Epoch

Operation of the Sui network is temporally partitioned into non-overlapping epochs, and the network aims to keep epochs roughly the same duration as each other. During a particular epoch the following data is fixed:

The epoch's id as a sequence number that starts at 0 and is incremented by one at every epoch change.

The minimum gas price that a quorum of validators are guaranteed to sign a transaction for.

Validator related properties, including the active validators.

The epoch's starting timestamp.

The epoch's ending timestamp.

The total number of checkpoints in this epoch.

The total number of transaction blocks in this epoch.

The total amount of gas fees (in MIST) that were paid in this epoch.

The total MIST rewarded as stake.

The amount added to total gas fees to make up the total stake rewards.

The storage fund available in this epoch. This fund is used to redistribute storage fees from past transactions to future validators.

The difference between the fund inflow and outflow, representing the net amount of storage fees accumulated in this epoch.

The storage fees paid for transactions executed during the epoch.

The storage fee rebates paid to users who deleted the data associated with past transactions.

The epoch's corresponding protocol configuration, including the feature flags and the configuration options.

SUI set aside to account for objects stored on-chain, at the start of the epoch. This is also used for storage rebates.

Information about whether this epoch was started in safe mode, which happens if the full epoch change logic fails for some reason.

The value of the version field of 0x5 , the 0x3::sui::SuiSystemState object. This version changes whenever the fields contained in the system state object (held in a dynamic field attached to 0x5) change.

Details of the system that are decided during genesis.

Parameters related to the subsidy that supplements staking rewards

A commitment by the committee at the end of epoch on the contents of the live object set at that time. This can be used to verify state snapshots.

The epoch's corresponding checkpoints.

The epoch's corresponding transaction blocks.

scanLimit restricts the number of candidate transactions scanned when gathering a page of results. It is required for queries that apply more than two complex filters (on function, kind, sender, recipient, input object, changed object, or ids), and can be at most serviceConfig.maxScanLimit .

When the scan limit is reached the page will be returned even if it has fewer than first results when paginating forward (last when paginating backwards). If there are more transactions to scan, pageInfo.hasNextPage (or pageInfo.hasPreviousPage) will be set to true , and PageInfo.endCursor (or PageInfo.startCursor) will be set to the last transaction that was scanned as opposed to the last (or first) transaction in the page.

Requesting the next (or previous) page after this cursor will resume the search, scanning the next scanLimit many transactions in the direction of pagination, and so on until all transactions in the scanning range have been visited.

By default, the scanning range consists of all transactions in this epoch.

epoch query

ActiveJwk object ● AuthenticatorStateExpireTransaction object ● AuthenticatorStateUpdateTransaction object ● ChangeEpochTransaction object ● Checkpoint object ● ConsensusCommitPrologueTransaction object ● EpochConnection object ● EpochEdge object ● RandomnessStateUpdateTransaction object ● StakedSui object ● TransactionBlock object ● TransactionBlockEffects object