Blockchain on Relational and NoSQL Databases

Vedant Chauhan (892758), Siddharth Malhotra (934336), and Saksham Agrawal ()

**Abstract**—Blockchain is a novel technique that chain blocks together as a public ledger on the nodes of peer-to-peer network. It is a tamper-resistance technique which technically is a database with no query capabilities. If blockchain is measured with database, it has low throughput, latency, and capacity. Traditional databases are centralized with greater capacity, higher throughput, and sub-second latency. This paper explores further about the underlying storage requirements associated with blockchain while exploring the current ecosystem of databases and form a comparative review of what would be the best architecture suited for blockchain.

**Index Terms**—Blockchain, NoSQL, Relational Databases

—————————— ◆ ——————————

# 1 Introduction

A

# 2 Background

## 2.1 Blockchain

A

## 2.2 Relational Databases

A

## 2.3 NoSQL Databases

A

# 3 Blockchain v/s database

The regular databases are centralized and use a client server architecture. Clearly, there is a fundamental difference between blockchain v/s the regular databases. Blockchain may well be related as a distributed database. Later, we shall see in the where does blockchain fit into the entire ecosystem of databases figuratively.

# 4 Blockchain with relational databases

## 4.1 Figures and Tables

A

## 4.2 Theorems and Proofs

A

|  |  |  |  |
| --- | --- | --- | --- |
|  | Blockchain | NoSQL | Blockchain with NoSQL |
| Immutability | O |  | O |
| No Central Authority | O |  | O |
| Assets over Network | O |  | O |
| High Throughput |  | O | O |
| Low Latency |  | O | O |
| High Capacity |  | O | O |
| Schema less |  | O | O |
| Integrated Caching |  | O | O |

# 5 Blockchain with nosql databases

## 5.1 Relation between blockchain and NoSQL

In earlier sections, we had detailed discussion about blockchain and NoSQL. But, there are few questions to answer before blockchain with NoSQL techniques provide better understanding about the relation of blockchain and NoSQL databases. First, **Why NoSQL is a better fit with blockchain?** Second, **How NoSQL can be changed to integrate with blockchain?**

First question can be answered with NoSQL advantages that elevate when integrated with blockchain.

1. Schema gives structure to data. Predefined templates can help in faster development. Traditional databases use this generic approach to handle data. Due to this predefined structure additional scenarios are hard to accommodate and permissioning is affected. But, NoSQL is schema less. It means it has a schema which can be key-pair like JSON, document, and graph. It usually helps in type of data that can be stored by blockchain.

2. Blockchain is a ledger which stores contracts, certificates, numbers, facts, etc. all categorized mostly towards unstructured data. NoSQL supports unstructured data, i.e. text-heavy, facts, numbers, etc. These type of data don't have a predefined structure. Traditional databases are not comfortable with such data. As, they focus more on systematic structure with rows and columns.

3. In modern applications, scaling and agility are some common challenges. Relational databases fail to cope with these challenges. Relational databases use scale-up architecture which increases load on the machine and makes adding servers expensive for the organization. But, NoSQL solve this problem of scaling. As the architecture of NoSQL allows horizontal scaling, i.e. sharding. Documents of the database are objects. These objects are key-pairs unlike rows and columns of relational. Therefore, they can be placed in part or complete on different nodes easily. This helps in handling the load easily and inexpensive.

4. In any database, querying is one of the fastest ways to know the expected result. One of the problem with blockchain is, it does not have query capabilities. NoSQL and relational databases provide querying capabilities, but NoSQL has huge processing power to iterate the results and produce the result instantly, however large is the database size.

Second question can be answered by tackling blockchain requirements and how those requirements can be framed using NoSQL.

1. One of the main features of blockchain is decentralization. Multiple nodes (anyone can become a miner) are interconnected with each other over peer-to-peer network performing proof-of-work to make a valid chain. Blockchain with NoSQL can provide voting permission to nodes on p2p network and operate over database’s consensus.

2. Another feature is tamper-resistance of blockchain. Once an information is stored in blockchain, it can’t be changed. Databases are not immutable. But, can be transformed in order to perform the activity. One such way is that sequence of blocks in the form of documents in database containing order of transactions are ordered. Each hash of the block is derived from data, transaction and previous hash, forming a chain of blocks.

3. Scalable capacity is important as chain is constantly growing after a transaction is performed by a miner. NoSQL as mentioned above, has more scaling power. Nodes can be easily added to increase the capacity. Hence, certificates and legal contracts will directly stored on blockchain database.

4. High performance is needed to mine the block and store the contents before any failure occurs. NoSQL already has high throughput and low latency which can help in optimization and computation of the process. Once these advantages integrates with blockchain, blockchain’s own throughput and latency in comparison to database elevates and provides fully-featured NoSQL query capability.

Both questions are significantly needed to be answered in order to check the relationship of blockchain with NoSQL databases. The questions gives a starting point about current understanding of the concept (NoSQL as a database for blockchain) and how the concept can be improved to enhance the needs of current blockchain technique.

## 5.2 Techniques

Blockchain with NoSQL Database is explored with the research in the area. The techniques provide how database is mined with blockchain to improve the current needs of the system. Initial technique emphasis on database with blockchain while other technique builds on top of it to improve security of the system. Following techniques provides clear understanding:

1. BigchainDB: A Scalable Blockchain Database [4]
2. It combines both blockchain and database resulting in database style decentralized storage. Decentralized control is obtained via cluster of nodes with voting permissions. This leads to a peer-to-peer network where permissioning is operated above database layer. This permissioning open doors for private blockchain databases to link with public blockchain databases.
3. Along with permission query capabilities to question the database is must for a database. BigChainDB provides the NoQL query capability to query the database what blockchain was lacking.
4. Scalability is an issue with traditional databases. BigchainDB emphasis on scaling. It combines NoSQL scaling advantages with blockchain to store certificates, legal contracts, and transactions on the database. It focuses on fractional replication, i.e. storing part of data on each node instead all data on one. This gives better scalable options.
5. Blockchain immutability is handled by sequence of blocks maintaining sequence of transactions. This ordering of blocks helps in validating the chain. If any unauthorized transactions is conducted, hash of the blocks will change and chain will be declared unvalidate. This operation is directly taken from the blockchain.
6. Creation and transferring of assets are improved with additional security implementation. Only entities which have permission can access the assets, any other compromised entity cannot change the data. This is called no single point-of-failure.
7. If number of nodes are increased in the chain, throughput increases to 1 million writes per second, and sub-second latency leading to petabytes of storage.
8. Due to all the above features BigchainDB can be applied to many use cases: directly storing legal contracts and certificates on blockchain database, supply chain management by tracking and creating high volume asset thereby reducing cost, fraud, and latency, intellectual property tracking, improving database reliability with no single point-of-failure, time stamping legal contracts reducing legal conflicts, and offers certificate of authenticity for security purposes.
9. With so many advantages BigchainDB is certainly a viable technique for combining blockchain and NoSQL databases. It presents usually all the advantages of blockchain and merges with the NoSQL database. The focal point was to improve the performance. But, this leads to some security compromises. One such disadvantage is to make concession on data integrity. If large number of miners are malicious, then consistency and accuracy of the system becomes critical. As, all the malicious miners will try to make the chain invalid by creating fake transactions. Normal miner will not be able to distinguish the valid transaction. Hence, next technique comes into picture which solves this major problem.
10. Blockchain based database to ensure data integrity [5]
11. Main focus is on data integrity in cloud computing environments. Due to low throughput, weak stability, and high latency of blockchain data integrity is compromised. As mentioned above this problem is solved by using two-layer blockchain based database as shown in the figure below.

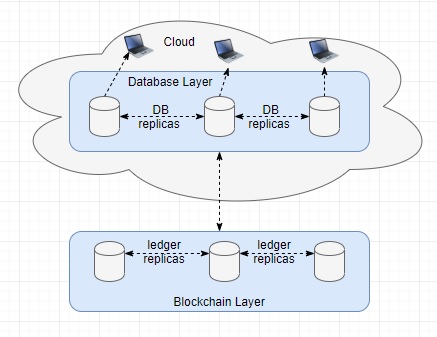


Fig. 5.2.1 Two-layer blockchain based database

1. First layer is database interface alongside blockchain. This layer stores every operations on the database. Due to the quick and reliable storage throughput is increased and latency is reduced. Each permissioned miner when executes the transaction stores the evidence on DB replicas. Private and public key pairs are used as a digital signatures of the messages. This layer has good performance but lacks data integrity due to no proof-of-work feature.
2. Second layer solves the problem of first layer. It is a proof-of-work based blockchain that stores part of database operation from first layer. Part of operation of first layer, i.e. hash of current operation are send to second layer for storage making it immutable. These hashes ensures the validity of the data stored in blockchain. This layer provides data integrity with weak performance.
3. Hence, both the layers play the crucial part of improving performance and ensuring data integrity by communicating with each other.
4. Data integrity provided by the technique is strong as Bitcoin’s, it takes approximately infinite time for the attacker to break the chain.
5. Bitcoin and Ethereum are permissionless blockchains. In these chains anybody can become a miner. These blockchains are adequate for the second layer of the model. Due to market fluctuations and mining mechanisms stability can be questioned. But, the stability is depended on application context. In this application, data integrity with performance is considered using database mainly non-relational database. Hence, can be a viable option.
6. This technique includes some disadvantages. When the operation happens in the first layer, all members are made aware that database operation is occurring. It is notification which malicious user can also get access.
7. Private keys can also be stealed by attackers in first layer while digital signature are applied in the first layer. The miner needs to be reactive while signing the messages.
8. Collusion attack aimed to change the information stored in first layer which was approved by all members makes the operation authenticated. If a valid member finds that the claim was wrong, then nothing can be done because in blockchain longest chain is the correct chain. Hence, this technique solves some problems but affected by problem of its own.

## 5.3 Comparison

Comparison between two techniques and overall relation between blockchain and NoSQL database is the ideal way to verify the claims made in this section.

Table 5.3.1 shows the comparison between BigchainDB and Blockchain based database. The table shows the current dynamics of both techniques.

|  |  |  |
| --- | --- | --- |
|  | BigchainDB | Blockchain based database ensuring data integrity |
| Layers | One- Both for blockchain and database | Two- One for database related operations and another for blockchain proof-of-work |
| Description | Database style decentralized storage | Data integrity in cloud environments |
| Goal | Emphasis on scale | Ensure data integrity |
| Concept | Practical | Theoretical (prototype in work) |
| Performance | High throughput and sub-second latency | Throughput and latency are improved (implementation in work) |
| Capacity | Petabytes of storage leads to increased capacity | Not mentioned |
| Security | As emphasis is more towards performance and scale, it suffers from data integrity problems | Solves data integrity problems but suffers from collusion attacks |

Table 5.3.1 Comparison between two mentioned techniques

The techniques have both advantages and disadvantages. BigchainDB provides the relation between blockchain and NoSQL databases, and blockchain based database builds on top of it to provide data integrity. From both techniques, BigchainDB is the technique which answers the goal we have mentioned in the abstract of this paper. This technique revamps the use of NoSQL database in blockchain and makes NoSQL a better contender for blockchain designs. Following comparison shows the relation of blockchain and NoSQL mentioned many times in this section and proves NoSQL is a better fit:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Blockchain | NoSQL | Blockchain with NoSQL |
| Immutability | O |  | O |
| No Central Authority | O |  | O |
| Assets over Network | O |  | O |
| High Throughput |  | O | O |
| Low Latency |  | O | O |
| High Capacity |  | O | O |
| Schema less |  | O | O |
| Integrated Caching |  | O | O |

Table 5.3.2 Comparison between blockchain and NoSQL

Table 5.3.2 shows the merits of blockchain, NoSQL, and their combination. Blockchain is immutable and decentralized, but has low throughput, latency, and capacity compared to NoSQL database. NoSQL on the other hand is schema less and supports integrated caching, i.e. in-memory storage which is better compared to relational database. As, BigchainDB suggested blockchain with NoSQL combines merits of both and provides a stable and efficient system with decentralization, tamper-resistance, high throughput, capacity, and query capabilities.

# 6 Conclusion

Collection of data and processing that data is constantly being refurbished with different technologies. With blockchain, there is security and immutability factor added while storing the data. Due to constant increase in the data and individual needs, just maintaining a public ledger with instability is not enough. Databases are always integral to store petabytes of data. In this paper we focus on relational and NoSQL databases to elevate current blockchain technique. Relational databases are schema-based centralized structure which provides high performance, but lacks in scaling, integrated caching, etc. NoSQL on the other hand is scalable, powerful and schema-less architecture. After comparing respective techniques based on these databases and blockchain, NoSQL provides the edge. It is debatable that which database technique is better with blockchain, but our research gives NoSQL an advantage. It is cost-effective to use NoSQL with blockchain and distributes multiple applications like supply chain, intellectual property, cryptocurrencies, legal contracts and certificates.

**References**

1. Blockchain.
2. Relational databases.
3. NoSQL databases.
4. McConaghy, Trent, Rodolphe Marques, Andreas Müller, Dimitri De Jonghe, Troy McConaghy, Greg McMullen, Ryan Henderson, Sylvain Bellemare, and Alberto Granzotto. "BigchainDB: a scalable blockchain database." *white paper, BigChainDB* (2016)
5. Gaetani, Edoardo, Leonardo Aniello, Roberto Baldoni, Federico Lombardi, Andrea Margheri, and Vladimiro Sassone. "Blockchain-based database to ensure data integrity in cloud computing environments." (2017)

[6]

[7]

[8]