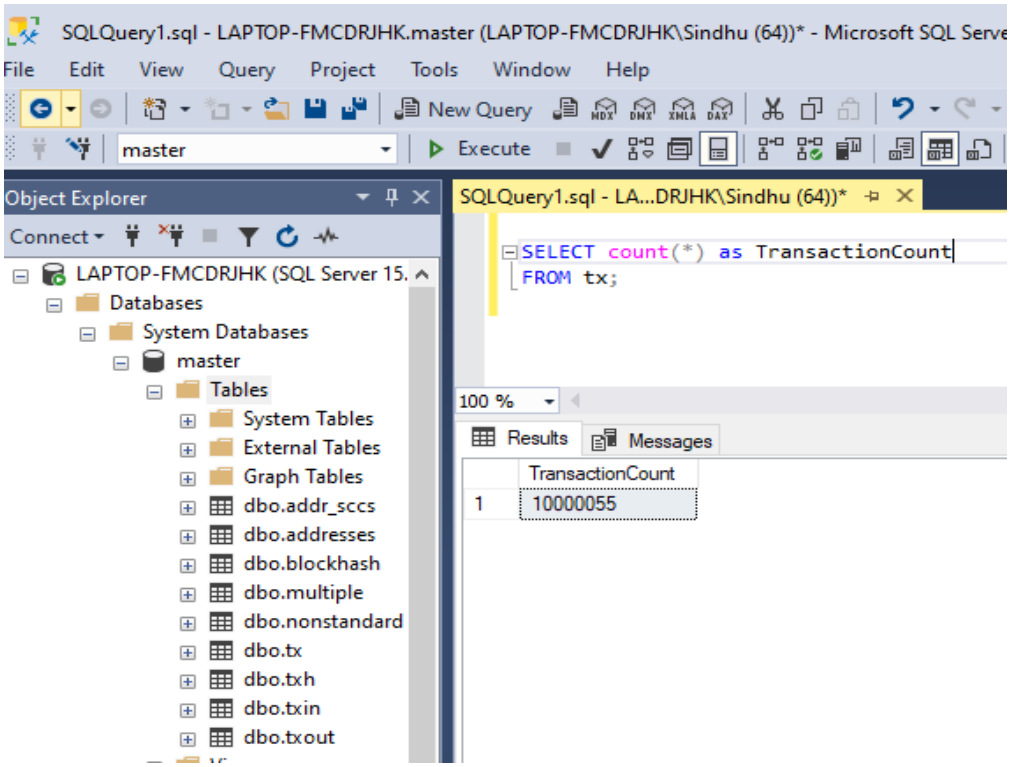
I. PART-1 Solutions

➢ **Question 1:**

Number of Transactions in the dataset: 10000055
Number of Addresses in the dataset: 8385065

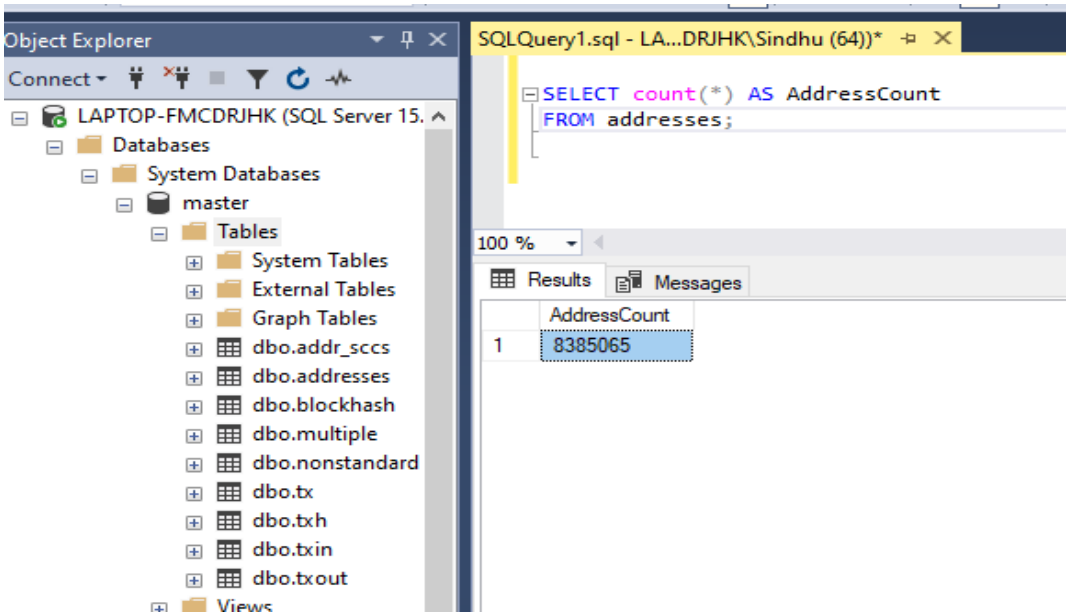
SQL Query:
```
SELECT count(*) as TransactionCount FROM tx;
```

Screenshot:



SQL Query: `SELECT count(*) AS AddressCount FROM addresses;`

Screenshot:



- **Question 2:**

What is the Bitcoin address that is holding the greatest amount of bitcoins - 1083442

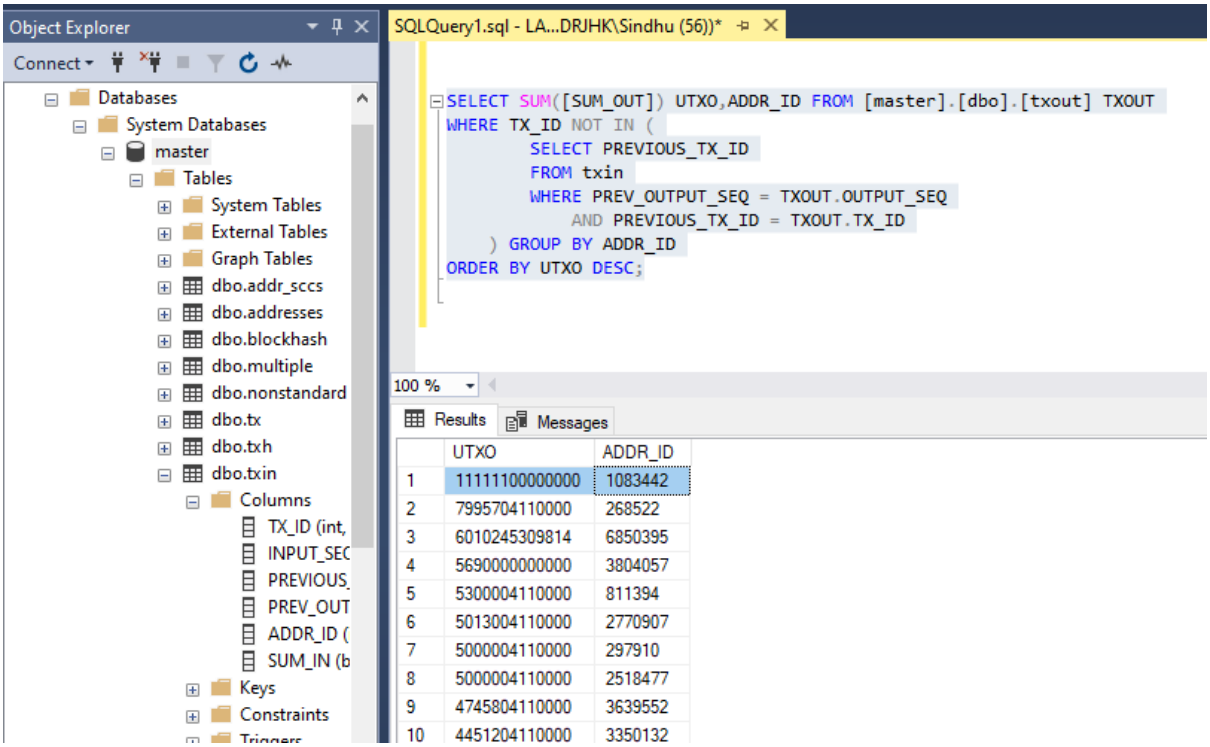
How much is that exactly – 11111100000000satoshi

SQL Query:

```
SELECT SUM([SUM_OUT]) UTXO,ADDR_ID FROM [master].[dbo].[txout] TXOUT WHERE
TX_ID NOT IN (
        SELECT PREVIOUS_TX_ID
        FROM txin
        WHERE PREV_OUTPUT_SEQ = TXOUT.OUTPUT_SEQ
                AND PREVIOUS_TX_ID = TXOUT.TX_ID
```

```
) GROUP BY ADDR_ID
ORDER BY UTXO DESC;
```

Screenshots:



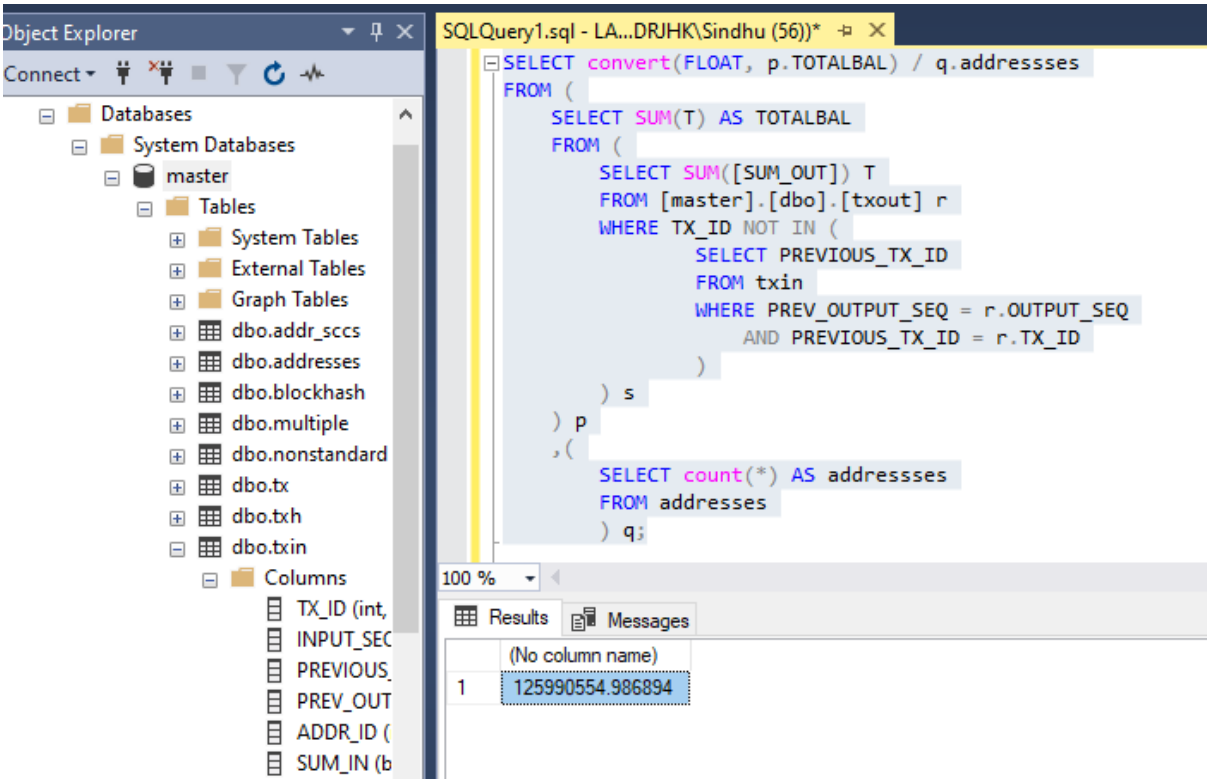
- **Question 3:**

What is the average balance per address – 125990554.986894satoshi

SQL Query:

```
SELECT convert(FLOAT, p.TOTALBAL) / q.addressses FROM (
SELECT SUM(T) AS TOTALBAL
FROM (
	SELECT SUM([SUM_OUT]) T
	FROM [master].[dbo].[txout] r
	WHERE TX_ID NOT IN (
			SELECT PREVIOUS_TX_ID
			FROM txin
			WHERE PREV_OUTPUT_SEQ = r.OUTPUT_SEQ
				AND PREVIOUS_TX_ID = r.TX_ID
			)
	) s
) p
,(
	SELECT count(*) AS addresses FROM addresses
		) q;
```

Screenshots:



- **Question 4:**

What is the average number of input and output transactions per address?
Input avg - 2.77479148939215
Output Avg - 2.4052948903795
What is the average number of transactions per address (including both inputs and outputs)?

SQL Queries:

```
SELECT convert(FLOAT, p.outTrans + q.inTrans) / r.addresses AS AvgTrans FROM ( SELECT count(*) outTrans FROM [master].[dbo].[txout] ) p,( SELECT count(*) inTrans FROM [master].[dbo].[txin] ) q,( SELECT count(*) addresses FROM [master].[dbo].[addresses] ) r;
```

Screenshots:

SQLQuery1.sql - LA...DRJHK\Sindhu (56))*

```
SELECT convert(FLOAT, p.outTrans) / q.addresses AS inputTransactionAvg
FROM (
    SELECT count(*) outTrans
    FROM [master].[dbo].[txout]
    ) p
    ,(
        SELECT count(*) addresses
        FROM [master].[dbo].[addresses]
        ) q;
```

100 %

Results | Messages

	inputTransactionAvg
1	2.77479148939215

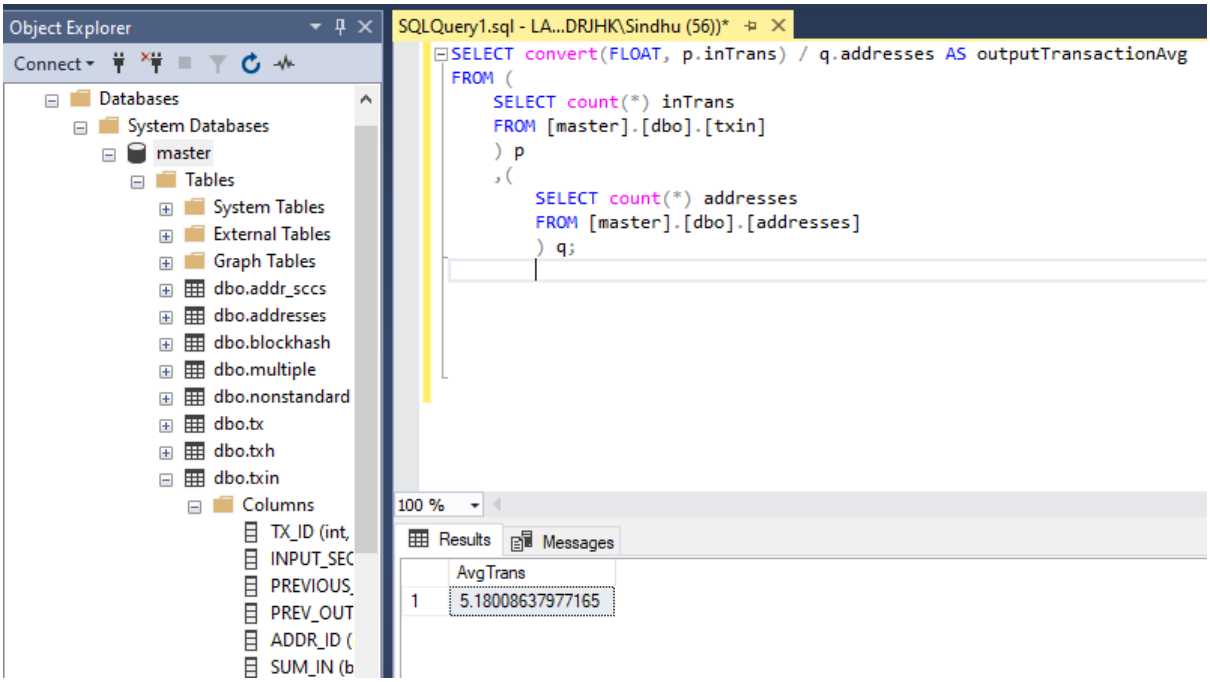
SQLQuery1.sql - LA...DRJHK\Sindhu (56))*

```
SELECT convert(FLOAT, p.inTrans) / q.addresses AS outputTransactionAvg
FROM (
    SELECT count(*) inTrans
    FROM [master].[dbo].[txin]
    ) p
    ,(
        SELECT count(*) addresses
        FROM [master].[dbo].[addresses]
        ) q;
```

100 %

Results | Messages

	outputTransactionAvg
1	2.4052948903795



➢ Question 5:

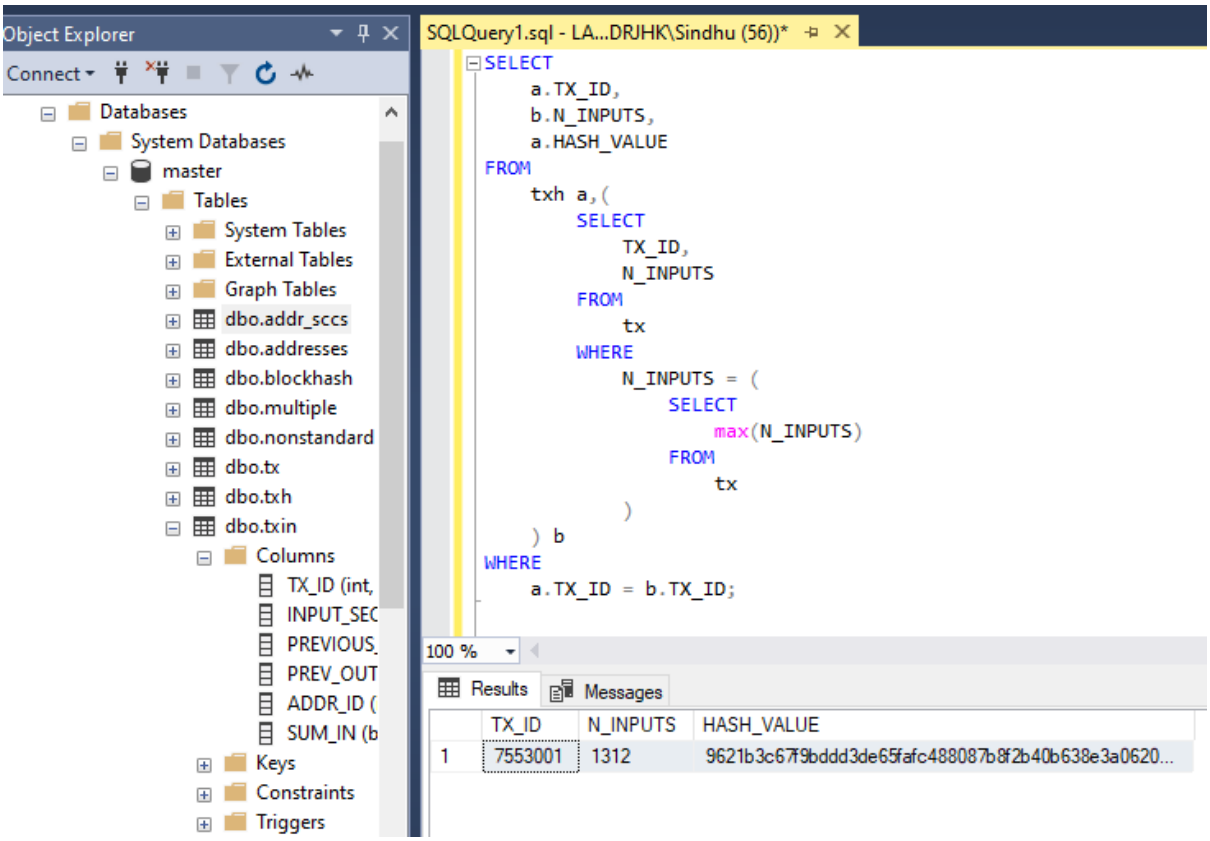
What is the transaction that has the greatest number of inputs? How many inputs exactly? Show the hash of that transaction. If there are multiple transactions that have the same greatest number of inputs, show all of them.

Transaction ID: 7553001
Number of inputs: 1312
Hash value: 9621b3c67f9bddd3de65fafc488087b8f2b40b638e3a06209a904c66c0b32982

SQL Query:

```
SELECT
	a.TX_ID,
	b.N_INPUTS,
	a.HASH_VALUE
FROM
	txh a,(
		SELECT
			TX_ID,
			N_INPUTS
		FROM
			tx
		WHERE
			N_INPUTS = (
				SELECT
					max(N_INPUTS)
				FROM
					tx
			)
	) b
WHERE
	a.TX_ID = b.TX_ID;
```

Screenshots:



- **Question 6:**

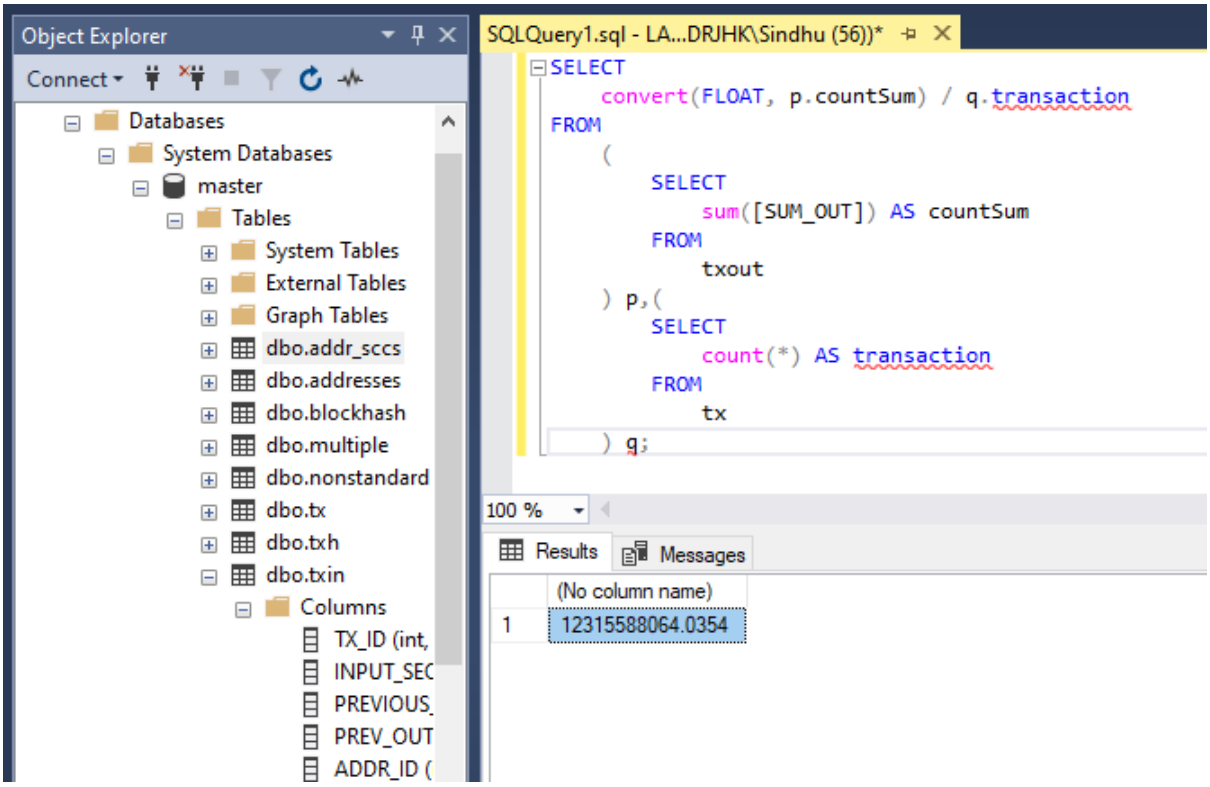
What is the average transaction value?

12315588064.0354

SQL Query:

```
SELECT
	convert(FLOAT, p.countSum) / q.transaction
FROM
	(
		SELECT
			sum([SUM_OUT]) AS countSum
		FROM
			txout
	) p,(
		SELECT
			count(*) AS transaction
		FROM
			tx
	) q;
```

Screenshots:



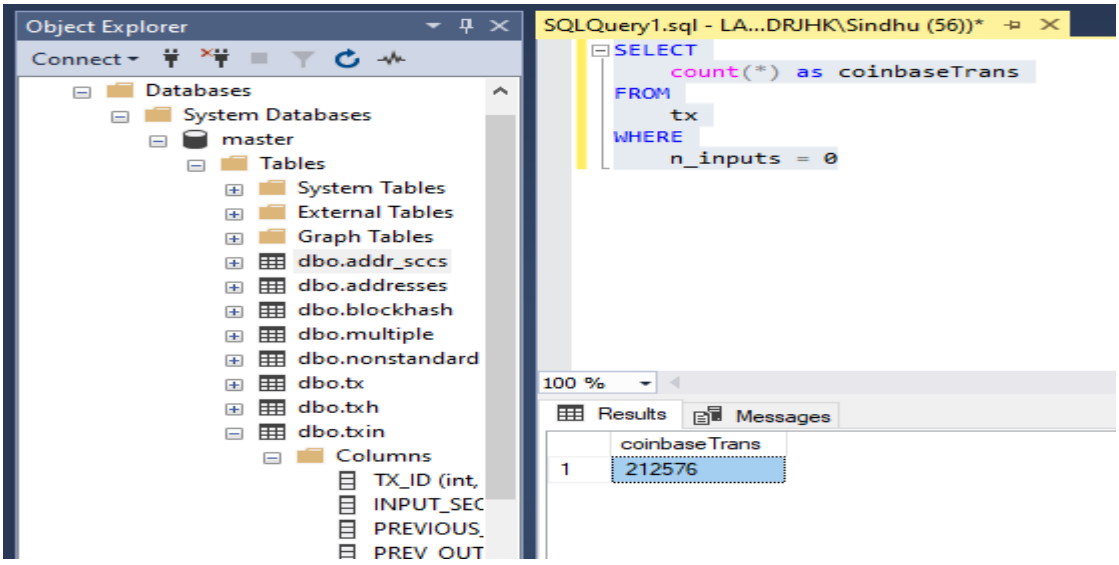
➢ Question 7:

How many coinbase transactions are there in the dataset?
212576

SQL Query:

```
SELECT
	count(*) as coinbaseTrans
FROM
	tx
WHERE
	n_inputs = 0
```

Screenshots:



➢ Question 8:

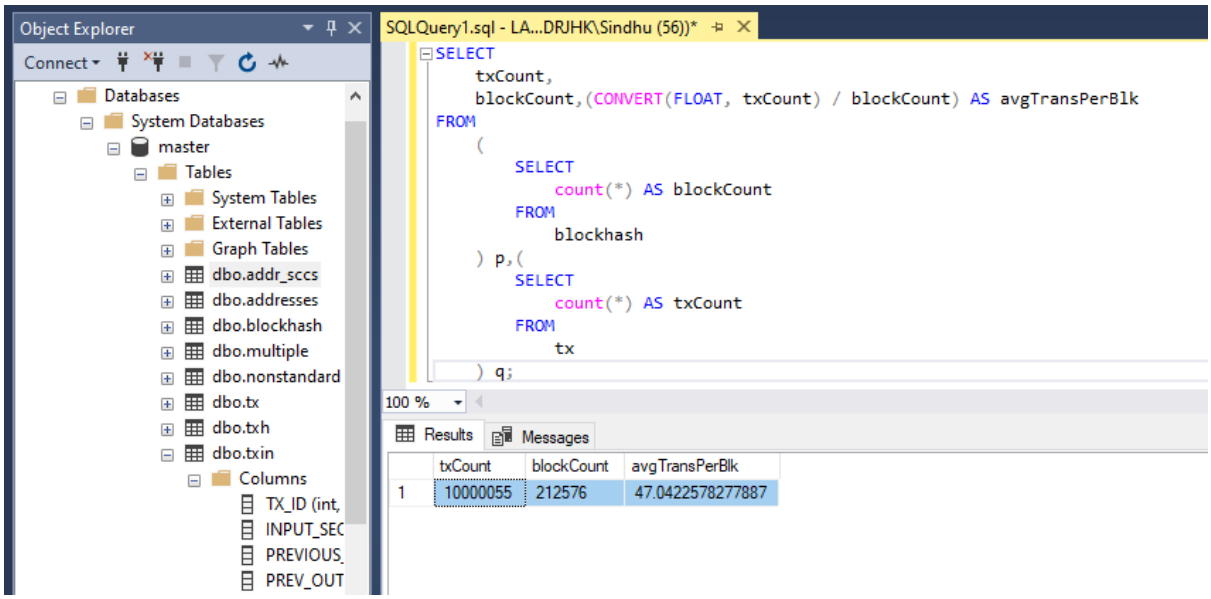
What is the average number of transactions per block?

txCount	**10000055**
blockCount	**212576**
avgTransPerBlk	**47.0422578277887**

SQL Query:

```
SELECT
	txCount,
	blockCount,(CONVERT(FLOAT, txCount) / blockCount) AS avgTransPerBlk
FROM
	(
		SELECT
			count(*) AS blockCount
		FROM
			blockhash
	) p,(
		SELECT
			count(*) AS txCount
		FROM
			tx
	) q;
```

Screenshots:



II. *PART-2 Solutions*

➢ Question 1

How many users are there in the dataset?

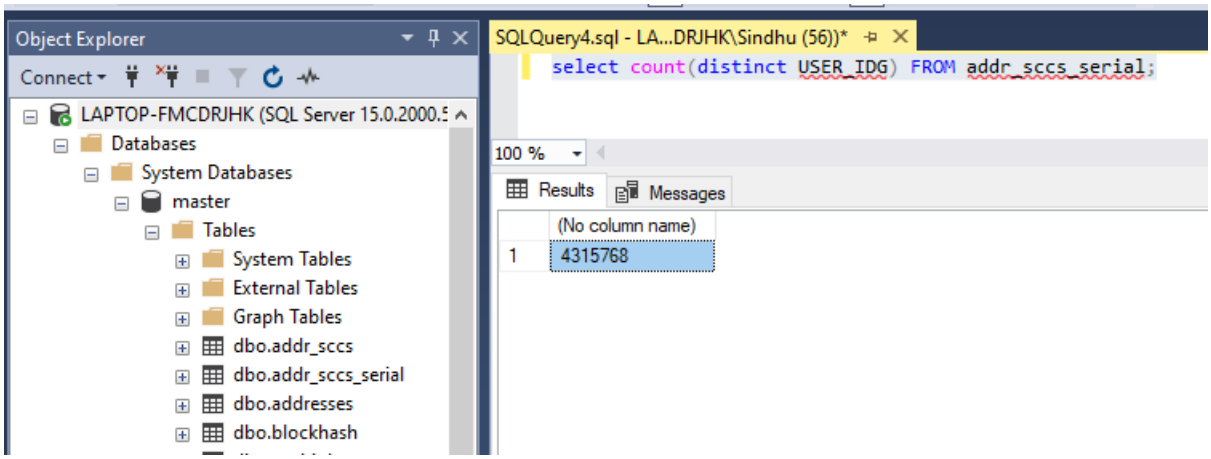
Distinct user IDs – 4315768

Distinct AddrsIDs -

SQL Query:

```
select
	count(distinct USER_IDG)
FROM
	addr_sccs_serial;
```

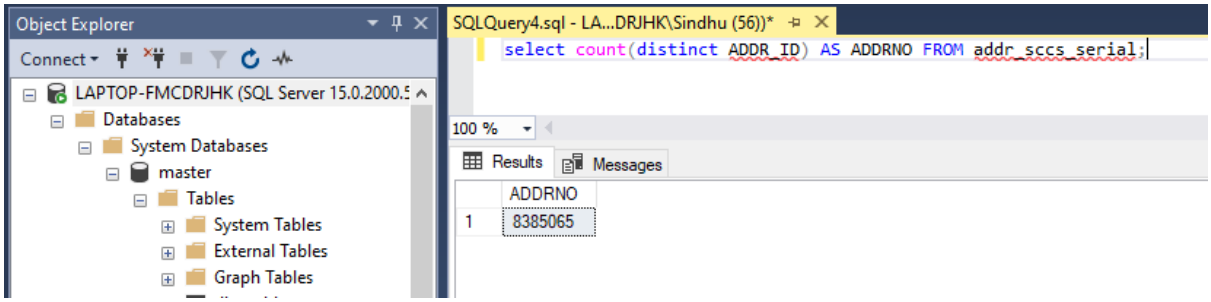
Screenshots:



SQL Query:

```
select
	count(distinct ADDR_ID)
FROM
	addr_sccs_serial;
```

Screenshot:



- **<u>Question 2.1:</u>**

What is the Bitcoin user that is holding the greatest amount of bitcoins?

12461805

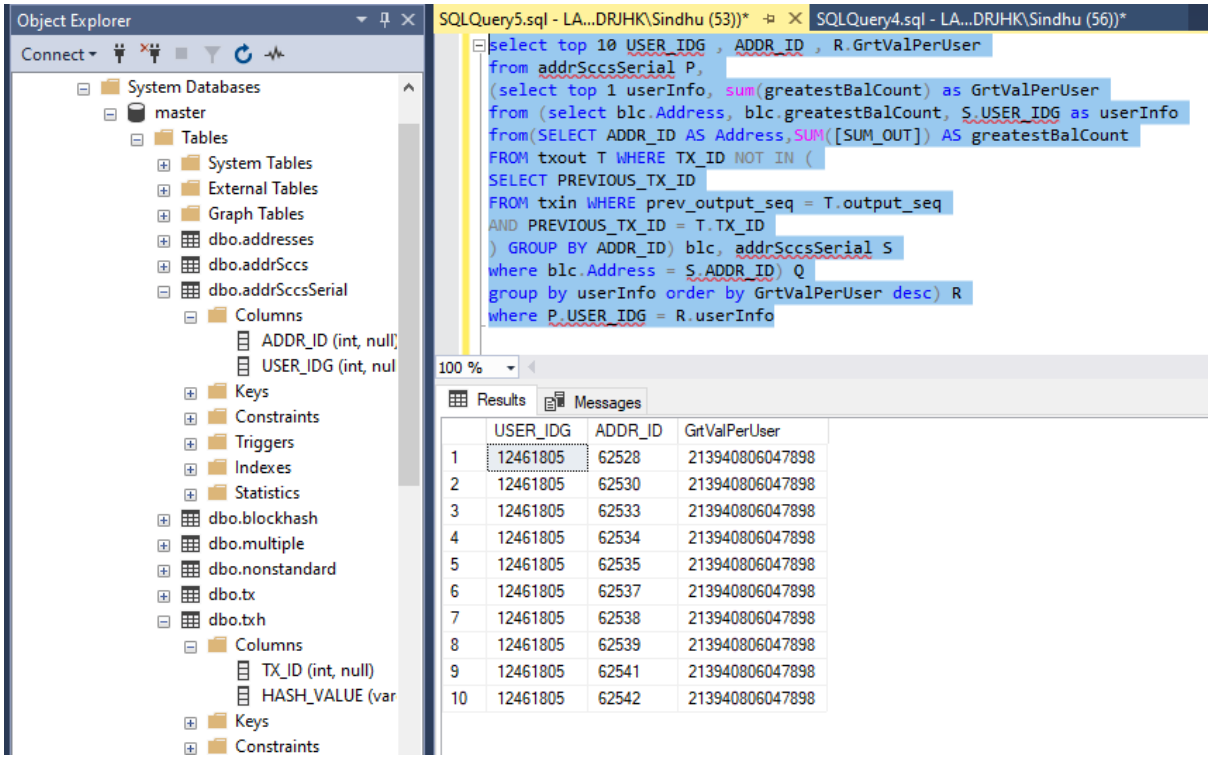
How much is that exactly?

2645440

SQL Query:

```
select top 10 USER_IDG , ADDR_ID , R.GrtValPerUser
from addrSccsSerial P,
(select top 1 userInfo, sum(greatestBalCount) as GrtValPerUser
from (select blc.Address, blc.greatestBalCount, S.USER_IDG as userInfo
```

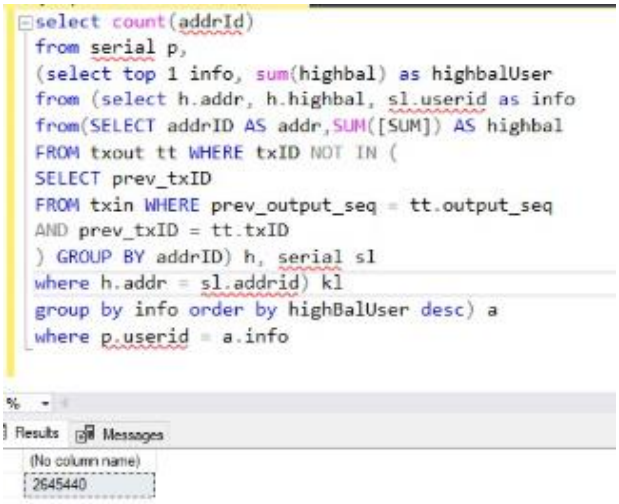
```
from(SELECT ADDR_ID AS Address,SUM([SUM_OUT]) AS greatestBalCount
FROM txout T WHERE TX_ID NOT IN (
SELECT PREVIOUS_TX_ID
FROM txin WHERE prev_output_seq = T.output_seq
AND PREVIOUS_TX_ID = T.TX_ID
) GROUP BY ADDR_ID) blc, addrSccsSerial S
where blc.Address = S.ADDR_ID) Q
group by userInfo order by GrtValPerUser desc) R
       where P.USER_IDG = R.userInfo
```



SQL Query:

```
select count(addrId)
from serial ss,
(select top 1 userInfo, sum(highestBalanceCount) as HighestBalPerUser
from (select blc.Address, blc.highestBalanceCount, srl.userid as userInfo
from(SELECT addrID AS Address,SUM([SUM]) AS highestBalanceCount
FROM txout tt WHERE txID NOT IN (
SELECT prev_txID
FROM txin WHERE prev_output_seq = tt.output_seq
AND prev_txID = tt.txID
) GROUP BY addrID) blc, serial srl
where blc.Address = srl.addrid) kl
group by userInfo order by HighestBalPerUser desc) bg
where ss.userid = bg.userInfo
```

Screenshot:



- **Question 2.2:**

What is the average balance per address?

SQL Query:

```
SELECT convert(FLOAT, t1.total) / t2.username
FROM ( SELECT SUM(T) AS total
FROM (SELECT SUM([SUM_OUT]) T
FROM txout e
WHERE TX_ID NOT IN (
SELECT PREVIOUS_TX_ID FROM txin
WHERE prev_output_seq = e.output_seq
AND PREVIOUS_TX_ID = e.TX_ID)) t11) t1
,(SELECT count(distinct USER_IDG) as username
       from addrSccsSerial) t2;
```

Screenshots:

- **Question 2.3:**

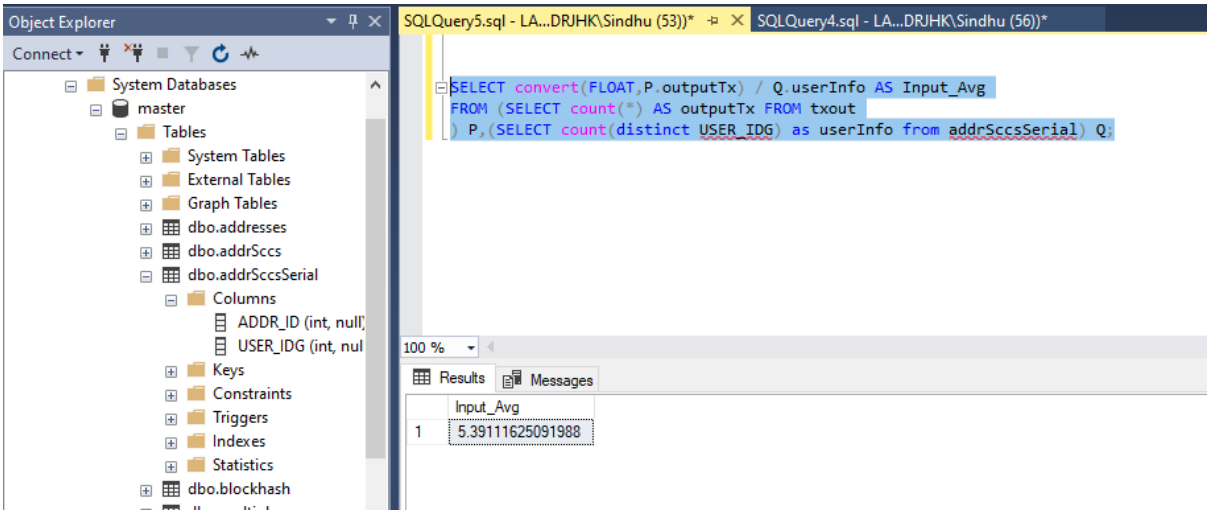
What is the average number of input and output transactions per address?
What is the average number of transactions per address

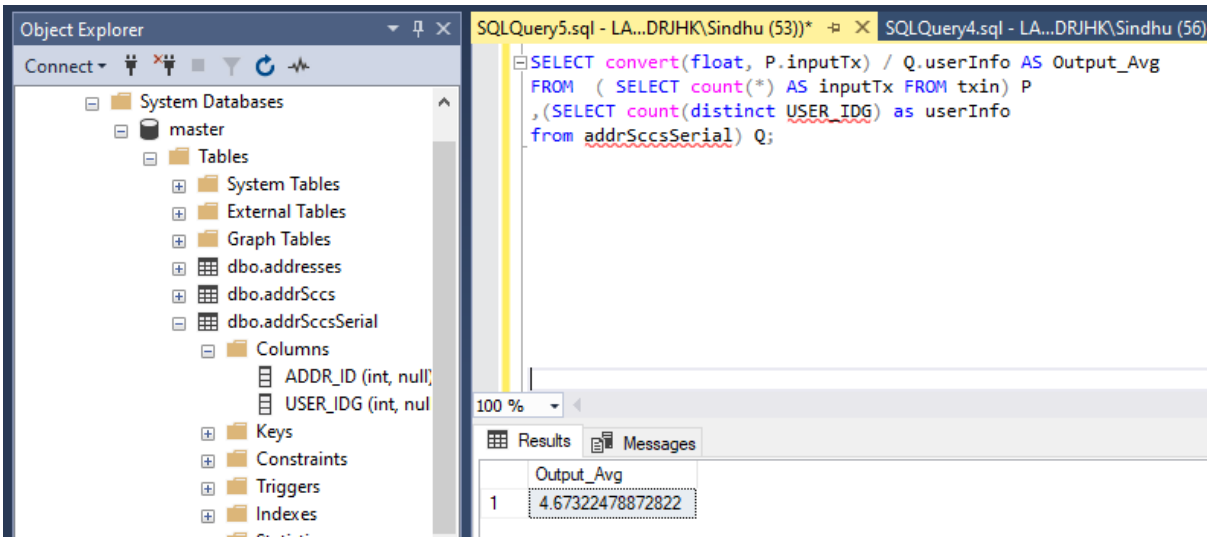
5.39111625091988
4.67322478872822
10.0643410396481

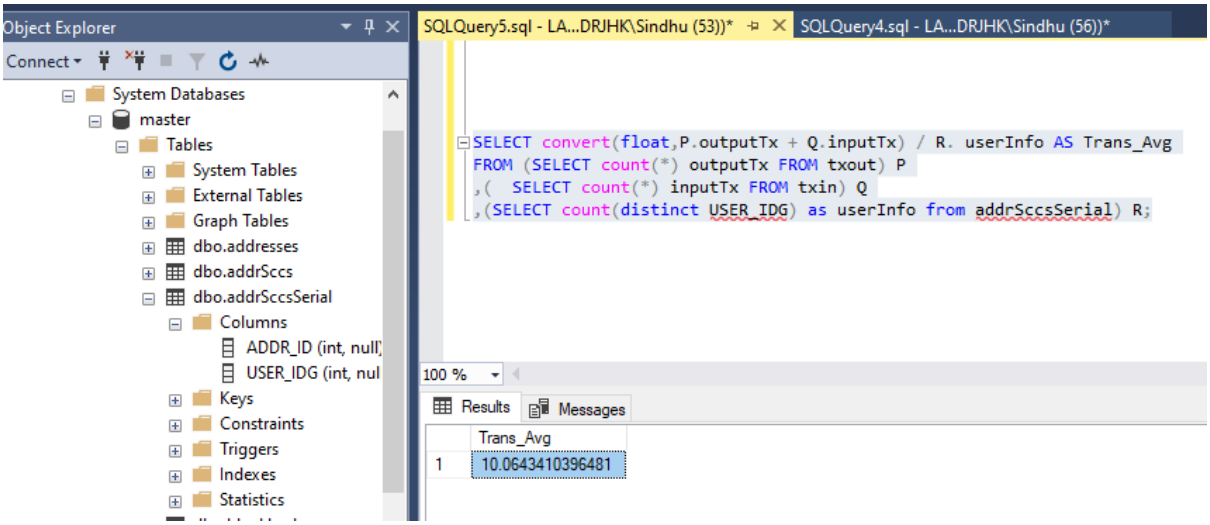
SQL Query:

```
SELECT convert(FLOAT,P.outputTx) / Q.userInfo AS Input_Avg
FROM (SELECT count(*) AS outputTx FROM txout
       ) P,(SELECT count(distinct USER_IDG) as userInfo from addrSccsSerial) Q;
```

Screenshots:







- **Question 3:**

Give the hash of the transaction sending the greatest number of bitcoins

to the user who is holding the greatest balance.

c246c27e7bacc667d27ace253abf2bba82aa1e5fcd1d73e1b85863f6b890e1bf

SQL Query:

```
with P  as (select userInfo, sum(greatestBal) as balncCount
from ( select blc.Address, blc.greatestBal, sU.USER_IDG as userInfo
from ( SELECT ADDR_ID AS Address,SUM([SUM_OUT]) AS greatestBal
FROM txout ttout WHERE TX_ID NOT IN (SELECT PREVIOUS_TX_ID FROM txin
WHERE prev_output_seq = ttout.output_seq AND PREVIOUS_TX_ID = ttout.TX_ID)
GROUP BY ADDR_ID
) blc, addrSccsSerial sU
where blc.Address = sU.ADDR_ID) S
group by userInfo)
select top 1 HASH_VALUE from
(select * from
(select top 1 userInfo from P order by balncCount desc) c join addrSccsSerial
op on c.userInfo = op.USER_IDG)R
join txin Q on Q.ADDR_ID = R.ADDR_ID join txh T on Q.TX_ID = T.TX_ID
order by SUM_IN desc ;
```

Screenshots: