Ethereum Data Analytics: Exploring the Ethereum Blockchain

Analytical Study

Pushpit Bhardwaj, Yuvraj Chandra, Deepesh Sagar

Department of Computer Science, Shaheed Sukhdev College of Business Studies, University of Delhi
18 September 2021

Abstract

With over $385 billion in market capitalization (as of 31st August, 2021), Ethereum is the largest public blockchain that supports smart contracts. It is also the second most valuable cryptocurrency, just after Bitcoin.

Ethereum has attracted the interest of many cryptocurrency fans, as well as investors, because it is not only a blockchain solution for the digital store of transaction records, but also offers the possibility of creating and executing smart contracts. Smart contracts are applications that are programmed as a digital agreement, with almost zero censorship, fraud, third party interference, or risk of going offline. As the contracts are on the blockchain, they become immutable making them attractive for various decentralised applications (or dApps) like e-governance, healthcare management and data provenance. However, the biggest advantage of smart contracts - their immutability also poses the biggest threat from a security standpoint. This is because any bug found in the smart contract after deployment cannot be patched.

Ethereum, and cryptocurrencies in general, have been in the center of criticism regarding their volatility and the speculation in their prices. In this study, an in-depth big data analysis on the Ethereum blockchain is performed, revealing interesting trends. The main focus is on understanding what the number of transactions in the Ethereum network represents and what the volume of ether in circulation between the different addresses is. For that purpose, the Google BigQuery Ethereum database is used in order to analyse all the available data and answer questions.

Keywords: Ethereum, Blockchain, Ether, ETH, Smart Contracts, Cryptocurrency, BigQuery

1. Introduction

New disruptive technologies are being born constantly and totally change the way markets work. 2009 was the year when Satoshi Nakamoto introduced Bitcoin, a digital currency for peer to peer transactions, to the world. What he mainly achieved was not only to introduce the first digital currency but to open the doors for the whole cryptocurrency world. The total market capitalization of the cryptocurrency market is around $2.1 Trillion as of 31st August, 2021 [CoinMarketCap]. Blockchain technology and cryptocurrencies such as Ethereum and Bitcoin have attracted the rapidly increasing attention of many industries and researchers from various fields. Bitcoin is a cryptocurrency that operates only as a digital form of money, whereas Ethereum has a multifunctional character allowing users to create smart contracts that can interact and be executed. As a result, Ethereum offers the vision of a decentralized future, where third-parties are eliminated and users have direct access to the markets. Understanding how Ethereum works helps us identify what the role of blockchain will be in tomorrow's world, and when it makes sense to use it. A significant portion of this work is inspired from Athina Voulgari's Master Thesis at ETH Zürich, Management Technology and Economics Department.

2. Background

The elementary idea that led to the birth of blockchain was first proposed in 1990 by S. Haber and W.S. Stornetta. They introduced a novel, computationally feasible set of procedures to timestamp digital data, which would make it impractical to alter the timestamp after its creation [1]. One of the most important implications of this approach was that there would be no need for a third-party service to keep record of the timestamps. Since then, blockchain technology has revolutionized many fields including healthcare [2,3], transportation [4,5], digital forensics [6], and cybersecurity [7,8] due to its reliability, immutability, and transparency. These characteristics are the direct results of the blockchain structure: data is divided into a collection of blocks that are all linked together by means of cryptography. This structure prevents tampering with any arbitrary block without changing all others; hence, achieving immutability. The decentralized data handling also prevents the two parties in a transaction from manipulating the data stored in the network. In essence, the blockchain establishes a general agreement that verifies the details of a transaction without the need to trust the parties involved [9].

3. Distributed Ledger Technology (DLT)

Distributed Ledger Technology (DLT) refers to a novel approach to recording and sharing data across multiple data stores (ledgers), which store the exact same data and are maintained and controlled by a distributed network of computer servers, which are called nodes [10]. The innovative feature is that the ledger is not maintained by a central authority.

Each node, independently, builds and records the updates in the ledger. After that, these updates must be accepted by most of the network. In order to overcome the challenge of ensuring that all the nodes agree upon the validity of the ledger updates, there have been different mechanisms that define the way of agreeing on this; this agreement is called consensus. Once the consensus has been reached, the distributed ledger updates itself and the latest, agreed-upon version of the ledger, is saved in each node separately [11].

There are different consensus mechanisms to validate the agreement given the set-up of the distributed ledger. Consensus algorithms define the steps and the order of the steps that need to be done in order to produce an output. They constitute a fundamental component of distributed networks and are crucial for their functionality.

Some consensus mechanisms are explained in the next section.

4. Consensus Mechanisms

4.1 Proof of Work (PoW)

PoW existed before cryptocurrencies, but it became famous after being used in Bitcoin and Ethereum. It would be very easy for someone to hack a trustless and distributed system where everyone can participate and verify transactions. However, PoW prevents an entity from gaining power over the whole network.

Essentially, the miner (the person doing the work) has to find a value (also known as the nonce) which after being concatenated with the block contents and taking the hash gives a value which starts with a fixed number of zeroes (also known as the difficulty). The solution to finding this value is brute-force which is computationally very expensive. This transaction verification process is called mining. The purpose of mining is to verify the legitimacy of a transaction, avoiding any double spending. Therefore if anyone has to change the blockchain contents, they have to calculate the nonce value for that block and all the blocks after it up to the current block. This is highly infeasible.

Since the work requires high computational power, the nodes with higher computational power become the miners and control the blockchain. However if a miner (or a group of miners) have more than 51% of the hash power of the network, they essentially control what goes into the blockchain and the integrity is compromised. Also, the energy consumption of PoW blockchains is extremely high. Programming an attack to a PoW network is very expensive, and someone would need more money than he can be able to steal to do so.

4.2 Proof of Stake (PoS)

Proof of Stake is another way to achieve the agreement of transactions in the network. It is quite close to the PoW but the process is different. It was first introduced on a bitcointalk forum in 2011 [12].

The concept of Proof of Stake is very similar to a shareholder voting in a company. Whoever owns the most stake (or the most coins) gets the privilege to mine the next block. The more the stake, the greater the opportunity to mine a block successfully. Some blockchains also define stake in terms of the amount of time a person holds a coin (also referred to as the coin-age).

A 51% attack is generally considered more expensive in case of a proof of stake chain. However, many people are opposed to it ideologically as it gives the ‘rich’ control over the blockchain. Also, there is a problem of ‘nothing-at-stake’ where a malicious miner loses nothing by betting on two different forks of the chain.

4.3 Delegated Proof of Stake (DPoS)

DPoS is an attempt to overpass the inefficiencies that both PoW and PoS deal with. Delegated proof of stake uses real-time voting combined with a social system of reputation to achieve consensus.

In the network, there are delegates who are voted by token holders in order to mine the blocks. The stake-holders elect a delegate by means of a weighted election (the weights are proportional to their stake in the network). This delegate verifies the transactions, makes a block and receives a block for that. This system is usually considered to be one of the fastest consensus algorithms that can scale up-to millions of transactions per second and is used by the EOS blockchain.

4.4 Practical Byzantine Fault Tolerance (PBFT)

Practical Byzantine Fault Tolerance (PBFT) was initially introduced by Miguel Castro and Barbara Liskov in 1999. The algorithm has been designed in a way to detect malicious nodes in the network and defend against them. The consensus is reached by the fair nodes in the network. One node is primary and all the rest constitute the backup nodes. All of them communicate and try to reach an agreement. The nodes need to prove that the message came from a specific source, as well as that there have not been changes in its recording during its transmission. Some of the advantages are the significantly less energy usage and the ability to execute a transaction without the need of confirmation. However, this consensus can work only in a consortium of participants and especially in a small-sized network. It is mainly used in Hyperledger Fabric [13].

There are also other consensus algorithms like Proof of Importance, Proof of Authority, Proof of Burn, Proof of Activity, etc. It is an active area of research and new variants keep coming up with new blockchains.

5. Blockchain Technology

Blockchain was the first fully functional Distributed Ledger Technology (DLT) and the only one people have known for many years. Even today there is a confusion between DLT and blockchain. However, blockchain is nothing more than a DLT with a specific set of features that distinguish blockchain from all the previously referred DLT technologies [14]. Blockchain is a database, shared by means of blocks, which are connected with each other in the form of a chain.

Blockchain was first mainly mentioned in the Bitcoin white paper [15] from Satoshi Nakamoto, published in 2008, as the underlying technology of Bitcoin. However, blockchain was not built from scratch in order to serve the Bitcoin concept. What was new and disruptive was the combination of some similar technologies to create the world's first ever digital currency. Some of these technologies are peer-to-peer network, cryptography, digital signatures, nodes, hashing, consensus protocols, and mining.

Blockchain offers the main functionalities of DLTs. The three main properties that determine the Blockchain technology are decentralization, transparency, and immutability. The term of transparency comes in contradiction to the concept of privacy that is claimed to be a feature of blockchain. A person's identity is represented by a public address which is cryptographed. It is not the name of the sender that is referred to in a transaction, but the hash representation. As a result, the physical identity of a user is secure, while the digital identity is monitored and recorded, so that transparency is secure as well. Furthermore, blockchain ensures that no recorded transaction can be altered or removed. Even the slightest change in a block will completely change the hash address of the block. This immutability feature makes blockchain fully reliable. Finally, in a decentralized system, the information is not stored in a single entity. On the contrary, a basic component of a blockchain is its peer-to-peer (P2P) network. Each user can use the network and provide resources at the same time. A node can be any active electronic device that is connected to the Internet and provides an IP address. Although all nodes are equal, they can serve the network in different ways. A special reference needs to be made to these nodes that are called "full nodes" or miners. The purpose of a full node is to copy the latest blockchain data to a single device, while the device is connected to the network. As a result, this information is stored in all full nodes of the network and can only be altered if all these nodes are destroyed, which makes the system less vulnerable.

Blockchains have the potential to build a new generation of transactional applications that establish trust, accountability, and transparency at

their core while streamlining business processes and legal constraints.
There are three main types of blockchains-

a. Public Blockchain

In a public blockchain, anyone can join the network, transact, and become a miner. Also, anyone can see and parse the contents of such blockchains. Bitcoin and Ethereum are popular examples of this category.

b. Private Blockchain

It is an invite-only blockchain. It is usually hosted on private networks by enterprises who do not wish anyone to view or modify the data on the blockchain. Unlike public blockchains, it is usually centralised, however the data is still cryptographically secured from the company's view point. Multichain is an example of a private blockchain.

c. Federated or Consortium Blockchain

It is mainly used by banks. In this, the consensus is controlled by a predetermined set of nodes. The right to read the blockchain may be restricted or open. Corda is an example in this category.

6. Ethereum Blockchain

The Ethereum blockchain is a type of distributed ledger technology (DLT), a general term used for databases that store and share information in a decentralized network of independent nodes.

Ethereum is an open source, blockchain-based platform, which eliminates the need for trusted intermediaries. According to Dr. Gavin Wood, co-founder of Ethereum organization, Ethereum, taken as a whole, can be viewed as a transaction-based state machine: we begin with a genesis state and incrementally execute transactions to morph it into some final state. It is this final state which we accept as the canonical "version" of the world of Ethereum [16]. It allows everyone to set their own rules in transactions and state transition functions. This is done by involving smart contracts, a set of cryptographic rules that are executed under the conditions that the creator of the contract has set up.

Smart Contracts are one of the most promising features of blockchain technology and have created a lot of buzz over the years. They are essentially small computer programs that exist on the blockchain. This makes them immutable as long as the blockchain's integrity is not compromised. Therefore, the end-users can be sure that the program has not been changed or manipulated. This provides the people with trust in a distributed environment where they do not have to trust the other nodes or a centralised third party. This has fostered a new area of software development called decentralized applications (or dApps). These are applications that utilize the features of blockchains (immutability and lack of a central authority) in the backend and combine them with user-friendly frontends for non-technical users. The development of dApps for different application areas is an active area of research with the most popular ones being decentralised identity management, land record management, e-governance systems, etc. Essentially, blockchains help in maintaining a tamper-proof log that is used to establish ownership of actions. Therefore, it has wide applications and can help establish the trust of the users in the system.

Ethereum allows developers to program their own smart contracts, or "autonomous agents", as the Ethereum whitepaper [17] calls them. The solidity language is "Turing-complete", which means that it supports a broader set of computational instructions. In a nutshell, smart contracts can:

- Function as "multi-signature" accounts, so that funds are spent only when a required percentage of people agree.
- Manage agreements between users, e.g. if one purchases an insurance product from another.

- Provide utility to other contracts (similar to how a software library works).
- Store information about an application, such as domain registration information or membership records.

Unfortunately, the most promising feature of smart contracts - immutability also poses unique challenges from a security and a software engineering standpoint. Traditional software development life cycle (SDLC) is an iterative process where new features are added over time. Also, security issues and bugs found are fixed by releasing patches. However, as smart contracts are immutable they cannot be changed or fixed once they are deployed on the blockchain. Since Ethereum is a public blockchain, all the data is available in the public domain for hackers and other malicious actors to exploit.

One trade-off of Ethereum is related to its consensus mechanism (Proof of Work). It is the most popular algorithm for verifying the correctness of a transaction happening in the network. On one hand, it requires a lot of computational power in order to verify any transaction and at the end to build a block and validate it. On the other hand, the fact that it requires so much computational power makes it difficult to replicate and change the whole chain and as a result, reduces the risk of a 51% attack. At the same time, each individual solution is easy for the community to verify, which makes it easy to check all transactions for trustworthiness. The fact that it doesn't rely on a single third-party entity makes it a "trustless" and transparent network.

Even though most terms used in the context of Ethereum are not exclusive to this particular blockchain, there are a few terms that are. This section provides a brief introduction to the technical terms related to the Ethereum blockchain that are used in this paper. A comprehensive explanation of terms and concepts can be found in the Ethereum whitepaper [17].

7. The Ethereum Model

7.1 Accounts

Accounts play an important role in Ethereum. There are two kinds of accounts: Externally Owned Accounts (EOAs) and Contract Accounts. All the accounts in blockchain are characterized by four core elements:

- The nonce, a counter that is used to ensure that each transaction can only be processed once representing the number of transactions successfully sent from this account (if it is an EOA), or the number of contracts created by it (if it is a contract account).
- The account's current ether balance (the number of wei owned by an account).
- The account's code (only for contract accounts).
- The account's storage (permanent data stored, only for contract accounts).

7.2 Smart Contracts

The concept of smart contracts was introduced by N. Szabo in 1994 in an unpublished manuscript and then formally in 1997 [18]. Smart contracts are essentially blockchain-based applications, concisely, self executing programs which contain the terms and agreements between the parties involved.

A smart contract is represented by its 20-byte address, similar to an EOA address, such as '0xC2C747E0F7004F9E8817Db2ca4997657a7746928', and it can be developed in high-level languages, such as Solidity. When the contract is deployed on the network, the solidity code is compiled as bytecode. The feature that sets smart contracts apart is that they automatically verify whether the terms of the agreement have been fulfilled or not. Additionally, to ensure reliability, fault tolerance, and

transparency, the codes (smart contracts) are replicated on many nodes in the blockchain.
There is an infinite amount of contracts that can be created in the network. However, there are some "templates" that are commonly used, as they provide some main functionalities for contracts that want to achieve similar behaviours. These are token standards that define a common list of rules and are called ERC (Ethereum Request for Comments). Some of these are described below:

ERC20: This standard is a simple interface for creating tokens and can be reused from any application. It is the most commonly used standard in Ethereum. It consists of 6 main functions and 2 events. These functions are:

- *Total Supply:* defines the initial total supply of tokens
- *Balance:* monitors the balance of the contract every time
- *Transfer to:* sends tokens to specific accounts (wallets)
- *Transfer from:* enables token holders to exchange tokens after the initial distribution of the tokens
- *Approve:* approves other accounts to withdraw tokens from the account calling the function
- *Allowance:* after the approval of withdrawing tokens, it is used to see the amount of tokens that is withdrawn from the account

ERC20 is used from wallets in decentralized exchange platforms. Moreover, it is the main standard used for ICO contracts [19].

ERC721: This is a standard developed for non-fungible tokens. These tokens are completely unique. This feature of the standard makes it suitable for representation of assets that cannot be duplicated. A good example is the ownership over assets, such as houses, art pieces, or digital collectibles.

7.3 Transactions

Ethereum works on an account-based model, as our known banking system. The state of Ethereum changes every time a transaction of value or information is executed between two accounts. So, the state consists of these account addresses.

There are three kinds of transactions:

- *Ether transfer:* An external account can transfer Ether to another external account or a contract. This kind of transaction is very similar to a Bitcoin transfer.
- *Contract creation:* An external account can create a contract by transferring ether to a zero recipient's account. Then a new contract is created. The transaction should contain a code defining what the new contract will do. The execution of the transaction will create the bytecode of the new contract.
- *Contract call:* After a contract is deployed, an external account can call a contract when an account intends to implement one or more functions of a contract. The input data contains all the instructions related to the execution of the contract.

7.4 Ether

Ether (ETH) is the currency of Ethereum. It has intrinsic value as any currency. Ether derives its value from different factors. It is used to pay transaction fees in Ethereum. Furthermore, it can be accepted as payment currency from some retailers, such as the largest online Swiss retailer Digitec Galaxus AG. It can also be lent or borrowed. Finally, it is used as a medium of exchange to purchase Ethereum-based tokens. One example of the latter is an Initial Coin Offering (ICO) [20].

Ether serves all the three conditions in order to be considered as money within an economy:

- It works as a store of value, where investors purchase it and hold it for investment purposes given its predictable supply growth and congenital utility.

- It serves as a medium of exchange within the Ethereum network, and not only.

- It is used as a unit of account by various parties (including companies that have raised Ether via ICOs) [20].

The smallest denomination of Ether's metric system is Wei, where 1 Ether = $1e^{18}$ Wei.

7.5 Gas

On the Ethereum blockchain, the cost of performing transactions or processing smart contracts is measured by gas. It can be considered as the fuel for operating Ethereum. The price of gas itself is not constant, but is reported by miners based on the complexity and computational resources required for the execution of each block.
Every time a transaction or a contract is executed, users spend tokens which are translated to gas to run the computations. A user has to pay for the computational power spent, regardless of whether the transaction was successful or not. Gas is therefore an implicit incentive for developers to produce contracts that are of low-cost execution and for miners to validate transactions. Like this, the community avoids a waste of resources and eliminates the motivation for denial-of-service attacks.

7.6 Ethereum Virtual Machine

Ethereum Virtual Machine (EVM) is the computation engine of the Ethereum network and handles smart contracts deployment and execution. EVM is required in order to update the Ethereum state. An ether transfer from an account to another does not require an update of the state and as a result, EVM is not involved. However, it is used for any other action in the network.

Specifically, EVM is responsible for all of the following executions in the network:

- EVM confirms the validity of a transaction. This means that it checks the correct number of values, the validity of the signature and the match of the nonce with the nonce of the particular transaction account. If any of these elements is not correct, an error will return back.

- EVM checks if the gas is enough in order to execute the transaction. In case gas is not enough, the transaction fails. However, the transaction fee in such a case would not be refunded and would be paid to the miner.

- EVM transfers the value of Ether to the recipient's address.

- EVM calculates the total gas used and the transaction fee, in order to initialize the gas payment to the miner.

7.7 Blocks

Blocks are a fundamental component of Ethereum. All the information is stored in blocks. Each block consists of a block header and a set of transactions mined in the block. The chain is created by hashing some of the information of the previous block and including it in the new one. The hash IDs and timestamp associated with each block are the virtual glue that connects the blocks and forms the blockchain. Any alteration of the ledger or an individual block will make the code unworkable and, as a result, would be detectable.
The Ethereum blockchain began its life with its own genesis block. After this zero-block initiation, all other activities, such as mining, transactions, and contracts change the state every time.

7.8 Mining

Mining is the process of validating new transactions and creating new blocks. As a result, miner nodes will collect transactions from the transaction pool, will execute them in the EVM, solve the nonce and create the new block that fits to the chain. However, Ethereum is on the verge of moving to a Proof of Stake (PoS) consensus mechanism and the whole way of validating and receiving rewards would change in proof of stake consensus and would reform the way mining is done in the Ethereum blockchain.

8. Ethereum Roadmap

December 2013: Ethereum whitepaper [17] released by a few founding members, including Vitalik Buterin, Charles Hoskinson, Gavin Wood and others.

January 2014: Ethereum project is introduced at the North American Bitcoin conference.

July 2014: An Ethereum ICO was organized and attracted a lot of funding. A total of 60,102,216 ETH were sold for 31,591 BTC, which was worth $18,439,086 at that time.

May 2015: Ethereum launches Ethereum Olympic, a testing release where coins are not compatible with "real" ETH.

July 2015: Ethereum launches Ethereum frontier, a release broader to people who can mine ETH and build contracts.

August 2015: Kraken becomes the first digital currency exchange platform to trade ether.

May/June 2016: The first Decentralized Autonomous Organization is hacked resulting in a fork of Ethereum(ETH) and Ethereum Classic(ETC).

DAO raised $150 million using ICO. However, due to a gap in the smart contract coding that allowed the withdrawal of twice as much ether as invested, the Ethereum network implemented a hard fork. As a result, $50 million of Ether was stolen and the Ethereum blockchain split into Ethereum (ETH) and Ethereum Classic (ETC). This fork caused Ethereum to lose its immutability. After the fork, Ethereum attracted more users and was mainly established as the common Ethereum.

October 2016: Tangerine Whistle- Another hard fork for Ethereum in the block 2,463,000 on 18th of October 2016. This fork occurred in order to increase the gas prices and make them reflect the real computational complexity of some operations. As a result, any attack attempt is getting automatically more expensive and more difficult.

February 2017: Enterprise Ethereum Alliance is established by a group of companies in order to make Ethereum suitable for big businesses. As a result, the concept of permissioned Ethereum is introduced.

October 2017: Metropolis- Byzantium hard fork was the first half of Metropolis update. It occurred in the block 4,370,000 on October 16th, 2017. It included changes to reduce the complexity of EVM and provide more flexibility to smart contract developers.

June 2018: The U.S. Securities and Exchange Commission states that Ether (ETH) is not a security.

Feb 2019: Metropolis- Constantinople hard fork is the second half of Metropolis update. It occurred in the block 7,280,000 on February, 28th 2019.

Aug 2020: Ethereum Launches ETH 2.0 Multiclient Testnet – Medalla [21]: Hudson Jameson announced that "Eth 2.0 is a success, it is going really well right now" and hopes that the transition from Ethereum 1.0 to Ethereum 2.0 mainnet will be smooth.

Aug 2021: Ethereum's London Hard Fork [22] alters the way transaction fees are calculated, ideally smoothing them out and making them less volatile.

Data Analysis Setup

9. Data Accumulation

According to Blockchair, the Ethereum database size is currently around 280 GB. There is no central administrative system where the data can be downloaded from. One feature of Ethereum is that all data is publicly available. The nodes, which share data among each other, store a copy of the data, while the network executes the Ethereum protocol, which defines the rules of interaction of nodes with each other and/or smart contracts over the network.

In August 2018, Google Big Query managed to download all the blocks' data in an SQL database, in real time. The first phase relies on Ethereum ETL - an open-source tool developed in order to export the Ethereum blockchain into CSV or JSON files. It connects to an Ethereum node via its JSON RPC interface. The exported files are then moved to Google Cloud Storage, loaded into BigQuery, and finally verified. Then, the user can query the data in the BigQuery console or via API [23]. Since all the data is available there and is updated every day, this source of data will be used as the main one for the analysis of the Ethereum data.

10. Data Extraction & Usage

We extract and use all relevant data from the crypto_ethereum dataset [24] under the Google Cloud bigquery-public-data repository.

Google BigQuery is a data warehouse that handles large-scale data and makes it easy to access via SQL interface. Ethereum blockchain data is available on it and is updated daily. The entire blockchain data is stored in seven different tables: 'blocks', 'contracts', 'logs', 'token_transfers', 'tokens', 'traces', and 'transactions'.

11. Structure of the Data

In tables 11.1 to 11.7, we describe all available data fields, together with their data types. There are some fields that overlap in the tables, such as the block number and the block timestamp. The tables are structured this way in order to have a relational structure and be easy for analysis. Although the tables consist of all the data fields displayed, only some of them will be used in our analysis. We also depict a relational diagram of the database in Figure 11.1.

Table 11.1 Blocks table

blocks		
Field	**Data Type**	**Description**
timestamp	timestamp	The timestamp for when the block was collated
number	integer	The block number
hash	string	Hash of the block
aparent_hash	string	Hash of the parent block
nonce	string	Hash of the generated proof-of-work
sha3_uncles	string	SHA3 of the uncles data in the block
logs_bloom	string	The bloom filter for the logs of the block
transactions_root	string	The root of the transaction trie of the block
state_root	string	The root of the final state trie of the block
receipts_root	string	The root of the receipts trie of the block
miner	string	The address of the beneficiary to whom the mining rewards were given
difficulty	numeric	Integer of the difficulty for this block
total_difficulty	numeric	Integer of the total difficulty of the chain until this block
size	integer	The size of this block in bytes
extra_data	string	The extra data field of this block
gas_limit	integer	The maximum gas allowed in this block
gas_used	integer	The total used gas by all transactions in this block
transaction_count	integer	The number of transactions in the block

Table 11.2 Contracts table

contracts		
Field	**Data Type**	**Description**
address	string	Address of the contract
bytecode	string	Bytecode of the contract
function_sighashes	Array of strings	4-byte function signature hashes
is_erc20	boolean	Whether this contract is an ERC20 contract
is_erc721	boolean	Whether this contract is an ERC721 contract
block_timestamp	timestamp	Timestamp of the block where this contract was created
block_number	integer	Block number where this contract was created
block_hash	string	Hash of the block where this contract was created

Table 11.3 Tokens table

tokens		
Field	**Data Type**	**Description**
address	string	The address of the ERC20 token
symbol	string	The symbol of the ERC20 token
name	string	The name of the ERC20 token
decimals	string	The number of decimals the token uses.
total_supply	string	The total token supply.
block_timestamp	timestamp	Timestamp of the block where this token was created
block_number	integer	Block number where this token was created
block_hash	string	Hash of the block where this token was created

Table 11.4 Transactions table

transactions		
Field	**Data Type**	**Description**
hash	string	Hash of the transaction
nonce	integer	The number of transactions made by the sender prior to this one
transaction_index	integer	Integer of the transactions index position in the block
from_address	string	Address of the sender
to_address	string	Address of the receiver. null when its a contract creation transaction
value	numeric	Value transferred in Wei
gas	integer	Gas provided by the sender
gas_price	integer	Gas price provided by the sender in Wei
input	string	The data sent along with the transaction
receipt_cumulative_gas_used	integer	The total amount of gas used when this transaction was executed in the block
receipt_gas_used	integer	The amount of gas used by this specific transaction alone
receipt_contract_address	string	The contract address created, if the transaction was a contract creation, otherwise null
receipt_root	string	32 bytes of post-transaction stateroot (pre Byzantium)
receipt_status	integer	Either 1 (success) or 0 (failure) (post Byzantium)
block_timestamp	timestamp	Timestamp of the block where this transaction was in
block_number	integer	Block number where this transaction was in
block_hash	string	Hash of the block where this transaction was in

Table 11.5 Logs table

logs		
Field	**Data Type**	**Description**
log_index	integer	Integer of the log index position in the block
transaction_hash	string	Hash of the transactions this log was created from
transaction_index	integer	Integer of the transactions index position log was created from
address	string	Address from which this log originated
data	string	Contains one or more 32 Bytes non-indexed arguments of the log
topics	Array of strings	Indexed log arguments (0 to 4 32-byte hex strings)
block_timestamp	timestamp	Timestamp of the block where this log was in
block_number	integer	The block number where this log was in
block_hash	string	Hash of the block where this log was in

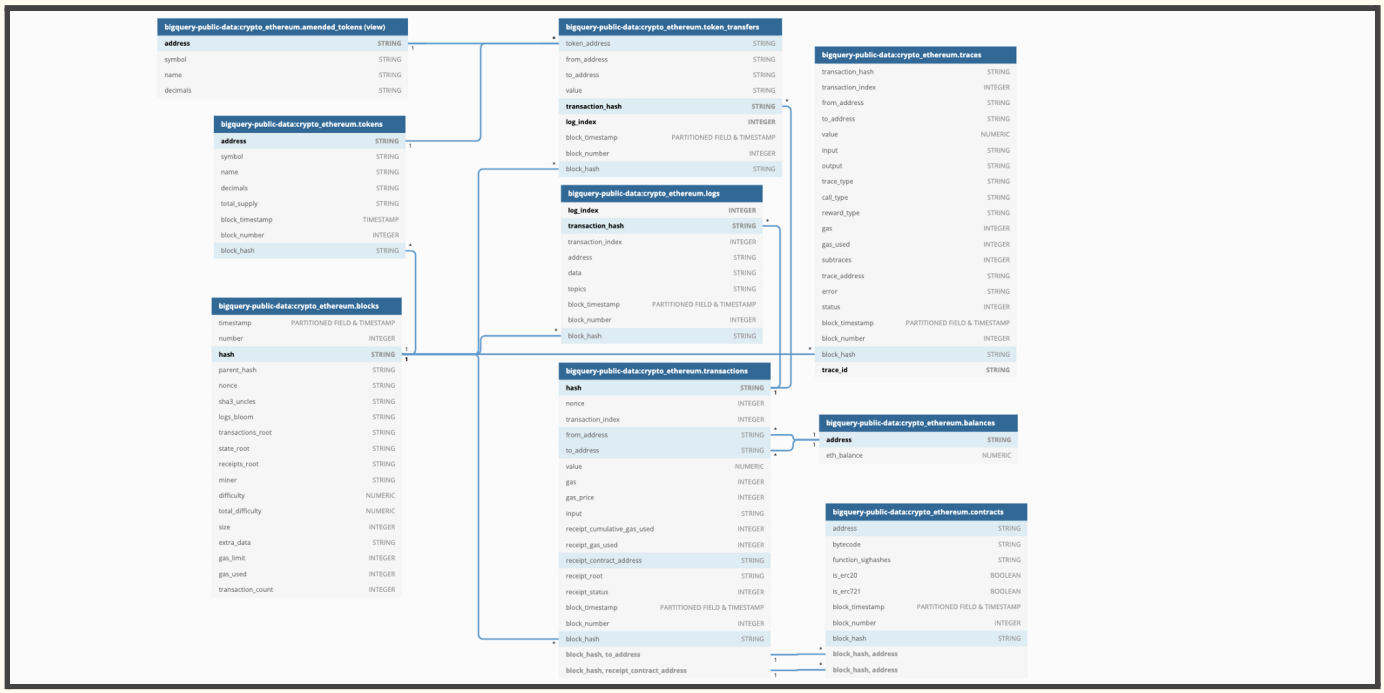
Table 11.6 Traces table

traces		
Field	**Data Type**	**Description**
transaction_hash	string	Transaction hash where this trace was in
transaction_index	integer	Integer of the transactions index position in the block
from_address	string	Address of the sender, null when trace_type is genesis or reward
to_address	string	Address of receiver or new contract or null
value	numeric	Value transferred in Wei
input	string	The data sent along with the message call
output	string	Output of message call/ bytecode of contract when trace_type is create
trace_type	string	One of call, create, suicide, reward, genesis, daofork
call_type	string	One of call, callcode, delegatecall, staticcall
reward_type	string	One of block, uncle
gas	integer	Gas provided with the message call
gas_used	integer	Gas used by the message call
subtraces	integer	Number of subtraces
trace_address	string	Comma separated list of trace address in call tree
error	string	Error if message call failed (Doesn't contain top-level trace errors)
status	boolean	Either 1 (success) or 0 (failure)
block_timestamp	timestamp	Timestamp of the block where this trace was in
block_number	integer	Block number where this trace was in
block_hash	string	Hash of the block where this trace was in

Table 11.7 Token transfers table

token_transfers		
Field	**Data Type**	**Description**
token_address	string	ERC20 token address
from_address	string	Address of the sender
to_address	string	Address of the receiver
value	string	Amount of tokens transferred (ERC20)/ id of the token transferred (ERC721)
transaction_hash	string	Transaction hash
log_index	integer	Log index in the transaction receipt
block_timestamp	timestamp	Timestamp of the block where this transfer was in
block_number	integer	Block number where this transfer was in
block_hash	string	Hash of the block where this transfer was in

Figure 11.1 Dataset relational diagram



Data Analytics

In the previous sections, distributed ledger technology and blockchain technology were analysed as concepts and the focus was mainly on the Ethereum structure. The Ethereum blockchain contains a lot of data, not only for the pure currency transactions, but also for the smart contracts that operate in the network and change the state of Ethereum.

In the succeeding sections, focus will be initially given to a comparison of how Ethereum is being used in general over time and to its "trends". Then, the number of transactions and total value of Ether will be used as a common metric in order to compare different external accounts, different contracts, and different tokens. Additionally, the top external accounts, contracts and tokens will be identified and compared with each other, in an attempt to define the reason why these accounts present a large number of transactions or value of Ether. At last, a closer look will be taken at the contracts analysis, and more particularly, their lifecycle and how they are used.

The SQL queries for the Google BigQuery Ethereum database that are used for this analysis will be provided in Appendix A: SQL Queries. Moreover, Etherscan.io, a Block Explorer, Search, API and Analytics Platform for Ethereum, will be used for identifying specific details for different addresses.

The complete notebook containing all analysis code and steps can be found on Kaggle at this link.

12. Generic Analytics

In this section we will analyse the demography of Ethereum blockchain and provide a snapshot of it as of 31st August, 2021.

12.1 Transactions

The Ethereum network has processed 1,266,449,457 transactions as of 31st August, 2021. Fig. 12.1 shows a visualisation of the number of transactions occurring over time in the Ethereum network. The peak was recorded on 9th May, 2021 when 1,716,600 transactions went through in a single day!

The Ethereum blockchain has attracted a lot of attention this year primarily because of NFTs being created on it. The number of transactions recorded on 31st August, 2021 is 1,206,314 which is more than double the total average number of daily transactions since the genesis of the blockchain, which is roughly 571,244.

It is also interesting to see what the total value of ether is in the transactions. There may be a lot of transactions, but of small ether value, or the other way around, meaning only a few transactions where a large amount of ether is transferred. The total value of ether transferred between accounts is 8,708,084,863 ETH. Fig. 12.2 shows a visualisation of the value of ether transferred per day.

The peak was recorded on 11th August, 2017 with approximately 46,000,000 ETH being transferred. The average value of ether transferred per day sits around 3,927,868 ETH per day.

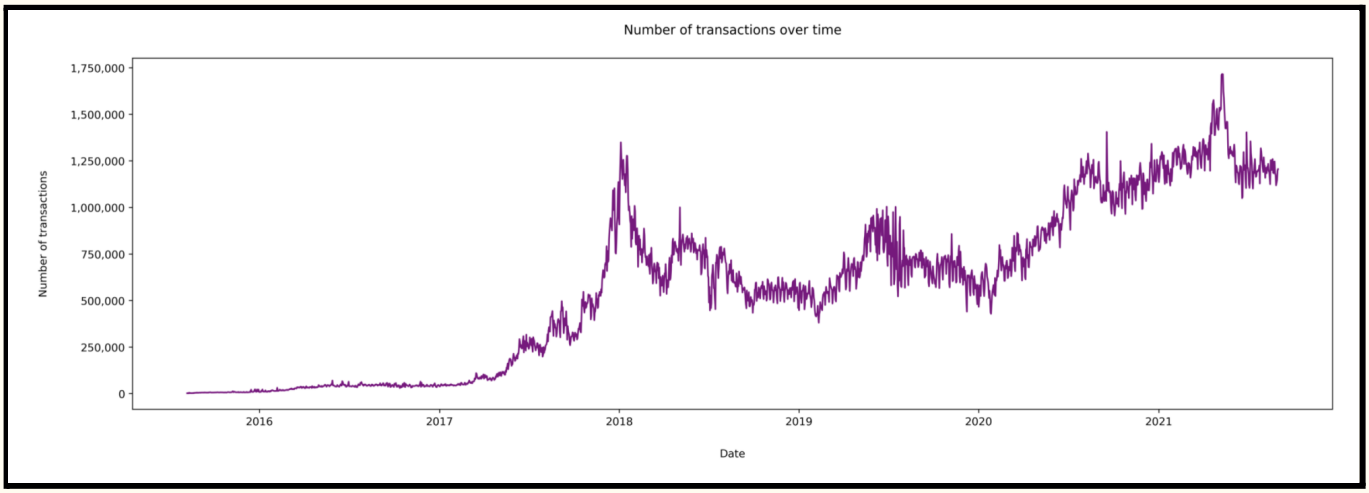


Figure 12.1 Number of transactions over time in the Ethereum network. The maximum number of transactions recorded on a single day was on 9th May, 2021 with 1,716,600 transactions.

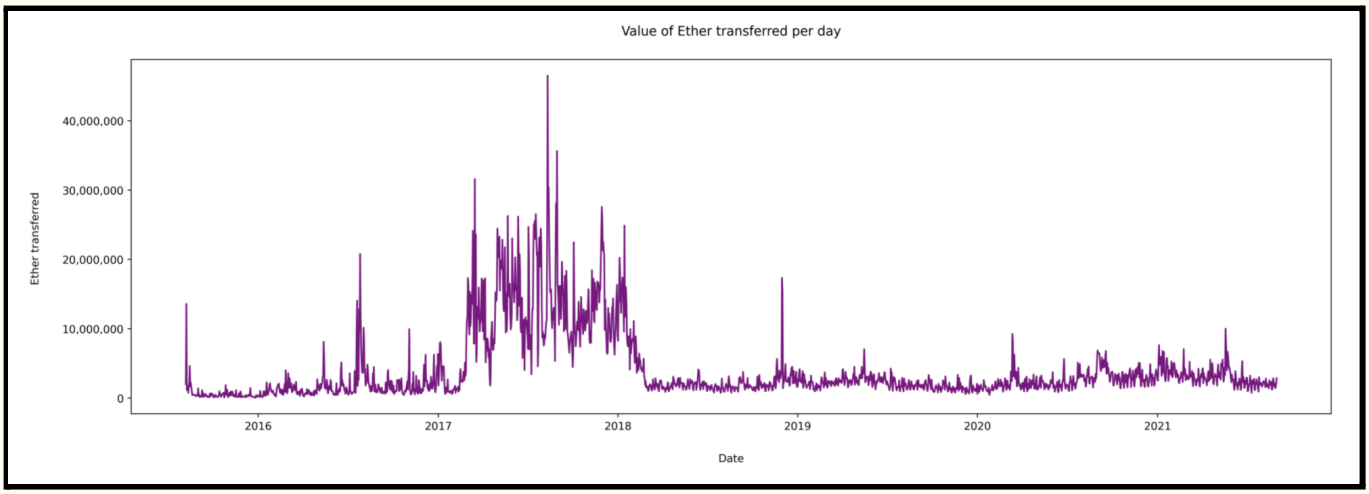


Figure 12.2 Value of Ether transferred per day

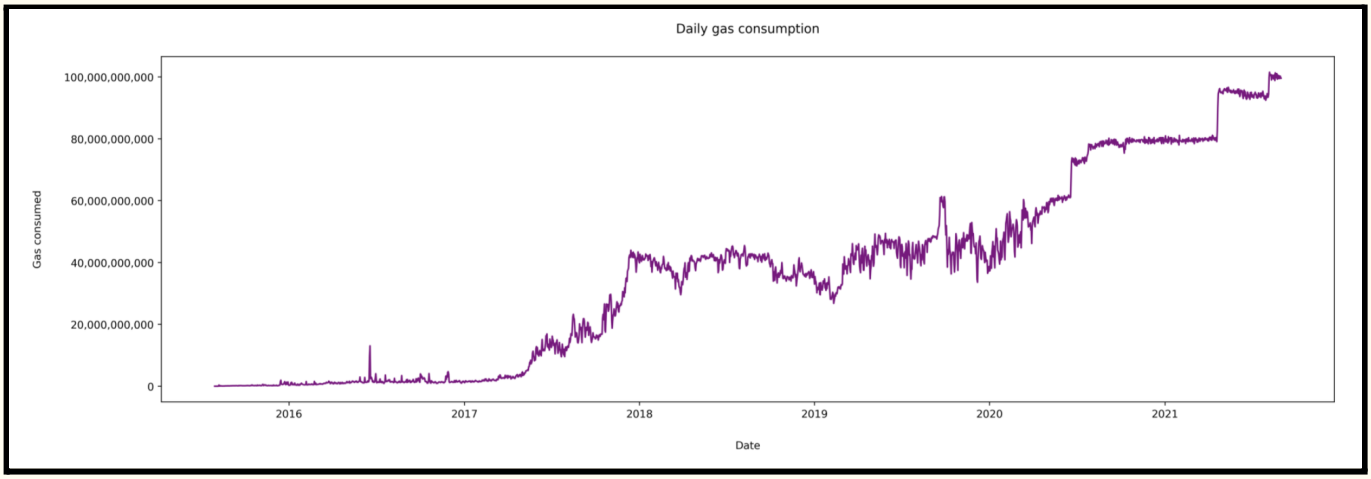


Figure 12.3 Daily gas consumption

12.2 Gas Used

The more complex a transaction is, the more gas is consumed to process the transaction. Fig. 12.3 shows a visualisation of the daily gas consumption in the Ethereum network. The continuous increase in gas being used to process transactions is proving to be a problem for the Ethereum network and therefore developers are looking at other alternatives like Solana and Cardano.

The average gas used is 36,438,799,019 per day. The highest amount of gas was used on 7th August, 2021 with 101,523,049,266 gas being used in a single day! Total gas used since the genesis of the Ethereum blockchain is 81,112,766,616,753.

In January 2018, the Ethereum network started operating, not only as a simple ether transfer network, but also as a contract execution network, and since the latter type of transactions costs more in terms of gas, we have seen continuous increase in gas being used to process transactions.

12.3 Gas Price

The gas price is set by the initiator of a transaction. The higher the price is, the more probable for the transaction to be prioritized higher than others.

Fig. 12.4 shows the average gas price per day in the Ethereum network. This average takes into consideration only the gas prices of the successful transactions. Gas prices of transactions that were not executed are not, since this would violate the real level of gas evaluation.

On June 11, 2020, the peak in the daily average gas price, just greater than 722,000,000,000 was recorded. The sum of gas price since the genesis is 61,575,600,617,544 and the average gas price of the Ethereum blockchain is 43,485,593,656.

12.4 Active addresses growth

Users constitute a major part of any network's success. The Ethereum network contains not only individuals, but also contracts. Fig. 12.5 shows a visualisation of the active address growth in the Ethereum network. The addresses are extracted based on their first transaction where the transaction value is greater than 0. As of 31st August 2021, there are 95,429,623 addresses in the Ethereum network which have done a transaction with a value larger than zero. In almost a year, compared to 2020, the number of distinct addresses has been doubled in 2021.

However, the same graph, considering also addresses with a zero balance, shows a significant difference. In Fig. 12.6, a sharp increase in the number of addresses by 20 Million can be observed. On 22nd September 2016, almost 20,000,000 addresses were suddenly created. This was the result of a DDos attack. A hacker took advantage of an underpriced Opcode and created around 20,000,000 useless Ethereum accounts. This led to the creation of millions of transaction traces, making data analysis difficult. At that specific time, the network was slowing down, making it more time consuming for miners and nodes to process blocks. The Ethereum network underwent a hard fork after that at block number 2,463,000.

So, it can be stated that the total number of addresses is approximately 95M, given the 115M transactions minus the 20M useless addresses.

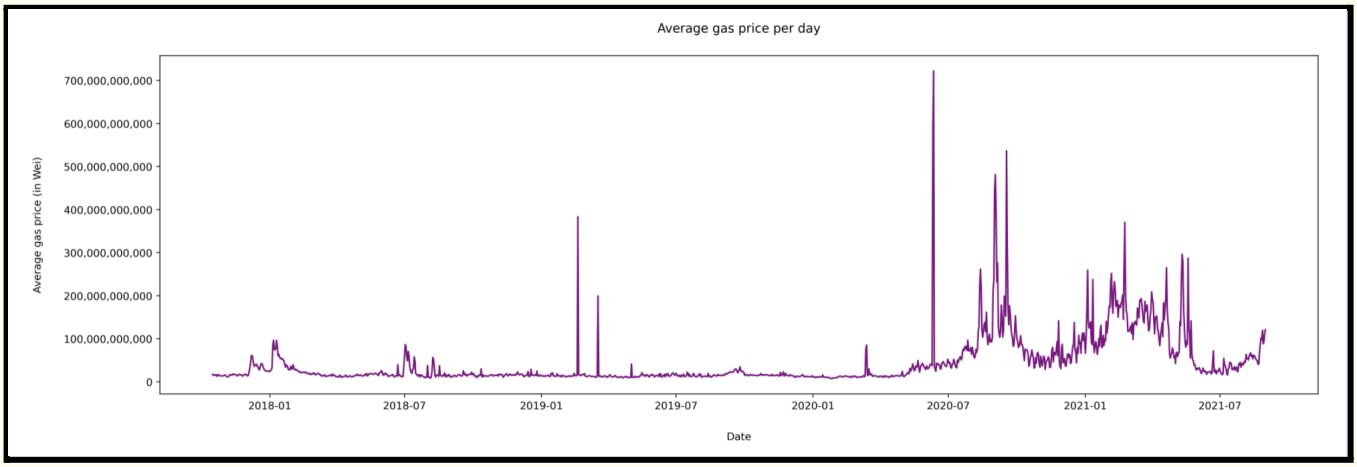


Figure 12.4 Daily average gas price (in Wei)

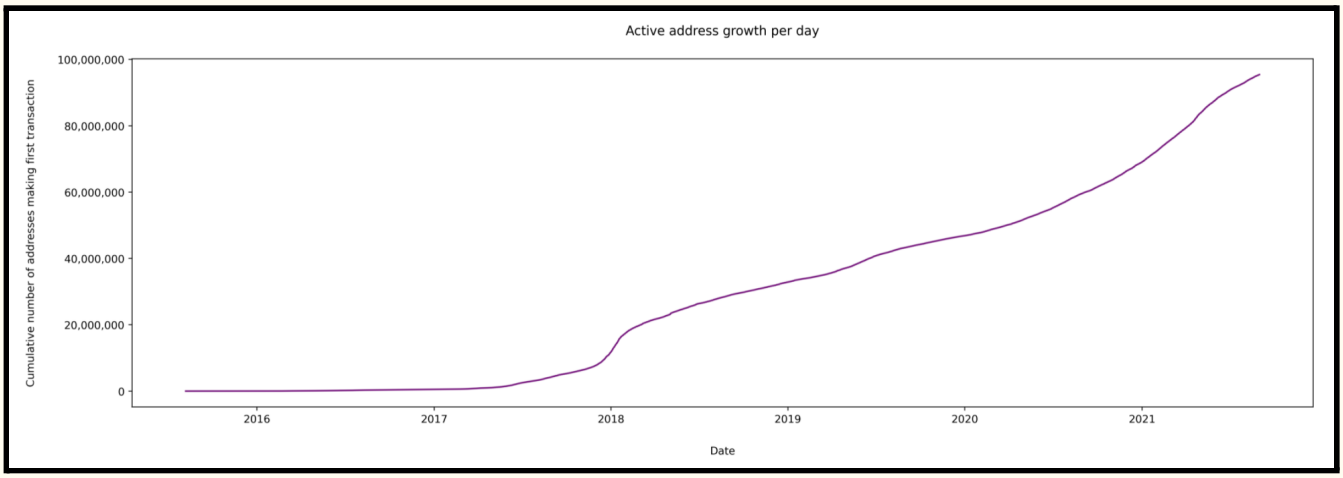


Figure 12.5 Cumulative active address growth per day (having made a transaction with value > 0)

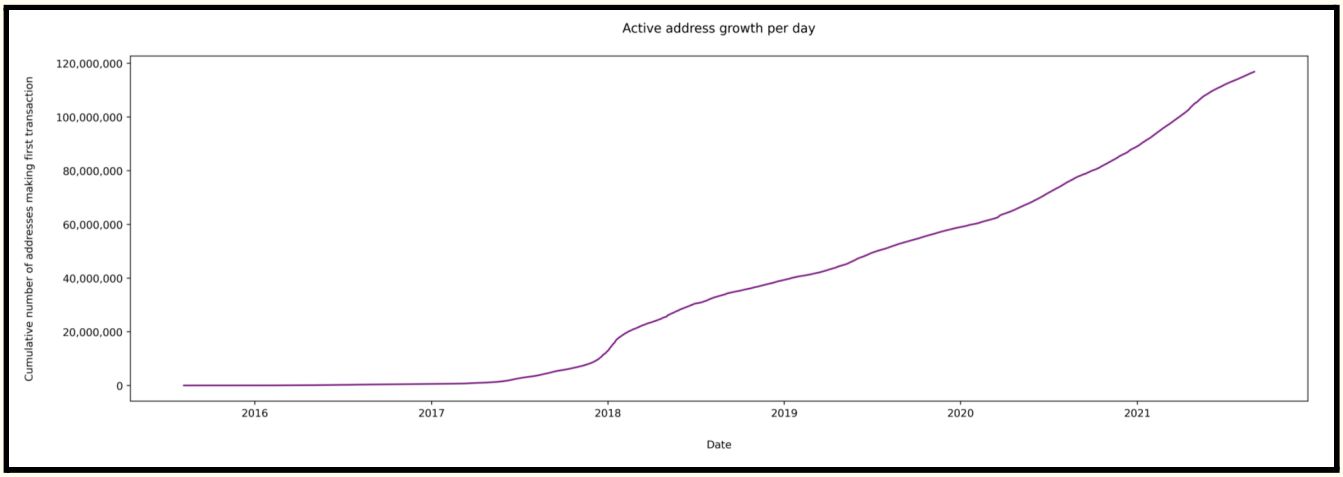


Figure 12.6 Cumulative active address growth (also considering addresses that haven't made any transaction)

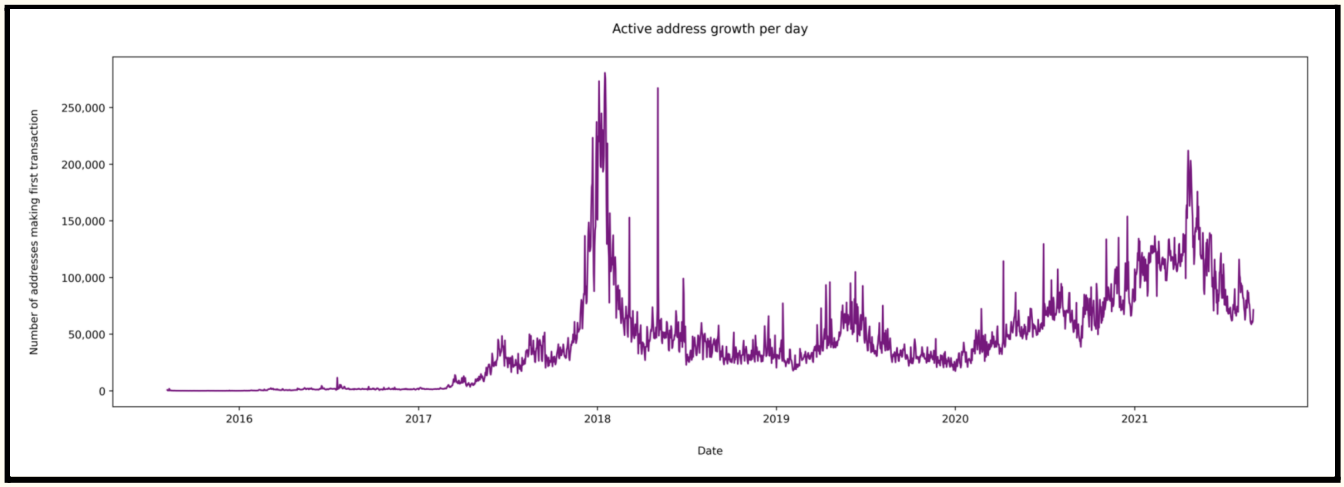


Figure 12.7 Active address growth per day (having made a transaction with value > 0)

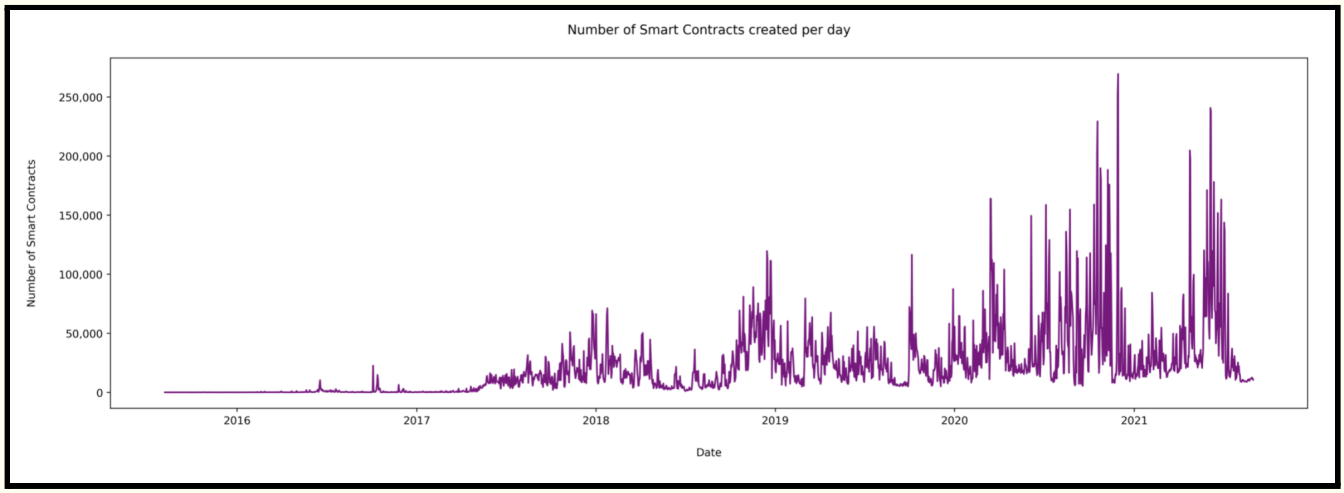


Figure 12.8 Number of Smart Contracts created per day

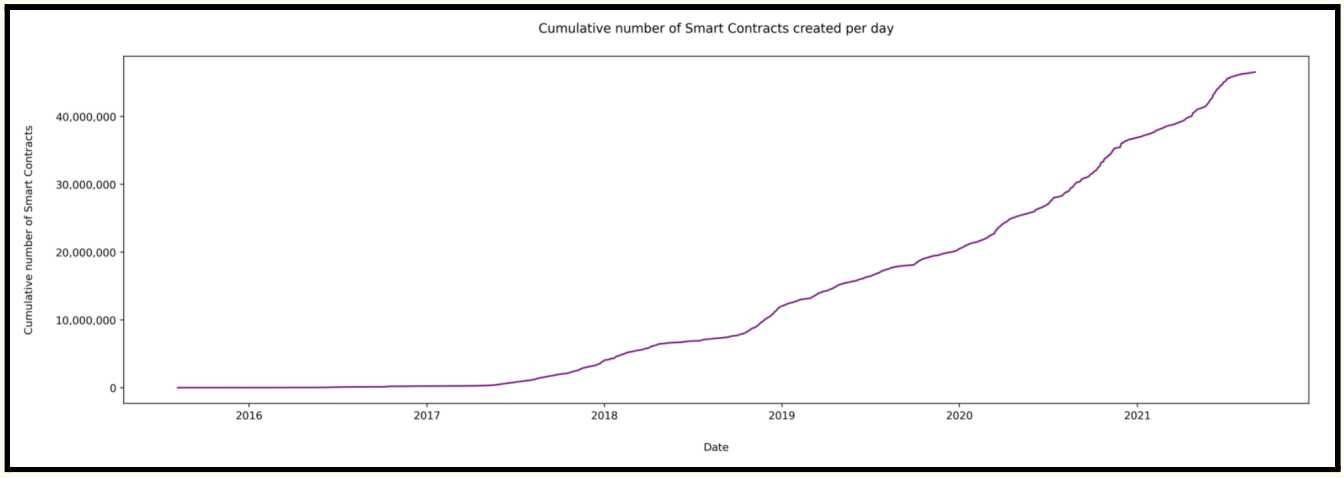


Figure 12.9 Cumulative number of Smart Contracts created per day

12.5 Smart Contracts

There are 46,376,065 addresses that represent smart contracts on the Ethereum blockchain. The growth of smart contract addresses is displayed in Fig. 12.8 and 12.9. Also, the contracts that are created directly from external users will be separated from the ones that are created indirectly from a contract generating another contract. This can be seen in Fig. 12.10 and 12.11. The number of smart contract addresses generated directly by users are 4,265,952. This means that around 42 Million smart contracts have been created indirectly from a contract generating another one.

The contract creation does not follow the pattern of transactions having high fluctuations. This means that over time users have understood the major functionality of Ethereum and its utility, i.e, Smart Contracts.

12.6 Classification of Addresses

In Figs. 12.5 to 12.7, the Ethereum address growth for external user addresses is shown and in Fig. 12.9, the Ethereum address growth for smart contracts is shown. It is visible that the allocation of the total addresses has changed since the inception of Ethereum. Initially most of the addresses were primarily external users and few contracts, but through time, contracts gradually gained more ground and are continuing to do so.

12.7 ETH Price

A visualisation representing ETH price in USD, as reported on [CoinMarketCap], is shown in Fig. 12.12. The ETH price has followed the same pattern as the number of transactions. The highest price of $4168.7 was recorded on 11th May, 2021. Ethereum's price as of 31st August, 2021 is $3433 with a total market cap of around $403,531,078,688.

13. Focused Analytics

13.1 Analytics of transactions

Transactions are one of the major sources for answering many questions about Ethereum. They display all the activity that is happening from external accounts to other external accounts or contracts. A transaction between two external accounts can only transfer value. On the other hand, a transaction from an external account to a contract account activates the function of the contract performing different actions such as creating new contracts, writing to internal storage, transferring tokens, performing other actions, etc.
There are 3 types of transactions:

- **Ether transfer:** An external account can transfer Ether to another account or a contract. The metadata of this transaction are the sender's address, the recipient's address, the timestamp and the value of the Ether sent.
- **Contract creation:** An external account can create a contract by transferring ether to a zero recipient's account. The metadata of this transaction are the sender's address, the timestamp and the input.
- **Contract call:** An external account can call a contract when the account intends to execute one or more functions of a contract. The input data contains all the instructions related to the execution of the contract. The metadata of this transaction are the sender's address, the recipient's address, the timestamp and the input.

Figure 13.1 shows that 92.7% of the transactions are Ether transfers. However, almost 7.2% of the transactions are calling a contract in order to execute a function. This indicates that a significant fraction of the activity in Ethereum is because of Ether transfers which is followed by contract calls, and just 0.1% of all transactions are contact creation transactions which is very low.

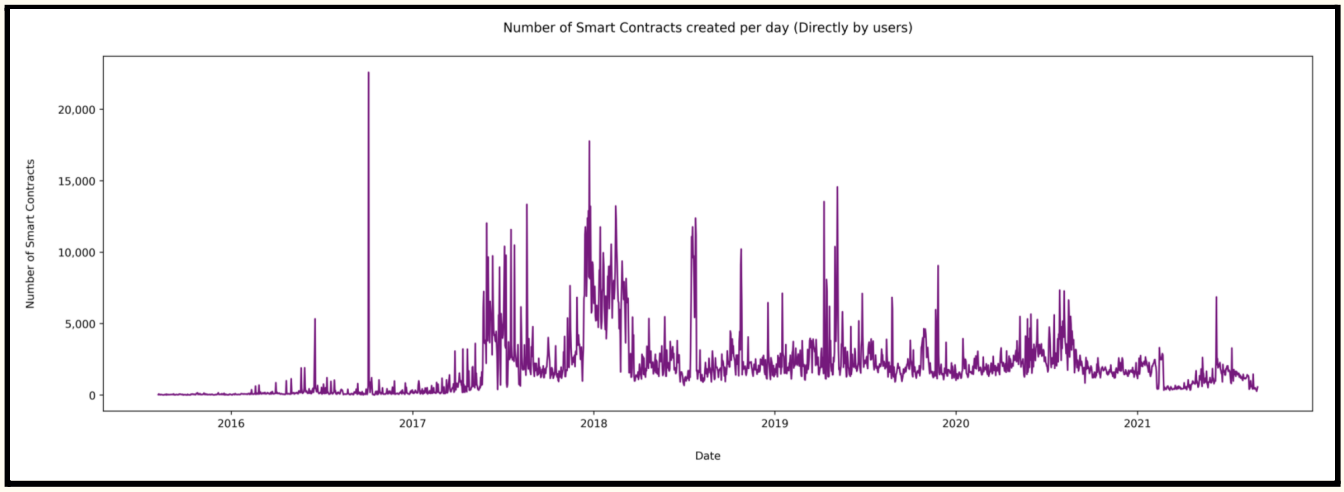


Figure 12.10 Number of Smart Contracts created per day (Directly by users)

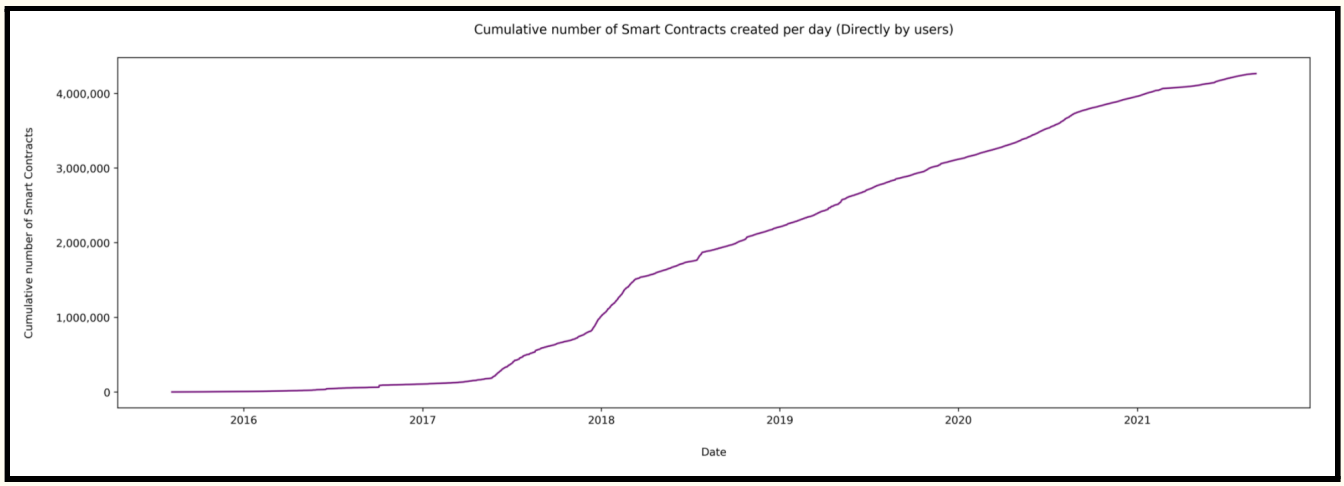


Figure 12.11 Cumulative number of Smart Contracts created per day (Directly by users)

Figure 12.12 ETH Price over time in USD

It is also important to mention that these numbers only refer to contracts created by external users, as transactions can be executed only from external users, and not from contracts.

The nature of transactions has changed a lot over the years. From the initialization of the Ethereum network until now, there is a gradual increase for all types of transactions but Ether transactions being in favour. The reason for this is that Ethereum launched after the successful story of Bitcoin and the generic evolution of digital currencies. Users were considering Ethereum just another cryptocurrency for peer to peer transactions or a speculative investment opportunity.

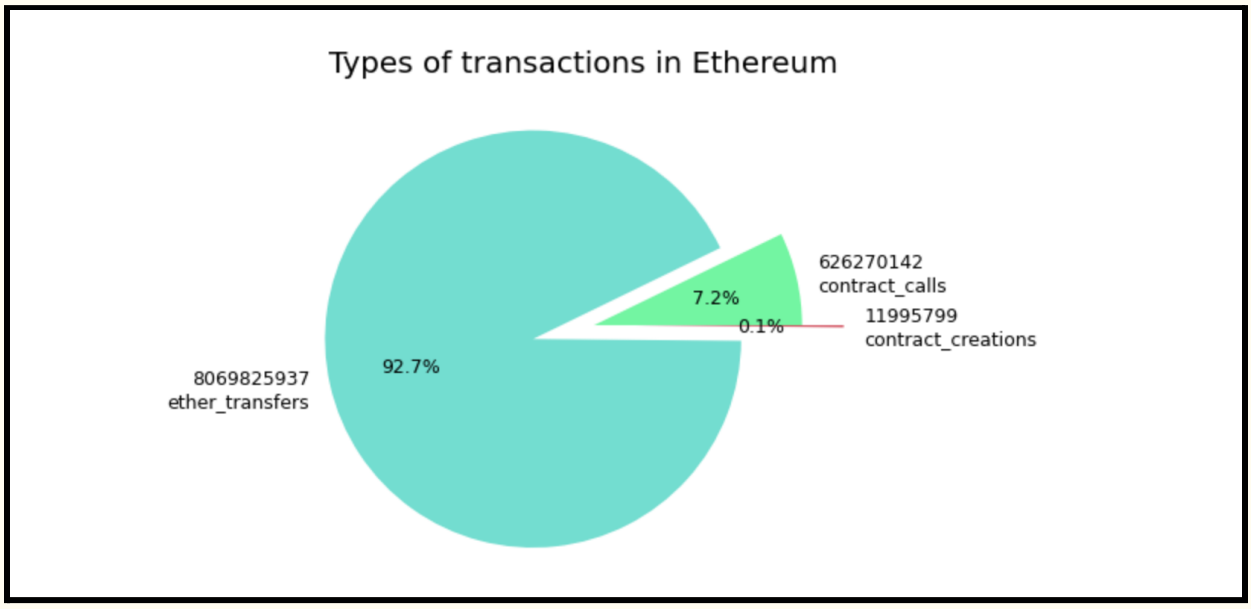


Figure 13.1 Types of transactions in Ethereum

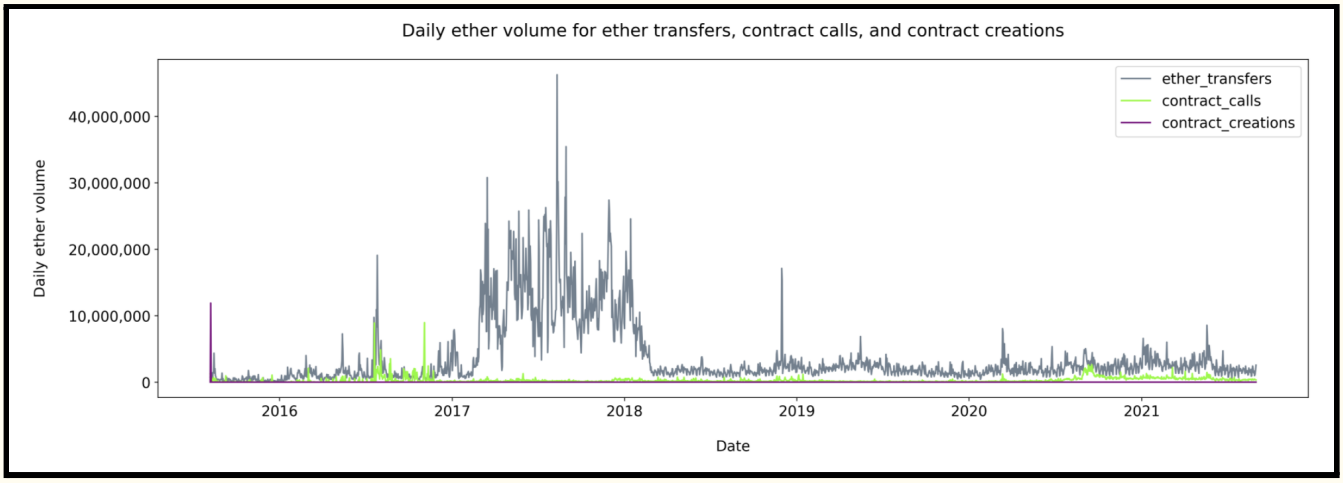


Figure 13.2 Daily ether volume for ether transfers, contract calls, and contract creations

13.2 Analytics of Volume

A total of 117 Million Ether has been supplied in the network, while all the Ethereum transactions have circulated more than 8.7 Billion Ether, as of 31st August, 2021. The daily aggregated ether value of these transactions, both for simple ether transfers, as well as for contract calls, can be found in Fig. 13.3. Most of the ether value is circulated in simple ether transfers, from account to account. For almost a year, from 2017 to 2018, the average volume of ether transfers is around 10M ether per day. The number of contract calls have risen significantly between 2020 and 2021. The primary reason for this being a surge in the NFT market.

13.3 Analytics of Accounts

This section presents some observations regarding the activity of the accounts, as measured by the ether volume and the number of transactions in the Ethereum network.

Let us identify in which category do the top 10 addresses, measured by ether balance, belong to. There are some basic categories that have been defined in [Etherscan.io], where all the transactions that are executed in the Ethereum network are displayed. The main categories that are used here are:

- Exchanges
- Decentralized exchanges
- Miners
- Token contracts
- ICO wallets

All the remaining addresses are assumed to be either simple contracts executing specific code or still unlabeled or external addresses. Etherscan identifies if an address is a contract address or a simple external address. There are some clear distinctions among the different categories.

• **Exchanges:** A list of centralized cryptocurrency exchanges which are online platforms that allow customers to buy and sell cryptocurrencies for other assets.
• **Decentralized exchanges:** A decentralized exchange (also known as a DEX) is an exchange market that does not rely on a 3rd party service to hold the client funds, but, instead, trades occur directly between users (peer to peer) through an automated process.
• **Miners:** Accounts that belong to big mining organizations or single individuals.
• **Token contracts:** A Token Contract is a type of smart contract that describes the rules of how the token could be generated and transferred between addresses, if it is splittable/fungible, etc.

In table 13.1, the top 10 addresses having the greatest balance of ether are displayed. Names of the address owners have been explored using Etherscan. Most of these addresses are ‘exchanges’.

The ranking of addresses based on the ether balance is only one metric of understanding which addresses are important for the Ethereum network. The number of transactions is another important indicator. Transactions are defined by an address which sends the transaction and an address which receives the transaction. The sender can only be an external account (single user, miner, or exchange), but a recipient address can be any address (external or contract address). In table 13.2, the top 10 addresses, measured by the number of transactions sent, are shown. Based on the data, most of these addresses are either exchanges or mining companies. We can see miners in this table because they are the ones who validate all the available transactions in the network, but we don’t see them in table 13.1 as they do not primarily invest in ether. Their balance is generally from the rewards they receive during the mining process.

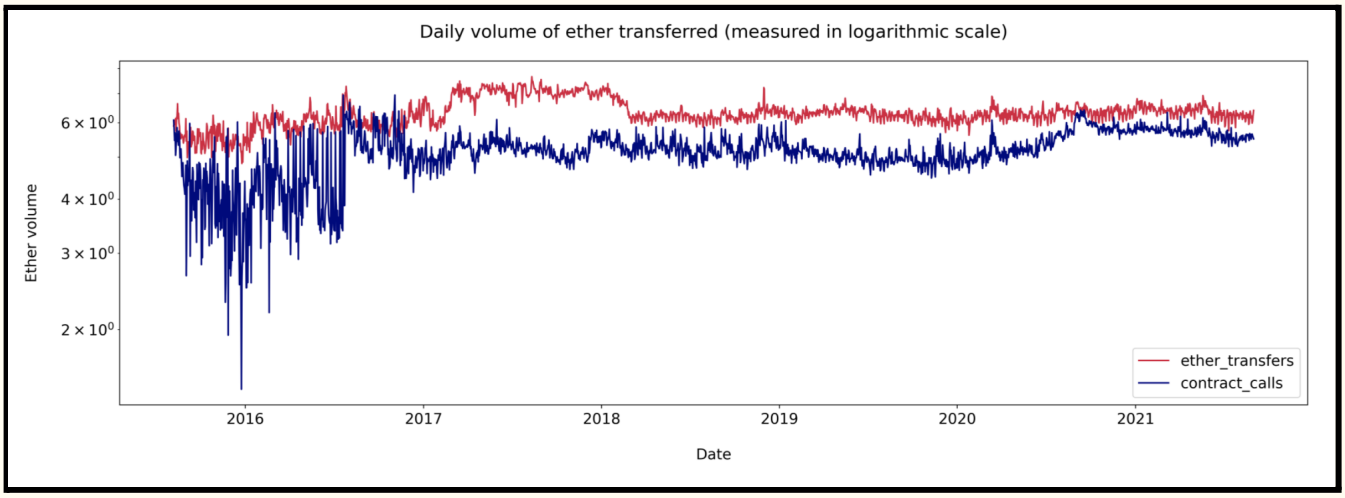


Figure 13.3 Daily volume of ether transferred (logarithmic scale)

Table 13.1 Top 10 addresses (by ether balance)

Address	Balance (ETH)	Name
0x00000000219ab540356cbb839cbe05303d7705fa	7,280,770	Ethereum 2.0 deposit contract
0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2	7,032,141	Wrapped Ether
0xbe0eb53f46cd790cd13851d5eff43d12404d33e8	2,296,896	Binance 7
0x73bceb1cd57c711feac4224d062b0f6ff338501e	1,949,046	-
0x4ddc2d193948926d02f9b1fe9e1daa0718270ed5	1,747,254	Compound Ether (cETH)
0x53d284357ec70ce289d6d64134dfac8e511c8a3d	1,378,734	Kraken 6
0x9bf4001d307dfd62b26a2f1307ee0c0307632d59	1,040,000	-
0x61edcdf5bb737adffe5043706e7c5bb1f1a56eea	989,498	Gemini 3
0xc61b9bb3a7a0767e3179713f3a5c7a9aedce193c	700,010	Proxy
0x229b5c097f9b35009ca1321ad2034d4b3d5070f6	657,334	Huobi 18

It is also important to identify the top addresses, based on the number of transactions received. Table 13.3, displays these addresses. This is a more diversified sample of addresses, since one can find exchanges, token contracts, as well as general contracts. It is surprising that although there are exchanges and token contracts in the table, these are not the same as the ones in table 13.2.

Again, the accounts that receive most of the transactions in the network just own very few ether. This makes sense as most of them are contracts and they are not supposed to hold ether but only to execute a sequence of orders. There is one interesting address in this list: CryptoKitties. CryptoKitties is a project utilizing ERC-721 tokens, which focuses on gaming where users can collect virtual cats. Each cat is represented by an Ethereum ERC-721 token, which means that they are all one-of-a-kind and can never be replicated, taken away from the owner, or destroyed. It is surprising to see a gaming platform ranking so high in the number of transactions received.

Table 13.2 Top 10 addresses (by number of transactions sent)

Address	Number of txns sent	Balance (ETH)	Name
0xea674fdde714fd979de3edf0f56aa9716b898ec8	37,053,732	38,526.8	Ethermine
0x52bc44d5378309ee2abf1539bf71de1b7d7be3b5	17,588,058	11,164.2	Nanopool
0x829bd824b016326a401d083b33d092293333a830	10,795,620	19,566.9	F2Pool Old
0xfbb1b73c4f0bda4f67dca266ce6ef42f520fbb98	10,572,809	14,156.8	Bittrex
0x3f5ce5fbfe3e9af3971dd833d26ba9b5c936f0be	7,735,908	3,079.2	Binance
0x5a0b54d5dc17e0aadc383d2db43b0a0d3e029c4c	6,061,515	34,644.3	Spark Pool
0xa7a7899d944fe658c4b0a1803bab2f490bd3849e	5,818,086	0	IDEX 2
0xd551234ae421e3bcba99a0da6d736074f22192ff	4,973,831	0.26	Binance 2
0x0681d8db095565fe8a346fa0277bffde9c0edbbf	4,943,611	2.46	Binance 4
0x564286362092d8e7936f0549571a803b203aaced	4,861,789	0.015	Binance 3

Table 13.3 Top 10 addresses (by number of transactions received)

Address	Number of txns received	Balance (ETH)	Name
0xdac17f958d2ee523a2206206994597c13d831ec7	110,324,410	0	Tether: USDT
0x7a250d5630b4cf539739df2c5dacb4c659f2488d	46,110,132	0	Uniswap V2: Router 2
0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48	13,381,871	0	USD Coin
0xa090e606e30bd747d4e6245a1517ebe430f0057e	12,161,990	2.1	Coinbase
0x8d12a197cb00d4747a1fe03395095ce2a5cc6819	11,538,109	17141.5	EtherDelta 2
0x2a0c0dbecc7e4d658f48e01e3fa353f44050c208	9,785,344	18076.9	IDEX
0x3f5ce5fbfe3e9af3971dd833d26ba9b5c936f0be	9,282,994	3079.2	Binance
0x174bfa6600bf90c885c7c01c7031389ed1461ab9	9,088,327	0	-
0x06012c8cf97bead5deae237070f9587f8e7a266d	4,875,001	6.9	CryptoKitties: Core
0x514910771af9ca656af840dff83e8264ecf986ca	4,764,625	0	ChainLink: LINK Token

13.4 Analytics of Smart Contracts

As referred to in section 12.5, there are in total 46,376,065 contract addresses. For these contracts, the source of creation is analyzed. There are contracts, which are created directly from an external address, and contracts which are contract-created. In the former ones, the user has to set the recipient address (to_address) to zero. In the latter ones, an external user calls the 'create' function of contract A, which in turn creates contract B. This is a message call (internal transaction) and it is not counted in the number of transactions. So, only the direct contract creations are considered transactions and represent only about 0.3% of the total transactions. The indirect contract creations are reflected in the contract calls. It is very interesting to see in Figs. 13.4 and 13.5 that most of the contracts were initially user-created contracts. However, as Ethereum is getting more mainstream and developers start participating and building more complicated contracts, contract-created contracts dominate the first ones since mid-2017. In fact, today there are over 42M contact-created contracts, while the user-created contracts are only over 4M. This means that an important fraction of the activity in the Ethereum network is using more complicated features of the contracts, such as the creation of another contract.

Diving into the contracts, we want to identify who is creating these contracts and how many of these contracts are totally identical. In order to identify the identical contracts, the output of the traces will be compared. The output is the bytecode of the contract whenever a new contract is created. The bytecode is a hex representation of what a contract is produced to execute.

From table 13.4, it can be clearly seen that a very small number of external users have directly created contracts. What is even more surprising is that only about 49K contracts are responsible for the creation of 42M contracts. Comparing the bytecodes of the user-created contracts, about 10% of them are unique, as there are only about 424K different bytecodes in the total of 4.2M contracts. At the same time, comparing the bytecode of the contract-created contracts, only about 35K of unique bytecodes are used in a total of 42M contracts! In conclusion, analysing the two types of contracts yields that only around 0.98% of the total contracts are completely unique.

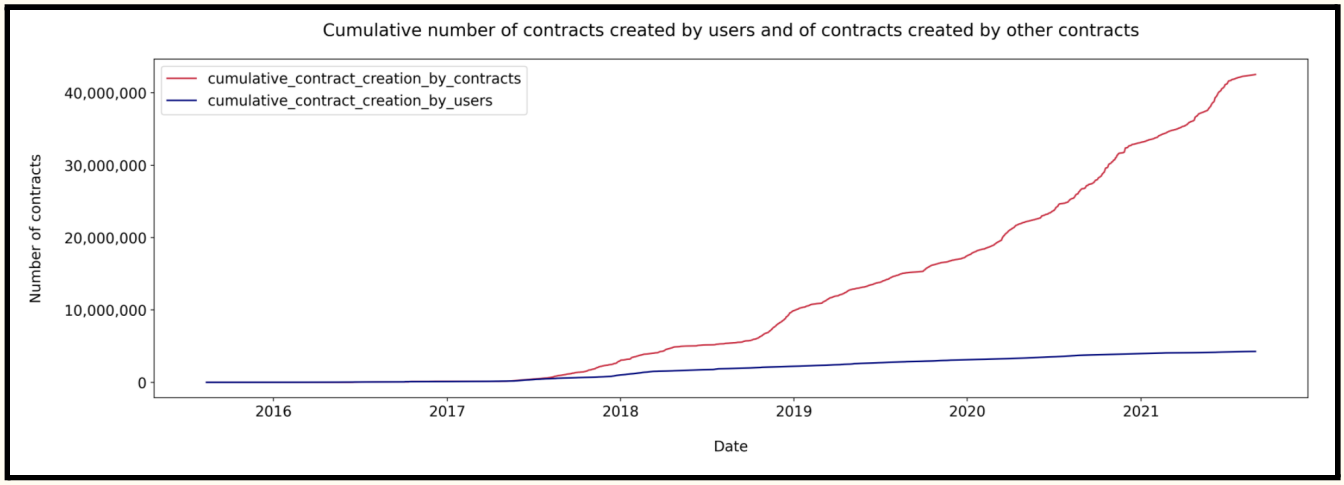


Figure 13.4 Cumulative number of contracts created by users, and contracts created by other contracts

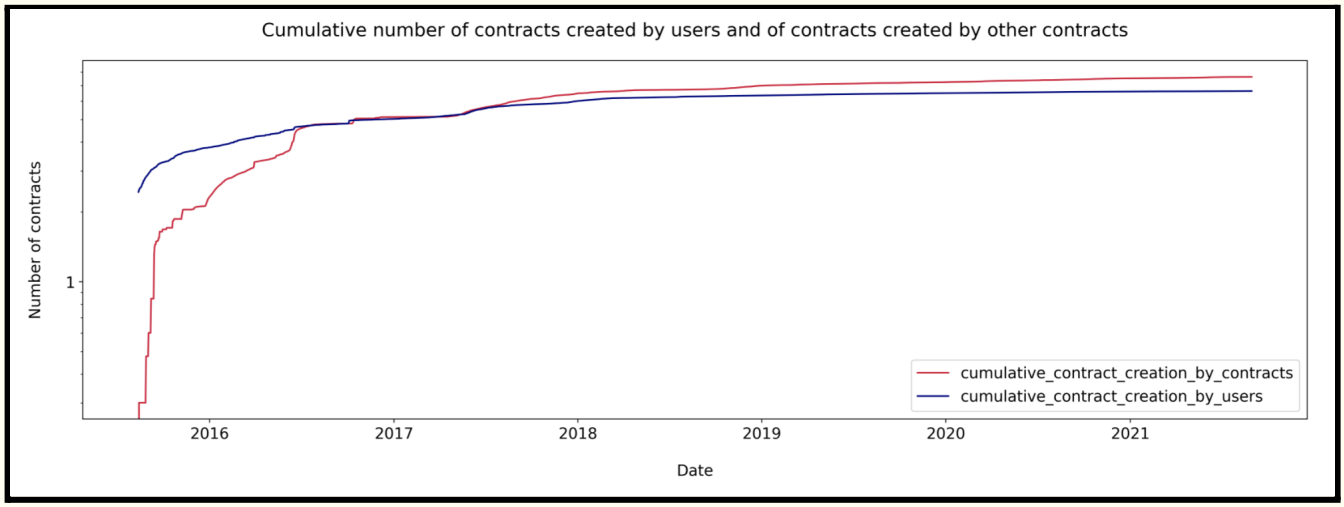


Figure 13.5 Cumulative number of contracts created by users, and contracts created by other contracts (Logarithmic scale)

Table 13.4 Different types of contracts

Type of contract	Number of contracts	Number of unique creators	Number of unique bytecodes
User-created contracts	4,209,516	210,172	424,875
Contract-created contracts	42,507,077	49,037	35,756

We have also segregated smart contracts based on their type, i.e, ERC20 and ERC721, and analysed them based on their popularity taking number of transactions as the metric. The results for this are presented in tables 13.5 and 13.6.

Table 13.5 shows that CryptoKitties, an ERC721 collectibles platform, is the most popular. Even after summing up the transaction count of the remaining top 9 contracts, CryptoKitties is still larger than all combined! The contracts in this list are primarily NFT (Non-Fungible Token) platforms or gaming platforms. The utility for cryptocurrencies in these areas is growing so fast and as a result we can see these contracts having so many transactions.

Similarly, it can be seen in table 13.6 that Tether: USDT is so much bigger than the other top ERC20 contracts. These ERC20 contracts mainly represent other cryptocurrencies and stable coins built on top of the Ethereum blockchain.

As we see so many transactions happening with smart contracts, they are moving the world towards mass adoption of decentralisation. There is a lot of potential in this space and we will surely see other use cases emerging for smart contracts in the near future.

Table 13.5 Top 10 Ethereum collectibles (ERC721 contracts) by popularity, based on number of transactions

Address	Transaction count	Name
0x06012c8cf97bead5deae237070f9587f8e7a266d	4,879,451	CryptoKitties: Core
0x06a6a7af298129e3a2ab396c9c06f91d3c54aba8	646,132	0xUniverse: PLANET Token
0xd73be539d6b2076bab83ca6ba62dfe189abc6bbe	442,760	Blockchain Cuties: Old BC Token
0x1a94fce7ef36bc90959e206ba569a12afbc91ca1	180,531	Crypton (CTN)
0xf5b0a3efb8e8e4c201e2a935f110eaaf3ffecb8d	147,500	Axie Infinity: AXIE Token
0x7fdcd2a1e52f10c28cb7732f46393e297ecadda1	119,472	HyperDragons Token
0x8c9b261faef3b3c2e64ab5e58e04615f8c788099	108,041	LucidSight-MLB-NFT
0x2a46f2ffd99e19a89476e2f62270e0a35bbf0756	95,986	MakersTokenV2 (MKT2)
0xf915bbfbb6c097dc327e64eec55e9ef4d110d627	78,666	Servant (SVT)
0xd2f81cd7a20d60c0d558496c7169a20968389b40	60,852	Etherbots (ETHBOT)

Table 13.6 Top 10 Ethereum tokens (ERC20 contracts) by popularity, based on number of transactions

Address	Transaction count	Name
0xdac17f958d2ee523a2206206994597c13d831ec7	111,998,435	Tether: USDT
0x174bfa6600bf90c885c7c01c7031389ed1461ab9	9,088,327	More Gold Coin (MGC)
0x514910771af9ca656af840dff83e8264ecf986ca	4,865,359	ChainLink Token (LINK)
0x86fa049857e0209aa7d9e616f7eb3b3b78ecfdb0	2,970,081	EOS: Old Token
0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2	2,899,686	Wrapped Ether (WETH)
0x0d8775f648430679a709e98d2b0cb6250d2887ef	2,725,166	Basic Attention: BAT Token
0xd26114cd6ee289accf82350c8d8487fedb8a0c07	2,509,541	OMG Network (OMG)
0x8fdcc30eda7e94f1c12ce0280df6cd531e8365c5	2,229,775	CpcToken (CPCT)
0xf230b790e05390fc8295f4d3f60332c93bed42e2	2,084,178	Tronix (TRX)
0x95ad61b0a150d79219dcf64e1e6cc01f0b64c4ce	1,815,190	SHIBA INU (SHIB)

Conclusion

In this work, we have done exploratory data analysis of the Ethereum blockchain. We started by giving an overview of the Digital Ledger Technology and then explained several consensus mechanisms. Further, we described blockchain technology and gave an explanation of how it works. Then we stated the different types of blockchains being used in the industry today.

A detailed description of the Ethereum blockchain is given by deeply talking about the Ethereum model and the Ethereum Roadmap. After that, the data analysis setup is reported and the dataset schema is presented along with a dataset relational diagram.

Further, we move on to the analytics and provide a generic analytics of the Ethereum blockchain related to transactions, gas, addresses, smart contracts and ETH price. Finally, we present a focused analytics of transactions, volume, accounts and smart contracts.

Summarizing all the research that has been done in this work, following conclusions can be drawn:

1. The number of transactions and the value of ether transferred are two good indicators to understand how the network behaves. Comparing these two, we have seen that they have not followed the same path during the genesis of the Ethereum blockchain. The number of transactions has an exponential growth, while the value of transferred ether fluctuates a lot.
2. Looking at ether price, we have concluded that it follows the pattern of the number of transactions. This fact helped us reach the conclusion that the number of transactions may affect the price of ether, or vice versa.
3. We have concluded that there are 95,429,623 active addresses in the Ethereum network as of 31st August, 2021, which includes external addresses, miner addresses and all contracts.
4. Focusing on transactions, we have found that almost 92.7% of them are simple ether transfers and 7.2% are contract calls. Only 0.1% are contract creation transactions.
5. There is a lack of diversity in the smart contracts ecosystem. There are just a few creators of the contracts, relative to their number. Moreover, this lack of diversity exists even in the bytecode. This means that most of the contracts in the Ethereum network are identical. More particularly, only about 0.98% of the total 46M contracts are totally unique!
6. We have also focused on the analysis of specific addresses, taking into consideration their holdings and the number of transactions. Generally, addresses with high ether balance are either exchanges or external users (individuals). On the other hand, taking into consideration the top 10 addresses that execute the most transactions, we have seen that exchanges are at the top of the list, as expected. However, miners have a dominant position in that list, as well. The main conclusion of this is, that, in most of the cases, the number of transactions in which an address is involved is inversely proportional to the ether holdings of the account.

This study also describes some key arguments that are worthy of further discussion. In this work, the main analysis is concentrated around addresses that are always measured by ether value (balance) and/or number of transactions (traffic). We feel that additional research of the Ethereum blockchain can be done by incorporating even more parameters to yield even more reliable results.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. DIMACS (GROUP), Haber S, Scott Stornetta W. How to Time-stamp a Digital Document. 1991.
2. Angraal S, Krumholz HM, Schulz WL. Blockchain Technology: Applications in Health Care. Circ Cardiovasc Qual Outcomes. 2017;10. doi:10.1161/CIRCOUTCOMES.117.003800
3. Du X, Chen B, Ma M, Zhang Y. Research on the Application of Blockchain in Smart Healthcare: Constructing a Hierarchical Framework. J Healthc Eng. 2021;2021: 6698122.
4. Subramanian N, Chaudhuri A, Kayıkcı Y. Blockchain Applications and Future Opportunities in Transportation. Blockchain and Supply Chain Logistics. 2020. pp. 39-48. doi:10.1007/978-3-030-47531-4_5
5. Niu B, Mu Z, Cao B, Gao J. Should multinational firms implement blockchain to provide quality verification? Transportation Research Part E: Logistics and Transportation Review. 2021. p. 102121. doi:10.1016/j.tre.2020.102121
6. Kumar M. Applications of Blockchain in Digital Forensics and Forensics Readiness. Blockchain for Cybersecurity and Privacy. 2020. pp. 339–364. doi:10.1201/9780429324932-20
7. Adebayo A, Rawat DB, Njilla L, Kamhoua CA. Blockchain‐enabled Information Sharing Framework for Cybersecurity. Blockchain for Distributed Systems Security. 2019. pp. 143–158. doi:10.1002/9781119519621.ch7
8. Sukhija N, Sample J-G, Bautista E. Advancing the Cybersecurity of Electronic Voting Machines Using Blockchain Technology. Essentials of Blockchain Technology. 2019. pp. 235–256. doi:10.1201/9780429674457-11
9. Werbach K. The Blockchain and the New Architecture of Trust. MIT Press; 2018.
10. Natarajan, S. K. Krause, and H. L. Gradstein. Distributed ledger technology (dlt) and blockchain. FinTech Note, 1, 2017.
11. S. Ray. The difference between blockchains and distributed ledger technology. 2018a
12. Blockgeeks. Proof of Work vs Proof of Stake
13. Blockgeeks. What is hyperledger? The most comprehensive step-by-step guide
14. BBVA. What is the difference between DLT and blockchain?
15. Bitcoin: A Peer-to-Peer Electronic Cash System
16. G. Wood. Ethereum yellow paper. Ethereum: A secure decentralised generalised transaction ledger.
17. Ethereum whitepaper: A Next-Generation Smart Contract and Decentralized Application Platform
18. Szabo N. Formalizing and securing relationships on public networks. 1997
19. V. Lai and K. O'Day. Ethereum ERC token standards. 2018
20. A. Sassano. Why Ether is Valuable. 2019
21. Ethereum Launches ETH 2.0 Multiclient Testnet - Medalla
22. Ethereum's London Hard Fork
23. E. Medvedev and A. Day. Ethereum in Big@uery: how we built this dataset. 2018
24. Kaggle: Ethereum blockchain: Complete live historical Ethereum blockchain data (BigQuery)

Appendix A: SQL Queries

SQL code for all the queries used in order to derive the proper information is presented in this section. Some of them are a mix of others, as the research questions had huge diversity.

- **Total number of transactions in the Ethereum network**

```
SELECT
  COUNT(transactions.hash)
FROM
  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
WHERE block_timestamp < '2021-09-01 00:00:00'
```

- **Average number of transactions per day**

```
SELECT
  COUNT(*) / COUNT(DISTINCT(DATE(block_timestamp)))
FROM
  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
WHERE block_timestamp < '2021-09-01 00:00:00'
```

- **Number of transactions over time**

```
SELECT
  DATE(block_timestamp) as date, COUNT(*) as number_of_transactions
FROM
  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
WHERE
  block_timestamp < '2021-09-01 00:00:00'
GROUP BY date
ORDER BY date
```

- **Value of Ether transferred per day**

```
SELECT
  DATE(block_timestamp) AS date, (SUM(value) / power(10,18)) AS total_value
FROM
  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
WHERE
  block_timestamp < '2021-09-01 00:00:00'
GROUP BY date
ORDER BY date
```

- **Daily gas consumption**

```
SELECT
  DATE(timestamp) as date, SUM(gas_used) as gas_used
FROM
  `bigquery-public-data.crypto_ethereum.blocks`
WHERE
  timestamp < '2021-09-01 00:00:00'
GROUP BY date
ORDER BY date
```

- **Average gas price per day**

```
SELECT
  DATE(block_timestamp) as date, avg(gas_price) as average_gas_price
FROM
  `bigquery-public-data.crypto_ethereum.transactions`
WHERE
  receipt_status = 1 AND block_timestamp < '2021-09-01 00:00:00'
GROUP BY date
ORDER BY date
```

- **Active Address Growth per day**

```
SELECT first_transaction AS date, COUNT(*) AS no_of_addresses_making_first_txn
FROM (
      SELECT from_address AS address, MIN(DATE(block_timestamp)) AS first_transaction
      FROM `bigquery-public-data.crypto_ethereum.transactions`
      WHERE value > 0 AND block_timestamp < '2021-09-01 00:00:00'
      GROUP BY address
      ORDER BY first_transaction
 )
GROUP BY first_transaction
ORDER BY date
```

- **Total number of smart contracts**

```
SELECT
  COUNT(DISTINCT(address)) as number_of_contracts
FROM
  `bigquery-public-data.crypto_ethereum.contracts`
WHERE
  block_timestamp < '2021-09-01 00:00:00'
```

- **Number of Smart Contracts created per day**

```
SELECT COUNT(DISTINCT(address)) as number_of_contracts, DATE(block_timestamp) as date
FROM `bigquery-public-data.crypto_ethereum.contracts`
WHERE
  block_timestamp < '2021-09-01 00:00:00'
GROUP BY date
ORDER BY date
```

- **Contract creation transactions per day (Directly by users)**

```
SELECT COUNT(*) as contracts_created, DATE(block_timestamp) as date
FROM `bigquery-public-data.crypto_ethereum.transactions`
WHERE to_address IS null AND block_timestamp < '2021-09-01 00:00:00'
GROUP BY date
ORDER BY date
```

- **Daily ether volume for ether transfers, contract calls, and contract creations**

```
WITH a AS (
  SELECT DATE(block_timestamp) AS date, SUM(value)/POWER(10,18) AS contract_calls
  FROM `bigquery-public-data.crypto_ethereum.transactions`
  WHERE input != '0x' AND to_address IS NOT null AND block_timestamp < '2021-09-01 00:00:00'
  GROUP BY date
),
b AS (
  SELECT DATE(block_timestamp) AS date2, SUM(value)/POWER(10,18) AS contract_creation
  FROM `bigquery-public-data.crypto_ethereum.transactions`
  WHERE to_address IS null AND block_timestamp < '2021-09-01 00:00:00'
  GROUP BY date2
),
c AS (
  SELECT DATE(block_timestamp) AS date3, SUM(value)/POWER(10,18) AS ether_transfers
  FROM `bigquery-public-data.crypto_ethereum.transactions`
  WHERE input = '0x' AND value IS NOT null AND block_timestamp < '2021-09-01 00:00:00'
  GROUP BY date3
)
SELECT contract_calls, ether_transfers, contract_creation, date
FROM a JOIN c ON date=date3 JOIN b ON date=date2
ORDER BY date
```

- **Top 10 addresses measured by balance (value of ether)**

```
WITH value_table AS (
   SELECT to_address AS address, value AS value
   FROM `bigquery-public-data.crypto_ethereum.traces`
   WHERE to_address IS NOT null
   AND block_timestamp < '2021-09-01 00:00:00'
   AND status=1
   AND (call_type NOT IN ('delegatecall', 'callcode', 'staticcall') OR call_type IS null)

   UNION ALL

   SELECT from_address AS address, -value AS value
   FROM `bigquery-public-data.crypto_ethereum.traces`
   WHERE from_address IS NOT null
   AND block_timestamp < '2021-09-01 00:00:00'
   AND status=1
   AND (call_type NOT IN ('delegatecall', 'callcode', 'staticcall') OR call_type IS null)

   UNION ALL

    SELECT miner as address, SUM(CAST(receipt_gas_used AS NUMERIC) * CAST(gas_price AS
NUMERIC)) AS value
   FROM `bigquery-public-data.crypto_ethereum.transactions` AS transactions
   JOIN `bigquery-public-data.crypto_ethereum.blocks` AS blocks
   ON blocks.number = transactions.block_number
   WHERE block_timestamp < '2021-09-01 00:00:00'
   GROUP BY blocks.miner

   UNION ALL

    SELECT from_address as address, -(CAST(receipt_gas_used AS NUMERIC) * CAST(gas_price AS
NUMERIC)) AS value
   FROM  `bigquery-public-data.crypto_ethereum.transactions`
   WHERE block_timestamp < '2021-09-01 00:00:00'
)
SELECT address, FLOOR(SUM(value) / power(10,18)) AS balance
FROM value_table
GROUP BY address
ORDER BY balance DESC
LIMIT 10
```

- **Top 10 addresses measured by number of sent transactions**

```
WITH value_table AS (
   SELECT to_address AS address, value AS value
   FROM `bigquery-public-data.crypto_ethereum.traces`
   WHERE to_address IS NOT null
   AND block_timestamp < '2021-09-01 00:00:00'
   AND status=1
   AND (call_type NOT IN ('delegatecall', 'callcode', 'staticcall') OR call_type IS null)

   UNION ALL

   SELECT from_address AS address, -value AS value
   FROM `bigquery-public-data.crypto_ethereum.traces`
   WHERE from_address IS NOT null
   AND block_timestamp < '2021-09-01 00:00:00'
   AND status=1
   AND (call_type NOT IN ('delegatecall', 'callcode', 'staticcall') OR call_type IS null)

   UNION ALL

    SELECT miner as address, SUM(CAST(receipt_gas_used AS NUMERIC) * CAST(gas_price AS
NUMERIC)) AS value
   FROM `bigquery-public-data.crypto_ethereum.transactions` AS transactions
   JOIN `bigquery-public-data.crypto_ethereum.blocks` AS blocks
   ON blocks.number = transactions.block_number
   WHERE block_timestamp < '2021-09-01 00:00:00'
   GROUP BY blocks.miner

   UNION ALL

    SELECT from_address as address, -(CAST(receipt_gas_used AS NUMERIC) * CAST(gas_price AS
NUMERIC)) AS value
   FROM  `bigquery-public-data.crypto_ethereum.transactions`
   WHERE block_timestamp < '2021-09-01 00:00:00'
),
a AS (
   SELECT SUM(value)/POWER(10,18) AS balance, address
   FROM value_table
   GROUP BY address
   ORDER BY balance DESC
),
b AS (
   SELECT to_address, COUNT(transactions.hash) AS tx_recipient
   FROM  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
```

```
    WHERE block_timestamp < '2021-09-01 00:00:00'
    GROUP BY to_address
),
c AS (
    SELECT from_address, COUNT(transactions.hash) AS tx_sender
    FROM  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
    WHERE block_timestamp < '2021-09-01 00:00:00'
    GROUP BY from_address
)
SELECT from_address, tx_sender, balance
FROM c LEFT JOIN a ON (a.address = c.from_address)
ORDER BY tx_sender DESC
LIMIT 10
```

- **Top 10 addresses measured by number of received transactions**

```
WITH value_table AS (
    SELECT to_address AS address, value AS value
    FROM `bigquery-public-data.crypto_ethereum.traces`
    WHERE to_address IS NOT null
    AND block_timestamp < '2021-09-01 00:00:00'
    AND status=1
    AND (call_type NOT IN ('delegatecall', 'callcode', 'staticcall') OR call_type IS null)

    UNION ALL

    SELECT from_address AS address, -value AS value
    FROM `bigquery-public-data.crypto_ethereum.traces`
    WHERE from_address IS NOT null
    AND block_timestamp < '2021-09-01 00:00:00'
    AND status=1
    AND (call_type NOT IN ('delegatecall', 'callcode', 'staticcall') OR call_type IS null)

    UNION ALL

     SELECT miner as address, SUM(CAST(receipt_gas_used AS NUMERIC) * CAST(gas_price AS
NUMERIC)) AS value
    FROM `bigquery-public-data.crypto_ethereum.transactions` AS transactions
    JOIN `bigquery-public-data.crypto_ethereum.blocks` AS blocks
    ON blocks.number = transactions.block_number
    WHERE block_timestamp < '2021-09-01 00:00:00'
    GROUP BY blocks.miner

    UNION ALL
```

```
    SELECT from_address as address, -(CAST(receipt_gas_used AS NUMERIC) * CAST(gas_price AS
NUMERIC)) AS value
   FROM  `bigquery-public-data.crypto_ethereum.transactions`
   WHERE block_timestamp < '2021-09-01 00:00:00'
),
a AS (
   SELECT SUM(value)/POWER(10,18) AS balance, address
   FROM value_table
   GROUP BY address
   ORDER BY balance DESC
),
b AS (
   SELECT to_address, COUNT(transactions.hash) AS tx_recipient
   FROM  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
   WHERE block_timestamp < '2021-09-01 00:00:00'
   GROUP BY to_address
),
c AS (
   SELECT from_address, COUNT(transactions.hash) AS tx_sender
   FROM  `bigquery-public-data.crypto_ethereum.transactions` AS transactions
   WHERE block_timestamp < '2021-09-01 00:00:00'
   GROUP BY from_address
)
SELECT to_address, tx_recipient, balance
FROM b LEFT JOIN a ON (a.address = b.to_address)
ORDER BY tx_recipient DESC
LIMIT 10
```

- **Cumulative number of contracts created by users and of contracts created by other contracts**

```
WITH a AS (
   SELECT DATE(block_timestamp) AS date, COUNT(*) AS contracts_creation
   FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
   WHERE block_timestamp < '2021-09-01 00:00:00'
      AND trace_type = 'create'
      AND trace_address IS null
      GROUP BY date
),
b AS (
   SELECT date, SUM(contracts_creation)
   OVER (ORDER BY date) AS ccc, LEAD(date, 1) OVER (ORDER BY date) AS next_date
   FROM a
   ORDER BY date
),
calendar AS (
```

```
    SELECT date
    FROM UNNEST(generate_date_array('2015-07-30', '2021-08-31')) AS date
),
c AS (
    SELECT calendar.date, ccc
    FROM b JOIN calendar ON b.date <= calendar.date
    AND calendar.date < b.next_date
    ORDER BY calendar.date
),
d AS (
    SELECT date(block_timestamp) AS date1, COUNT(*) AS contracts_creation1
    FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
    WHERE block_timestamp < '2021-09-01 00:00:00'
        AND trace_type = 'create'
        AND trace_address IS NOT null
        GROUP BY date1
),
e AS (
    SELECT date1, SUM(contracts_creation1)
    OVER (ORDER BY date1) AS ccc1, LEAD(date1, 1) OVER (ORDER BY date1) AS next_date1
    FROM d
    ORDER BY date1
),
calendar1 AS (
    SELECT date1
    FROM UNNEST(generate_date_array('2015-07-30', '2021-08-31')) AS date1
),
f AS (
    SELECT calendar1.date1, ccc1
    FROM e JOIN calendar1 ON e.date1 <= calendar1.date1
        AND calendar1.date1 < e.next_date1
    ORDER BY calendar1.date1
)
SELECT date1, date, f.ccc1 as cumulative_contract_creation_by_contracts, c.ccc as
cumulative_contract_creation_by_users
FROM c JOIN f ON f.date1 = c.date
ORDER BY f.date1
```

- **Number of unique creators of user-created contracts**

```
SELECT COUNT(DISTINCT(from_address)) AS unique_creators
FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
WHERE block_timestamp < '2021-09-01 00:00:00'
    AND trace_type='create'
    AND trace_address IS null
```

- **Total number of bytecodes of user-created contracts**

```
SELECT COUNT(output) AS unique_bytecodes
FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
WHERE block_timestamp < '2021-09-01 00:00:00'
   AND trace_type='create'
   AND trace_address IS null
```

- **Number of unique bytecodes of user-created contracts**

```
SELECT COUNT(DISTINCT(output)) AS unique_bytecodes
FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
WHERE block_timestamp < '2021-09-01 00:00:00'
   AND trace_type='create'
   AND trace_address IS null
```

- **Number of unique creators of contract-created contracts**

```
SELECT COUNT(DISTINCT(from_address)) AS unique_creators
FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
WHERE block_timestamp < '2021-09-01 00:00:00'
   AND trace_type='create'
   AND trace_address IS NOT null
```

- **Total number of bytecodes of contract-created contracts**

```
SELECT COUNT(output) AS unique_bytecodes
FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
WHERE block_timestamp < '2021-09-01 00:00:00'
   AND trace_type='create'
   AND trace_address IS NOT null
```

- **Number of unique bytecodes of contract-created contracts**

```
SELECT COUNT(DISTINCT(output)) AS unique_bytecodes
FROM  `bigquery-public-data.crypto_ethereum.traces` AS traces
WHERE block_timestamp < '2021-09-01 00:00:00'
   AND trace_type='create'
   AND trace_address IS NOT null
```

- **10 most popular Ethereum collectibles (ERC721 contracts), by number of transactions**

```
SELECT contracts.address, COUNT(1) AS tx_count
  FROM `bigquery-public-data.crypto_ethereum.contracts` AS contracts
        JOIN `bigquery-public-data.crypto_ethereum.transactions` AS transactions ON
(transactions.to_address = contracts.address)
  WHERE contracts.is_erc721 = TRUE AND contracts.block_timestamp < '2021-09-01 00:00:00'
  GROUP BY contracts.address
  ORDER BY tx_count DESC
  LIMIT 10
```

- **10 most popular Ethereum tokens (ERC20 contracts), by number of transactions**

```
SELECT contracts.address, COUNT(1) AS tx_count
  FROM `bigquery-public-data.crypto_ethereum.contracts` AS contracts
        JOIN `bigquery-public-data.crypto_ethereum.transactions` AS transactions ON
(transactions.to_address = contracts.address)
  WHERE contracts.is_erc20 = TRUE AND contracts.block_timestamp < '2021-09-01 00:00:00'
  GROUP BY contracts.address
  ORDER BY tx_count DESC
  LIMIT 10
```