Introduction to Indexers

Here you will find everything you need to know in order to familiarize yourself with the concept of indexers and even build your own one.

Disclaimer The explanation on this page assumes you have a certain level of understanding of the blockchain technology.

Blockchains and their nature

Blockchain data is optimized for serializedwrites , one block at a time, as the chain is being created. Querying the blockchain for data about a specific block or account is fairly straightforward or a "narrow" query. However, querying data across many blocks can be cumbersome because we have to aggregate results from multiple single-block queries. Therefore, we can consider these"wide" queries .

Given the fact that a blockchain itself is a distributed database, and a smart contract (decentralized application, dApp) is an application that runs on a virtual machine inside a blockchain, we need to understand that smart contracts shouldnot be considered as a "backend". While some applications might consist only of smart contracts, building a dApp with only smart contracts, in most cases, is not possible.

Smart contracts are limited in terms of interactions. By "interactions" we mean things that are very common in the real world, like user notifications, integration with third-party applications, etc.

However, the nature of a blockchain is that itmust be deterministic. A critical feature of a blockchain is that it knows the state at a given time, and for blockchains that time unit is a block. Think of them as being snapshots. A blockchain does snapshots of its state on every block. We as users can call smart contracts for a specific block, and the blockchain provides guarantees that execution will always produce the same result for the same block any time we call it.

The deterministic nature of a blockchain closes it from external (off-chain) variables. It is totally impossible to perform a call to an API from within a smart contract. A blockchain and a smart contract are closed off from the external (off-chain) world.

Blockchains are great at providing a way to apply the requested changes to the state in a decentralized manner. However, in order to observe the changes, you need to actively pull the information from the network.

Instead of abstract explanations let's look at an example.

Example dApp Say, we have a smart contract that sells e-books. Once a user buys a book we want to send them a copy via email. The dApp has a helper deployed somewhere off-chain, and this helper has code that can send an email with a copy of an e-book. But how would we trigger the helper?

Getting the data from a blockchain from the external world

NEAR blockchain implements aJSON-RPC endpoint for everyone to interact with the blockchain. Through the JSON-RPC API users can call smart contracts triggering them to be executed with given parameters. Also, users can view the data from the blockchain.

So, continuing with our example we can make our helper pull aBlock every second, then pull all theChunks and analyze the Transactions included in the Block to check if there is a transaction to our smart contract with "buy an e-book" function call. If we observe such a Transaction, we need to ensure it is successful, so we don't send the e-book to a user whose "buy e-book" Transaction failed.

After the process is complete we can trigger the helper's code to send the user an email with the e-book they bought.

This approach is so-calledpull model of getting the data. There is nothing wrong with this approach, but sometimes you might find it is not the most comfortable or reliable approach.

Also, not all the data is available through the JSON-RPC.Local Receipts for example are not available through the JSON-RPC, because they are not stored in NEAR node's internal database.

Indexer

A blockchain indexer is an implementation of thepush model of getting the data. Instead of actively pulling the data from the source, your helper waits for the data to be sent to it. The data is complete and so the helper can start analyzing it immediately; ideally the data is complete enough to avoid additional pulls to get more details.

Getting back to our example, the helper becomesan indexer that receives everyBlock , along withChunks ,Transactions with its statuses, etc. In the same way the helper analyzes the data and triggers the code to send the user an email with the e-book they bought.

An indexer concept An indexer listens to thestream of data as it's being written on chain and can then be immediately

filtered and processed to detect interesting events or patterns.

Indexers and "wide" queries

The term"wide" queries was mentioned in the beginning of this document. Here's a recap:

"Wide" queries definition To query data across many blocks requires the aggregation of results from multiple single-block queries. We can consider these aggregates as coming from"wide" queries . Because indexers listen to thestream of data from the blockchain and the data can be immediately filtered and processed according to defined requirements, they can be used to simplify the "wide" queries execution. For example, a stream of data can be written to a permanent database for later data analysis using a convenient query language like SQL. That is whatIndexer for Explorer is doing.

Another example that highlights the need for a "wide query" is when you use a seed phrase to recover one or more accounts. Since a seed phrase essentially represents a signing key pair, the recovery is for all accounts that share the associated public key. Therefore, when a seed phrase is used to recover an account viaNEAR Wallet , the query requires that all accounts with a matching public key are found and recovered.NEAR Indexer for Explorer is storing this data in a permanent database and this allowsNEAR Wallet to perform such "wide queries". This is impossible to achieve using JSON-RPC only.

Summary

We hope this article gives you an understanding of the Indexer concept. Also, we hope now you can easily decide whether you need an indexer for your application.

What's next?

We encourage you to learn more about theLake Indexer project . Please, proceed toTutorials section to learn how to build an indexer on practice.

Alternatively, there are a few other third-party indexers that are tightly integrated with the NEAR ecosystem. You can review all of your data sourcing options (including The Graph, Pagoda, Pipespeak, and SubQuery) underdata tools . Edit this page Last updatedonMar 14, 2023 byJames Bayly Was this page helpful? Yes No

Previous Data on the Blockchain Next Indexer Framework