Create your first query

How to write a query to find out the monthly gas spent for your wallet

New to Dune? Embark on your analytical adventure with us! In this guide, we provide a detailed walkthrough to craft your inaugural query, revealing your wallet's monthly gas spent in USD.

Here is a visual click-through version if you are more of a visual learner. Below is a written version.

Getting Started

Initiate a New Query : Begin by selecting"Create New Query" . Start laying the basic structure of your query.

Copy SELECT FROM WHERE GROUP BY

Choosing the Right Table

1. Determine Your Data Source
2. : Instead of delving into the intricate raw blockchain tables to fetch gas expenditures for your wallet, utilize the pre-built, analytics-ready Spell table namedgas.fees
3. . This table facilitates an effortless aggregation of gas expenses across eight EVM chains specifically for your wallet.
4. What's the Spellbook?
5. Think of Spellbook as a transformative layer that aggregates and standardizes blockchain data, both raw and decoded. With thousands of models both contributed by our active community and curated by the Dune team, it's an invaluable resource. For a deeper dive into Spellbook, explore here.

Copy SELECT FROM gas. feesWHERE GROUP BY

Crafting Your Query

1. Identify Your Columns
2. : Once you've zeroed in on the right table, pinpoint thetx_fee_usd
3. column to track USD gas expenditures. To specify the wallet you're examining, use thetx_sender
4. field. For demonstration purposes, we'll be using Vitalik's wallet (0xd8dA6BF26964aF9D7eEd9e03E53415D37aA96045).
5. Aggregation & Time Period
6. : Our goal is to get monthly gas spendings, so we need to truncate theblock_date
7. to month and applying theSUM()
8. function, then group by each month go derive the results.

Sum of sum was applied to get the cumulatively gas spendings.

Copy SELECT DATE_TRUNC('month' , block_date) AS period, SUM (tx_fee_usd) AS monthly_gas_spent, SUM (SUM (tx_fee_usd)) OVER (ORDER BY DATE_TRUNC('month' , block_date)) AS cumulative_gas_spentFROM gas. feesWHERE tx_sender= 0xd8dA6BF26964aF9D7eEd9e03E53415D37aA96045 -- vitalik.eth GROUP BY 1 ORDER BY 1
Here is thelink to demo query .

Boost Flexibility with Parameters

In the example, you'll seetx_sender = {{Your Wallet Address}} -- default is vitalik.eth . This denotes the use of the parameter feature, enhancing the query's flexibility. To mark a field as a parameter, wrap it in "". Alternatively, activate the "Add Parameter" button.

Copy SELECT DATE_TRUNC('month' , block_date) AS period, SUM (tx_fee_usd) AS monthly_gas_spent, SUM (SUM (tx_fee_usd)) OVER (ORDER BY DATE_TRUNC('month' , block_date)) AS cumulative_gas_spentFROM gas. feesWHERE tx_sender= {{Your Wallet Address}}-- default is vitalik.eth GROUP BY DATE_TRUNC('month' , block_date) ORDER BY 1

Execute & Fetch the Result

All set! Click “Run” and behold the outcome. Kudos on crafting your first query!

Now that you have successfully run your query, you can go ahead and save it so you can keep monitoring your gas usage going forward. To save, clickSave in the upper right part of the screen.

For a more intuitive insight, consider visualizing this data. Navigate toCreate your first visualization and conjure up some graphical magic!

Was this page helpful?

Yes No Raise issue Web3 Data Resources Create AI-Assisted Queries linkedin github twitter discord telegram youtube Powered by Mintlify