BigQuery Public Dataset

Blockchain data indexing in NEAR Public Lakehouse is for anyone wanting to understand blockchain data. This includes:

- Users
- : create queries to track NEAR assets, monitor transactions, or analyze on-chain events at a massive scale.
- Researchers
- : use indexed data for data science tasks, including on-chain activities, identifying trends, or feeding AI/ML pipelines for predictive analysis.
- Startups
- : can use NEAR's indexed data for deep insights on user engagement, smart contract utilization, or insights across tokens and NFT adoption.

Benefits:

- NEAR instant insights
- : Historical on-chain data queried at scale.
- Cost-effective
- : eliminate the need to store and process bulk NEAR protocol data; query as little or as much data as preferred.
- Easy to use
- : no prior experience with blockchain technology is required; bring a general knowledge of SQL to unlock insights.

Getting started

1. Login into yourGoogle Cloud Account
2. .
3. Open theNEAR Protocol BigQuery Public Dataset
4. .
5. Click in theVIEW DATASET
6. button.
7. Click in the+
8. to create a new tab and write your query, click in theRUN
9. button, and check theQuery results
10. below the query.
11. Done :)

info TheNEAR Public Lakehouse repository contains the source code for ingesting NEAR Protocol data stored as JSON files in AWS S3 byNEAR Lake Indexer .

Example Queries

- How many unique signers and accounts have interacted with my smart contract per day?

SELECT ra . block_date collected_for_day , COUNT (DISTINCT t . signer_account_id)

as total_signers , COUNT (DISTINCT ra . receipt_predecessor_account_id)

as total_accounts FROM

bigquery-public-data.crypto_near_mainnet_us.receipt_actions ra JOIN

bigquery-public-data.crypto_near_mainnet_us.receipt_origin_transaction ro ON ro . receipt_id = ra . receipt_id JOIN

bigquery-public-data.crypto_near_mainnet_us.transactions t ON ro . originated_from_transaction_hash = t . transaction_hash WHERE ra . action_kind =

'FUNCTION_CALL' AND ra . receipt_receiver_account_id =

'social.near'

-- change to your contract GROUP

BY

1 ORDER

BY

1

DESC ;

How much it costs?

- NEAR pays for the storage and doesn't charge you to use the public dataset.To learn more about BigQuery public datasetscheck this page
- .
- Google GCP charges for the queries that you perform on the data. For example, in today's price "Sep 1st, 2023" the On-demand (per TB) query pricing is 6.25 per TB where the first 1 TB per month is free.CheckGoogle's pricing page
- for detailed pricing info, options, and best practices.

tip You can check how much data it will query before running it in the BigQuery console UI. Again, since BigQuery uses a columnar data structure and partitions, it's recommended to select only the columns and partitions (block_date) needed to avoid unnecessary query costs.

Architecture

The data is loaded in a streaming fashion usingDatabricks Autoloader into raw/bronze tables, and transformed withDatabricks Delta Live Tables streaming jobs into cleaned/enriched/silver tables.

The silver tables are also copied into theGCP BigQuery Public Dataset .

info Databricks Medallion Architecture .

Available Data

The current data that NEAR is providing was inspired byNEAR Indexer for Explorer .

info NEAR plans to improve the data available in the NEAR Public Lakehouse making it easier to consume by denormalizing some tables. The tables available in the NEAR Public Lakehouse are:

- blocks
- : A structure that represents an entire block in the NEAR blockchain.Block
- is the main entity in NEAR Protocol blockchain. Blocks are produced in NEAR Protocol every second.
- chunks
- : A structure that represents a chunk in the NEAR blockchain.Chunk
- of aBlock
- is a part of aBlock
- from aShard
- . The collection ofChunks
- of theBlock
- forms the NEAR Protocol Block.Chunk
- contains all the structures that make theBlock
- :Transactions
- ,Receipts
- , andChunk Header
- .
- transactions
- :Transaction
- is the main way of interaction between a user and a blockchain. Transaction contains: Signer account ID, Receiver account ID, and Actions.
- execution_outcomes
- : Execution outcome is the result of execution ofTransaction
- orReceipt
- . In the result of the Transaction execution will always be a Receipt.
- receipt_details
- : All cross-contract (we assume that each account lives in its own shard) communication in Near happens through Receipts. Receipts are stateful in a sense that they serve not only as messages between accounts but also can be stored in the account storage to awaitDataReceipts
- . Each receipt has apredecessor_id
- (who sent it) andreceiver_id
- the current account.
- receipt_origin
- : Tracks the transaction that originated the receipt.
- receipt_actions
- : Action Receipt represents a request to apply actions on thereceiver_id
- side. It could be derived as a result of aTransaction
- execution or anotherACTION

- Receipt processing. Action kind can be:ADD_KEY
- ,CREATE_ACCOUNT
- ,DELEGATE_ACTION
- ,DELETE_ACCOUNT
- ,DELETE_KEY
- ,DEPLOY_CONTRACT
- ,FUNCTION_CALL
- ,STAKE
- ,TRANSFER
- .
- receipts (view)
- : It's recommended to select only the columns and partitions (block_date
-) needed to avoid unnecessary query costs. This view join the receipt details, the transaction that originated the receipt and the receipt execution outcome.
- account_changes
- : Each account has an associated state where it stores its metadata and all the contract-related data (contract's code + storage).

References * Protocol documentation * Near Data flow * Lake Data structures * Protocol specification Edit this page Last updatedonMar 12, 2024 bys-n-park Was this page helpful? Yes No

Previous What is Data Infrastructure? Next Introduction