Client Module

This module provides a function to get the data from Bandchain gRPC and send the data to Bandchain gRPC.

Note: Get the here

get_data_source(id)

This function returns data source details of the given ID.

Parameter

- id
-
- : Data source ID

Return

- oracle_type.DataSource

Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

id

=

1

c

Client (grpc_url) data_source = c . get_data_source (id) print (MessageToJson (data_source)) Result

{ "owner" :

"band1jfdmjkxs3hvddsf4ef2wmsmte3s5llqhxqgcfe" , "name" :

"DS1" , "description" :

"TBD" , "filename" :

"32ee6262d4a615f2c3ca0589c1c1af79212f24823453cb3f4cfff85b8d338045" , "treasury" :

"band1jfdmjkxs3hvddsf4ef2wmsmte3s5llqhxqgcfe" }

get_oracle_script(id)

This function returns oracle script details of the given ID.

Parameter

- id
-
- : Oracle Script ID

Return

- oracle_type.OracleScript

Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

id

=

1

c

Client (grpc_url) oracle_script = c . get_oracle_script (id) print (MessageToJson (oracle_script)) Result

{ "owner" :

"band1jfdmjkxs3hvddsf4ef2wmsmte3s5llqhxqgcfe" , "name" :

"OS1" , "description" :

"TBD" , "filename" :

"f86b37dbe62c3b8c86ae28523bf09e9963a6b2951dd1a5be79f29f66d8236abf" , "schema" :

"{gas_option:string}/{gweix10:u64}" }

get_request_by_id(id)

This function returns request details of the given ID.

Parameter

- id

-
- : Request ID

Return

- oracle_query.QueryRequestResponse

Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

id

=

1

c

Client (grpc_url) request = c . get_request_by_id (id) print (MessageToJson (request)) Result

{ "result" :

{ "clientId" :

"from_bandd" , "oracleScriptId" :

"37" , "calldata" :

"AAAABgAAAANCVEMAAAADRVRIAAAAA01JUgAAAANBTkMAAAAERE9HRQAAAARMVU5BAAAAADuaygA=" , "askCount" :

"1" , "minCount" :

"1" , "requestId" :

"1" , "ansCount" :

"1" , "requestTime" :

"1624374833" , "resolveTime" :

"1624374844" , "resolveStatus" :

"RESOLVE_STATUS_SUCCESS" , "result" :

"AAAABgAAHBpu4YHAAAABqf4l8EAAAAABFPDhkAAAAACK0bhAAAAAAAsYHsgAAAABLZl5AA==" } }

get_reporters(validator)

This function returns a list of reporters associated with the given validator.

Parameter

- validator
-
- : Validator address

Return

- List

Example

from pyband import Client

grpc_url

""

without https://

validator

"bandvaloper1p46uhvdk8vr829v747v85hst3mur2dzlhfemmz"

c

Client (grpc_url) print (c . get_reporters (validator)) Result

["band1p46uhvdk8vr829v747v85hst3mur2dzlmlac7f", "band1zgly2mgx7ykckfgm4dqgc58vrntdlgcemy62eq", "band1sfmc6995mk0d55zy2vy8dxu8s54y5e7yqquxkr", "band1jtjuucr5wea43up4d3d98e3xn737lry800j6tf", "band1jd95fjm3j43pqurc2k4suzmznhux85hsjrx0a8", "band1l5kxfkd7gvvtpd37gmjnpvd0suzwypgz5u9ysc"]

get_latest_block

This function returns the latest block in the chain.

Return

- tendermint_query.GetLatestBlockResponse

Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

c

Client (grpc_url) latest_block = c . get_latest_block ( ) print (MessageToJson (latest_block)) Result

{ "blockId" :

{ "hash" :

"FHpPvUlEF2WxXGGkrN9Lc4pI5oj/3Z5/U1UvLXsX/z0=" , "partSetHeader" :

{ "total" :

1 , "hash" :

"wgzw1SgKKd+uZwKSHWONJ6qNigdwOPhqW2nTq3AYKv0=" } } , "block" :

{ "header" :

{ "version" :

{

"block" :

"11"

} , "chainId" :

"band-laozi-testnet2" , "height" :

"603404" , "time" :

"2021-07-12T08:05:22.386235207Z" , "lastBlockId" :

{ "hash" :

"XMfyuM/nZZcoWVCQG+SQ93zaYuVt47i/ISlgi3KJqJw=" , "partSetHeader" :

{ "total" :

1 , "hash" :

"u4XfI/858RuZExpK0D1mtQzC90R7fbhBAyIboqRQcoI=" } } , "lastCommitHash" :

"caU5MlsAHEpy29PzMDQJ0OdIGkCfIyttQLQoTrVNpfA=" , "dataHash" :

"TyORhRUl3QS/stvNbHsld2uU47aOMhbHfbWbmzCgt9s=" , "validatorsHash" :

"e0hw8Ieib1SLF87P75KUsV/Zh4UZDZocxtN13v+temM=" , "nextValidatorsHash" :

"e0hw8Ieib1SLF87P75KUsV/Zh4UZDZocxtN13v+temM=" , "consensusHash" :

"ek5k0qm1ziK3XpVuICUnTcA7aEbM13JRUqa8DQcn4z4=" , "appHash" :

"VPpwYxRcdOtU5OclkZ+W1zuVCgo+P5TOkqwWXYY/Stw=" , "lastResultsHash" :

"RTYNyG6gfZr3/J5OPnmv49+qR5wQmUUFypdaoZDT188=" , "evidenceHash" :

"47DEQpj8HBSa+/TlmW+5JCeuQeRkm5NMpJWZG3hSuFU=" , "proposerAddress" :

"Mi9ClpvJaLKLAUv/2dg6nGeAZGw=" } , "data" :

{ "txs" :

[

"CuYCCrYCChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESmQIIot0SEg8IDBoLMC4wMDA4NzM5NAoSDQgKGgkwLjAwMDg3MAoSDwgNGgswLjAwMDg2NjAwChISCAUaDjk2OTUuMCw5NjYwLjAKEhI

,

"CqMECvMDChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES1gMIn90SEl8IAhABGlk0MjkgQ2xpZW50IEVycm9yOiBUb28gTWFueSBSZXF1ZXN0cyBmb3IgdXJsOiBodHRwczovL2FwaS5jb2luZ2Vja28uY2

,

"CvACCsACChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESowIIoN0SEl8IAhABGlk0MjkgQ2xpZW50IEVycm9yOiBUb28gTWFueSBSZXF1ZXN0cyBmb3IgdXJsOiBodHRwczovL2FwaS5jb2luZ2Vja28uY29

,

"CvACCsACChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESowIIoN0SEl8IAhABGlk0MjkgQ2xpZW50IEVycm9yOiBUb28gTWFueSBSZXF1ZXN0cyBmb3IgdXJsOiBodHRwczovL2FwaS5jb2luZ2Vja28uY29

,

"CtABCs0BChkvb3JhY2xlLnYxLk1zZ1JlcXVlc3REYXRhEq8BCCUSWAAAAAoAAAAEVVNEVAAAAANCU1YAAAADWE1SAAAABFVTREMAAAAETEVOORAAAAAREQVNIAAAAA1pFQwAAAANFVEM

,

"CqcBCqQBChkvb3JhY2xlLnYxLk1zZ1JlcXVlc3REYXRhEoYBCCUSOAAAAAYAAAADQlRDAAAAA0VUSAAAAANNSVIAAAADQU5DAAAABERPR0UAAAAETFVOQQAAAAA7msoAGAYgAyoPbWlycr

,

"CoABCn4KGS9vcmFjbGUudjEuTXNnUmVxdWVzdERhdGESYQgrEhwAAAACAAAABEJUQ0IAAAAEQkVUSAAAAAA7msoAGAYgAyoGbGluZWFyOMCaDEDAhD1KK2JhbmQxODJ3cHg0ODd3cXVw

,

"CpoCCpcCChkvb3JhY2xlLnYxLk1zZ1JlcXVlc3REYXRhEvkBCCwSqgEAAAAVAAAABEFBUEwAAAAFR09PR0wAAAAEVFNMQQAAAARORkxYAAAAA1FRUQAAAARUV1RSAAAABEJBQkEAAAADS

,

"CsYBCsMBChkvb3JhY2xlLnYxLk1zZ1JlcXVlc3REYXRhEqUBCCUSTgAAAAkAAAADTVlCAAAABE5QWFMAAAADT1NUAAAAA1BBWQAAAARQQlRDAAAAA1BMUgAAAARQTFRDAAAAA1BOSwA

,

"CpEECvADChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES0wMIpN0SEkIIAho+MC45OTg2MjksMTQwLjgsMjEzLjc2LDEuMCwzLjE5LDEzMy4xNCwxMTEuNzEsNTAuMDcsMTQuMzksNS42MgoSNAgBC

,

"CpAECu8DChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES0gMIpd0SEhUIBhoRMzQyODEuMDcsMjE0NC40NAoSFhoUMzQyODQuMDU1NCwyMTM1LjkwMgoSPQgDGjkzNDI1Ny4xMTQsMjE0My40MD

,

"Co8ECu4DChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES0QMIpd0SEhUIBhoRMzQyODEuMDcsMjE0NC40NAoSJggBGiIzNDI2Ny4zNCwyMTQzLjA0LDIuMTIsMC4yMTcyLDguMjUKEj0IAxo5MzQyNTc

,

"CvEDCtADChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESswMIn90SEj0IAho5OS45LDAuODE4ODY1LDEzLjA5LDQzMy45MSw0LjIyLDI2OTAuMDIsMjEwLjg3LDAuMDQxOTc0NzMKEIAIAxpMOS44NzA

,

"CoYDCsoCChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESrQIIo90SEhcIoDgaEjkwMi4zMjM1MjgzMzM1NDU2ChIXCKM4GhI5MDIuMzIzNTI4MzMzNTQ1NgoSFwiiOBoSOTAyLjMyMTY4Mzg3NTU1NzEK

,

"CvkCCskCChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESrAIIo90SEhcIozgaEjkwMi4zMjM1MjgzMzM1NDU2ChIXCKA4GhI5MDIuMzIzNTI4MzMzNTQ1NgoSFwihOBoSOTAyLjMyMzUyODMzMzU0NTYK

,

"CsQCCpQCChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES9wEIp90SEiMIARofMzQyNjcuMzM5OTk5OTk5IDIxNDMuMTM5OTk5OTk5ChIjCAIaHzM0MjY3LjMzOTk5OTk5OSAyMTQzLjEzOTk5OTk5OQ

,

"Co0ECuwDChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESzwMIpd0SEj0IAxo5MzQyNTcuMTE0LDIxNDMuNDA1OSwzLjY5Nzg4NSwyLjEyMjA2NywwLjIxNzIyNTY2LDguMjUyNTgKEhYaFDM0Mjg0LjA1M

,

"CogFCpQCChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES9wEIp90SEiEaHzM0MjY3LjMzOTk5OTk5OSAyMTQzLjEzOTk5OTk5OQoSIwgBGh8zNDI2Ny4zMzk5OTk5OTkgMjE0My4xMzk5OTk5OTkKEiM

,

"CvEDCrUDChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGESmAMIpt0SEpQBCAEajwExNDUuMDM1LDI1MDYuODg1LDY1OS4wNSw1MzYuMjM1LDM2MC44NSw2OC43MSwyMDMuNTQ1LDM0LjM2NSw

,

"CpEECvADChgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES0wMIpN0SEkIIAho+MC45OTg2MjksMTQwLjgsMjEzLjc2LDEuMCwzLjE5LDEzMy4xNCwxMTEuNzEsNTAuMDcsMTQuMzksNS42MgoSCxoJM

,

"CskECo0ECihgvb3JhY2xlLnYxLk1zZ1JlcG9ydERhdGES8AMIpN0SEjQIARowMSwxNDAuNywyMTMuNjIsMSwxLjU0NiwxMzIuOSwxMTEuOTksMTQuMjksNS42MzUKEl8IAhABGlk0MjkgQ2xpZW50IEV
]] , "evidence" :

{ } , "lastCommit" :

{ "height" :

"603403" , "blockId" :

{ "hash" :

"XMfyuM/nZZcoWVCQG+SQ93zaYuVt47i/ISlgi3KJqJw=" , "partSetHeader" :

{ "total" :

1 , "hash" :

"u4Xfl/858RuZExpK0D1mtQzC90R7fbhBAyIboqRQcoI=" } } , "signatures" :

[{ "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"Zdyy3QL8E8XZYhDrA8fAUD4m2Jc=" , "timestamp" :

"2021-07-12T08:05:22.304030411Z" , "signature" :

"ISRrl36Kv2m6vOv7d2OpoOQtR6fmeodsg/rxGKXfch5IutvFw4/p8/LMDg3zUT64raHz8YW2aXid88BdZNUp6g==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"xLnySmLJL6Qq4ebq+oPMs+KErU=" , "timestamp" :

"2021-07-12T08:05:22.435650954Z" , "signature" :

"fxFAi9FQEe98Dm7cv8psNwWdQg1B/h6RaElrfKuPwBhfiN80WE36+ioNKK7BIvfOx8QuBBuNQ+ic6mMF+AcUZA==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"D3OpjoewGqrIf2g+qADd6sKpM24=" , "timestamp" :

"2021-07-12T08:05:22.395247997Z" , "signature" :

"Ulku3yfZDtUxXdidUwlClrKAgns002z4suYZSef+rdkPxVxmCZayA4wwMUWVNnTfEdkLzsSkl6TVBSfKkfc5/g==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"KLbUBDhTpSjBO2cco4cmyhJiR7w=" , "timestamp" :

"2021-07-12T08:05:22.454990956Z" , "signature" :

"pf/usYgYQrfmcIqul9YxUbu8Ruajit0L2HXU0xz8ayg1SBZqfJcu3l7tO77iAw76S5PMZHBvFO+cig3YC3nXSA==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"biAZveTotPyzzCPfYXiWCcKDKr8=" , "timestamp" :

"2021-07-12T08:05:22.302760757Z" , "signature" :

"dSf7bobDcdvvJdKKIK9CbIGGiASyBDhis1fMEivpXmozjvAAMRcBtxoekLMzNfbu7ZhXZkIFH7DzoVsjSdWubw==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"AFmBCh4x3+WV/oQfZTcQ4a97ms0=" , "timestamp" :

"2021-07-12T08:05:22.303799803Z" , "signature" :

"Nrwb1ztSW27fEqzfgJ6G4lUqecILKRGowezkIOmX4l4OZNpJfihHuh7AHoPhj4SLkeNBud1vlW96zmruqbNqFQ==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"GXmy0WdxkkwltyFvLJbIa9v5Kt8=" , "timestamp" :

"2021-07-12T08:05:22.409681572Z" , "signature" :

"HF5CZDNMKih/fe35Cwpqa5Elw2s1oTT87iVMRL5Fo51keUOB+Ly21SguFb6SeXusJ3W6KF+4YjtMM7OWabz5sA==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"HW6680aDSSb7WNZU6GKkL1PYgiA=" , "timestamp" :

"2021-07-12T08:05:22.394570566Z" , "signature" :

"t1brNDe7aCu3Q13ePI8Lsp6Jwl8nTjKBvnRBOtdvbcUcSIy4WbadIbnXou8bXdoKnXeEZCqf7zsULZtKz8QcEg==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"Mi9CIpvJaLKLAUv/2dg6nGeAZGw=" , "timestamp" :

"2021-07-12T08:05:22.304367649Z" , "signature" :

"w9nyw1GsE4EVViKnLuTHGHdcCUBA7EoetN6ztVHdqs4moh82yrhLxtJjoRgQy54G8S8Qzg7BS8dJuwpPkuO/1A==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"TO8fbCX9iAPgPb7NP60N5c+1XCw=" , "timestamp" :

"2021-07-12T08:05:22.478276190Z" , "signature" :

"UJ5YrJDIR55lEm9iwTtaodaKz38aTJC9ba4NoULsh0cZKWRMdXSsEo62gQd4XWZesRU9VbpwET0kr7aUnOTY1Q==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"UX8oPBDTFGmRZsWIhGKc2EUE86Q=" , "timestamp" :

"2021-07-12T08:05:22.412614197Z" , "signature" :

"KfxPKErTsEtpAKtSRE9XlfekK0XSUEdka5B+/i7OXGt0P2d7T7rztBKqYSASuPaAvCicTgEvWIhyNaC2DzzmjA==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"aAyok6Qr/EGfdJr8DiG9Yycw4VA=" , "timestamp" :

"2021-07-12T08:05:22.502941178Z" , "signature" :

"Z9S74/SM6J5Hvqce4i7KlrXZwh6kG39pU5jUs8NVlFZvye+s4lnqR0vxQafLi8apyk2UdNW4KWLDUXA8KjZqAA==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"adhSwssyskWcBBWKzyItym8rK8U=" , "timestamp" :

"2021-07-12T08:05:22.330196188Z" , "signature" :

"qah8/op1dvTB8CaPW/TPy89nJkUw7NEn7ZkgkQ/r6eA8bwT/+QJz2qHlUNxIZ/7KE6LsqDiy+Ss4xsZPkbpH2A==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"tWg//Wbq1Rj5b0nPLi1yZE794FI=" , "timestamp" :

"2021-07-12T08:05:22.386235207Z" , "signature" :

"SrQW8WDsO0USkDZmRYNDzCXLj/y4AEWCLJ3JzU12LOIFBs+uIRZ5dHDx2jsBytYOmDdXsh1udKiRrkBqa4jXuw==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"056lMQcoc6aX+fZODSBBYOPz5DY=" , "timestamp" :

"2021-07-12T08:05:22.402139725Z" , "signature" :

"RF65UL0Hyd/0Z7aAPXqpOhcc0TaqZ6l4BN7IXOMNjfANXycZYgiNZvbppumWwsgI9AeAv/+6xPKXAG/ghytl0w==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"2MqI5T0aGLK0h3fNWAcQEKLplio=" , "timestamp" :

"2021-07-12T08:05:22.416599751Z" , "signature" :

"stsHa0JHU1ipj8ahGdNNxm/FrYGyAGYh16ry8tsC0kx7ymBhj3uvmu9CQpBMdb7bTQNOuOTLRKLaln+8waUumw==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"299xvCm2ZUwTV/Tcty4zcOBT1AU=" , "timestamp" :

"2021-07-12T08:05:22.356259712Z" , "signature" :

"VX+TAitw0TeG25vdAyRTTqFD92QCMLyCFDfhzA8MmNQDAXLDF79l4sQGkQFnuAOYvAANKZtGdXdx+1nYX/AR0A==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"+nva6pxhpfHNdpnVtUg9kpTft8s=" , "timestamp" :

"2021-07-12T08:05:22.400389951Z" , "signature" :

"haUPT+DIwnXFU+zxwZukakxLU/Vzz3tpwel1MA9tgN4YNLsK1DOKJjjs4U5Zj4qvYXLH5tvGUXz2XhNPb/7sJA==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"lZEVu5kiC4YXL2FxwaAMRv282MQ=" , "timestamp" :

"2021-07-12T08:05:22.393823192Z" , "signature" :

"QuhuFlfGNzHzoAijw/PrFRUR7TmaaphyI/ybpFTpG8JxNGcmUZ+YtNJtzJwRhUqkhV/i5Ngv3Xx/uQX0Z0RaCg==" } , { "blockIdFlag" :

"BLOCK_ID_FLAG_COMMIT" , "validatorAddress" :

"a0hzz+h4moaQrE64yXSjzRWM+p4=" , "timestamp" :

"2021-07-12T08:05:22.389074186Z" , "signature" :

"8XggPj2IR9uWseeZKtejNwrkBUWcQDdThz4CcGH+X1l2RuFlZq/xIP2v0j2tw0Et3aFzOyMdMHl9N1/KUCXWBQ==" }] } } }

get_account(address)

This function returns the account details of the specified address.

Parameter

- address
-

Return

- Optional[auth_type.BaseAccount
-]

Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

address

"band1ee656yzw6y9swqayu9v0kgu5pua2kgjq3hd6g3"

c

Client (grpc_url) addr = c . get_account (address) print (MessageToJson (addr)) Result

{ "address" :

"band1ee656yzw6y9swqayu9v0kgu5pua2kgjq3hd6g3" , "pubKey" :

{ "@type" :

"/cosmos.crypto.secp256k1.PubKey" , "key" :

"AsBzzfeupPh2IM9xJ7SnhtIl7kVGX2QoY3Ro2DRKsmIF" } , "accountNumber" :

"171" , "sequence" :

"10" }

get_request_id_by_tx_hash(tx_hash)

This function returns request ID of the given transaction hash.

Parameter

- tx_hash
-
- : Transaction hash

Return

- List[int]

Exception

Type Description NotFoundError Request Id is not found Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

tx_hash

"DCC09AD0087DFB30AD552DAFA6C52FE9676F157B24812FF4B9994B97CAC914AC"

c

Client (grpc_url) print (c . get_request_id_by_tx_hash (tx_hash)) Result

[37625, 37626, 37627, 37628, 37629]

get_chain_id

This function returns a chain ID.

Return

-

Example

from pyband import Client

grpc_url

""

without https://

c

Client (grpc_url) print (c . get_chain_id ( )) Result

band-laozi-testnet2

get_reference_data(pairs, min_count, ask_count)

This function returns the rates of the given cryptocurrency pairs.

Parameter

- pairs
- List
- List of cryptocurrency pairs.
- min_count
-
- Minimum number of validators necessary for the request to proceed to the execution phase.
- ask_count
-
- : Number of validators that are requested to response to the corresponding request.

Return

- List[ReferencePrice
-]

Exception

Type Description EmptyMsgError Pairs are required Example

from pyband import Client

grpc_url

""

without https://

client

Client (grpc_url)

min_count

3 ask_count =

4

pairs

["BTC/USD" ,

"ETH/USD"]

print (client . get_reference_data (pairs , min_count , ask_count)) Result

[ReferencePrice((pair = "BTC/USD") , (rate = 33373.93) , (updated_at = ReferencePriceUpdated((base = 1625715297) , (quote = 1625715749)))) , ReferencePrice((pair = "ETH/USD") , (rate = 2261.97) , (updated_at = ReferencePriceUpdated((base = 1625715297) , (quote = 1625715749))))]

get_latest_request(oid, calldata, min_count, ask_count)

This function returns the latest request.

Parameter

- oid
-
- : Oracle script ID
- calldata
-
- : Calldata of a request.
- min_count
-
- : Minimum number of validators necessary for the request to proceed to the execution phase.
- ask_count
-
- : Number of validators that are requested to response to the corresponding request.

Return

- oracle_query.QueryRequestSearchResponse

Example

from pyband import Client from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

c

Client (grpc_url)

oid

43 calldata =

"000000020000000442544342000000044245544800000003b9aca00" min_count =

3 ask_count =

4

latest_req

c . get_latest_request (oid , calldata , min_count , ask_count) print (MessageToJson (latest_req)) Result

{ "request" :

{ "request" :

{ "oracleScriptId" :

"43" , "calldata" :

"AAAAAgAAAARCVENCAAAABEJFVEgAAAAAO5rKAA==" , "requestedValidators" :

["bandvaloper1p46uhvdk8vr829v747v85hst3mur2dzlhfemmz" , "bandvaloper17n5rmujk78nkgss7tjecg4nfzn6geg4cqtyg3u" , "bandvaloper1e9sa38742tzhmandc4gkqve9zy8zc0yremaa3j" , "bandvaloper1zl5925n5u24njn9axpygz8lhjl5a8v4cpkzx5g" , "bandvaloper1ldtwjzsplhxzhrg3k5hhr8v0qterv05vpdxp9f" , "bandvaloper19eu9g3gka6rxlevkjlvjq7s6c498tejnwxjwxx"] , "minCount" :

"3" , "requestHeight" :

"603449" , "requestTime" :

"1626077260" , "clientId" :

"linear" , "rawRequests" :

[{ "dataSourceId" :

"74" , "calldata" :

"aHR0cHM6Ly91cy1ycGMuYmFuZGNoYWluLm9yZy9vcmFjbGUvcmVxdWVzdF9wcmljZXMgQlRDIEVUSA==" } , { "externalId" :

"1" , "dataSourceId" :

"74" , "calldata" :

"aHR0cHM6Ly9ldS1ycGMuYmFuZGNoYWluLm9yZy9vcmFjbGUvcmVxdWVzdF9wcmljZXMgQlRDIEVUSA==" } , { "externalId" :

"2" , "dataSourceId" :

"74" , "calldata" :

"aHR0cHM6Ly9hc2lhLXJwYy5iYW5kY2hhaW4ub3JnL29yYWNsZS9yZXF1ZXN0X3ByaWNlcyBCVEMgRVRI" } , { "externalId" :

"3" , "dataSourceId" :

"74" , "calldata" :

"aHR0cHM6Ly9hdXMtcnBjLmJhbmRjaGFpbi5vcmcvb3JhY2xlL3JlcXVlc3RfcHJpY2VzIEJUQyBFVEg=" }] , "executeGas" :

"1000000" } , "reports" :

[{ "validator" :

"bandvaloper1p46uhvdk8vr829v747v85hst3mur2dzlhfemmz" , "inBeforeResolve" :

true , "rawReports" :

[{

"externalId" :
"1" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"2" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"3" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
}] } , { "validator" :
"bandvaloper1zl5925n5u24njn9axpygz8lhjl5a8v4cpkzx5g" , "inBeforeResolve" :
true , "rawReports" :
[{
"externalId" :
"2" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"3" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"1" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
}] } , { "validator" :
"bandvaloper19eu9g3gka6rxlevkjlvjq7s6c498tejnwxjwxx" , "inBeforeResolve" :
true , "rawReports" :
[{
"externalId" :
"3" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"2" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"1" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
}] } , { "validator" :

"bandvaloper1e9sa38742tzhmandc4gkqve9zy8zc0yremaa3j" , "inBeforeResolve" :
true , "rawReports" :
[{
"externalId" :
"3" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"2" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"1" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
}] } , { "validator" :
"bandvaloper17n5rmujk78nkgss7tjecg4nfzn6geg4cqtyg3u" , "inBeforeResolve" :
true , "rawReports" :
[{
"externalId" :
"2" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"3" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"1" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
}] } , { "validator" :
"bandvaloper1ldtwjzsplhxzhrg3k5hhr8v0qterv05vpdxp9f" , "inBeforeResolve" :
true , "rawReports" :
[{
"externalId" :
"3" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"1" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {
"externalId" :
"2" ,
"data" :
"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"
} , {

"data" :

"MzQyNzkuOTYgMjE0NC43MTk5OTk5OTkK"

}] }] , "result" :

{ "clientId" :

"linear" , "oracleScriptId" :

"43" , "calldata" :

"AAAAAgAAAARCVENCAAAABEJFVEgAAAAAO5rKAA==" , "askCount" :

"6" , "minCount" :

"3" , "requestId" :

"306920" , "ansCount" :

"6" , "requestTime" :

"1626077260" , "resolveTime" :

"1626077266" , "resolveStatus" :

"RESOLVE_STATUS_SUCCESS" , "result" :

"AAAAAgAAHy1s1rYAAAAB81tGE/4=" } } }

send_tx_sync_mode(tx_bytes)

This function sends a transaction in sync mode, that is, send and wait until a transaction has passed CheckTx phase.

Parameter

- tx_bytes
- : Transaction raw bytes that is already signed.

Return

- abci_type.TxResponse

Example

import os

from pyband . client import Client from pyband . transaction import Transaction from pyband . wallet import PrivateKey

from pyband . proto . cosmos . base . v1beta1 . coin_pb2 import Coin from pyband . proto . oracle . v1 . tx_pb2 import MsgRequestData from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

c

Client (grpc_url)

MNEMONIC

os . getenv ("MNEMONIC") private_key = PrivateKey . from_mnemonic (MNEMONIC) public_key = private_key . to_public_key ( ) sender_addr = public_key . to_address ( ) sender = sender_addr . to_acc_bech32 ( )

request_msg

MsgRequestData (oracle_script_id = 37 , calldata = bytes . fromhex ("0000000200000003425443000000034554480000000000000064") , ask_count = 4 , min_count = 3 , client_id = "BandProtocol" , fee_limit = [Coin (amount = "100" , denom = "uband")] , prepare_gas = 50000 , execute_gas = 200000 , sender = sender ,)

account

c . get_account (sender) account_num = account . account_number sequence = account . sequence

fee

[Coin (amount = "0" , denom = "uband")] chain_id = c . get_chain_id ( )

txn

(Transaction ( ) . with_messages (request_msg) . with_sequence (sequence) . with_account_num (account_num) . with_chain_id (chain_id) . with_gas (2000000) . with_fee (fee) . with_memo (""))

sign_doc

txn . get_sign_doc (public_key) signature = private_key . sign (sign_doc . SerializeToString ( )) tx_raw_bytes = txn . get_tx_data (signature , public_key)

tx_sync

c . send_tx_sync_mode (tx_raw_bytes) print (MessageToJson (tx_sync)) Result

{ "txhash" :

"FEE8A58F7A68326A50B974C13721B55A6ABA6A1761A2D466A9940FF393F02C9E" , "rawLog" :

"[]" }

send_tx_async_mode(tx_bytes)

This function sends a transaction in async mode, that is, send and return immediately without waiting for the transaction process.

Parameter

- tx_bytes
- : Transaction raw bytes that is already signed.

Return

- abci_type.TxResponse

Example

import os

from pyband . client import Client from pyband . transaction import Transaction from pyband . wallet import PrivateKey

from pyband . proto . cosmos . base . v1beta1 . coin_pb2 import Coin from pyband . proto . oracle . v1 . tx_pb2 import MsgRequestData from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

c

Client (grpc_url)

MNEMONIC

os . getenv ("MNEMONIC") private_key = PrivateKey . from_mnemonic (MNEMONIC) public_key = private_key . to_public_key ( ) sender_addr = public_key . to_address ( ) sender = sender_addr . to_acc_bech32 ( )

request_msg

MsgRequestData (oracle_script_id = 37 , calldata = bytes . fromhex ("0000000200000003425443000000034554480000000000000064") , ask_count = 4 , min_count = 3 , client_id = "BandProtocol" , fee_limit = [Coin (amount = "100" , denom = "uband")] , prepare_gas = 50000 , execute_gas = 200000 , sender = sender ,) account = c . get_account (sender) account_num = account . account_number sequence = account . sequence

fee

[Coin (amount = "0" , denom = "uband")] chain_id = c . get_chain_id ( )

txn

(Transaction ( ) . with_messages (request_msg) . with_sequence (sequence) . with_account_num (account_num) . with_chain_id (chain_id) . with_gas (2000000) . with_fee (fee) . with_memo (""))

sign_doc

txn . get_sign_doc (public_key) signature = private_key . sign (sign_doc . SerializeToString ( )) tx_raw_bytes = txn . get_tx_data (signature , public_key)

tx_async

c . send_tx_async_mode (tx_raw_bytes) print (MessageToJson (tx_async)) Result

{

"txhash" :

"C685F799E4D870353364155602C14520416FC274293DFC9EFC3575357F9A8893"

}

send_tx_block_mode(tx_bytes)

This function sends a transaction in block mode, that is, send and wait until the transaction has been committed to a block.

Parameter

- tx_bytes
- : Transaction raw bytes that is already signed.

Return

- abci_type.TxResponse

Example

import os

from pyband . client import Client from pyband . transaction import Transaction from pyband . wallet import PrivateKey

from pyband . proto . cosmos . base . v1beta1 . coin_pb2 import Coin from pyband . proto . oracle . v1 . tx_pb2 import MsgRequestData from google . protobuf . json_format import MessageToJson

grpc_url

""

without https://

c

Client (grpc_url)

MNEMONIC

os . getenv ("MNEMONIC") private_key = PrivateKey . from_mnemonic (MNEMONIC) public_key = private_key . to_public_key ( ) sender_addr = public_key . to_address ( ) sender = sender_addr .

to_acc_bech32 ( )

request_msg

MsgRequestData (oracle_script_id = 37 , calldata = bytes . fromhex ("0000000200000003425443000000034554480000000000000064") , ask_count = 4 , min_count = 3 , client_id = "BandProtocol" , fee_limit = [Coin (amount = "100" , denom = "uband")] , prepare_gas = 50000 , execute_gas = 200000 , sender = sender ,)

account

c . get_account (sender) account_num = account . account_number sequence = account . sequence

fee

[Coin (amount = "0" , denom = "uband")] chain_id = c . get_chain_id ( )

txn

(Transaction ( ) . with_messages (request_msg) . with_sequence (sequence) . with_account_num (account_num) . with_chain_id (chain_id) . with_gas (2000000) . with_fee (fee) . with_memo (""))

sign_doc

txn . get_sign_doc (public_key) signature = private_key . sign (sign_doc . SerializeToString ( )) tx_raw_bytes = txn . get_tx_data (signature , public_key)

tx_block

c . send_tx_block_mode (tx_raw_bytes) print (MessageToJson (tx_block)) Result

{ "height" :

"603561" , "txhash" :

"A50970334A74461CF045D962EEA1230B18AAAC2CEE2E96C3C348672100D46A93" , "data" :

"0A090A0772657175657374" , "rawLog" :

"[{\"events\":[{\"type\":\"message\",\"attributes\":[{\"key\":\"action\",\"value\":\"request\"}]},{\"type\":\"raw_request\",\"attributes\":[{\"key\":\"data_source_id\",\"value\":\"61\"},
{\"key\":\"data_source_hash\",\"value\":\"07be7bd61667327aae10b7a13a542c7dfba31b8f4c52b0b60bf9c7b11b1a72ef\"},{\"key\":\"external_id\",\"value\":\"6\"},{\"key\":\"calldata\",\"value\":\"BTC ETH\"},
{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"57\"},{\"key\":\"data_source_hash\",\"value\":\"61b369daa5c0918020a52165f6c7662d5b9c1eee915025cb3d2b9947a26e48c7\"},
{\"key\":\"external_id\",\"value\":\"0\"},{\"key\":\"calldata\",\"value\":\"BTC ETH\"},{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"62\"},
{\"key\":\"data_source_hash\",\"value\":\"107048da9dbf7960c79fb20e0585e080bb9be07d42a1ce09c5479bbada8d0289\"},{\"key\":\"external_id\",\"value\":\"3\"},{\"key\":\"calldata\",\"value\":\"BTC ETH\"},
{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"60\"},{\"key\":\"data_source_hash\",\"value\":\"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac\"},
{\"key\":\"external_id\",\"value\":\"5\"},{\"key\":\"calldata\",\"value\":\"huobipro BTC ETH\"},{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"59\"},
{\"key\":\"data_source_hash\",\"value\":\"5c011454981c473af3bf6ef93c76b36bfb6cc0ce5310a70a1ba569de3fc0c15d\"},{\"key\":\"external_id\",\"value\":\"2\"},{\"key\":\"calldata\",\"value\":\"BTC ETH\"},
{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"60\"},{\"key\":\"data_source_hash\",\"value\":\"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac\"},
{\"key\":\"external_id\",\"value\":\"4\"},{\"key\":\"calldata\",\"value\":\"binance BTC ETH\"},{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"60\"},
{\"key\":\"data_source_hash\",\"value\":\"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac\"},{\"key\":\"external_id\",\"value\":\"9\"},{\"key\":\"calldata\",\"value\":\"bittrex BTC
ETH\"},{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"60\"},{\"key\":\"data_source_hash\",\"value\":\"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac\"},
{\"key\":\"external_id\",\"value\":\"7\"},{\"key\":\"calldata\",\"value\":\"kraken BTC ETH\"},{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"60\"},
{\"key\":\"data_source_hash\",\"value\":\"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac\"},{\"key\":\"external_id\",\"value\":\"8\"},{\"key\":\"calldata\",\"value\":\"bitfinex BTC
ETH\"},{\"key\":\"fee\"},{\"key\":\"data_source_id\",\"value\":\"58\"},{\"key\":\"data_source_hash\",\"value\":\"7e6759fade717a06fb643392bfde837bfc3437da2ded54feed706e6cd35de461\"},
{\"key\":\"external_id\",\"value\":\"1\"},{\"key\":\"calldata\",\"value\":\"BTC ETH\"},{\"key\":\"fee\"}]},{\"type\":\"request\",\"attributes\":[{\"key\":\"id\",\"value\":\"307081\"},
{\"key\":\"client_id\",\"value\":\"BandProtocol\"},{\"key\":\"oracle_script_id\",\"value\":\"37\"},{\"key\":\"calldata\",\"value\":\"0000000200000003425443000000034554480000000000000064\"},
{\"key\":\"ask_count\",\"value\":\"4\"},{\"key\":\"min_count\",\"value\":\"3\"},{\"key\":\"gas_used\",\"value\":\"111048\"},{\"key\":\"total_fees\"},
{\"key\":\"validator\",\"value\":\"bandvaloper1l2hchtyawk9tk43zzjrzr2lcd0zyxngcjdsshe\"},{\"key\":\"validator\",\"value\":\"bandvaloper17n5rmujk78nkgss7tjecg4nfzn6geg4cqtyg3u\"},
{\"key\":\"validator\",\"value\":\"bandvaloper1e9sa38742tzhmandc4gkqve9zy8zc0yremaa3j\"},{\"key\":\"validator\",\"value\":\"bandvaloper1lm2puy995yt8dh53cnazk3ge3m27t7cay4ndaq\"}]}]}]" , "logs" :

[{ "events" :

[{ "type" :

"message" , "attributes" :

[{

"key" :

"action" ,

"value" :

"request"

}] } , { "type" :

"raw_request" , "attributes" :

[{

"key" :

"data_source_id" ,

"value" :

"61"

} , { "key" :

"data_source_hash" , "value" :

"07be7bd61667327aae10b7a13a542c7dfba31b8f4c52b0b60bf9c7b11b1a72ef" } , {

"key" :

"external_id" ,

"value" :

"6"

} , {

"key" :

"calldata" ,

"value" :

"BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"57"

} , { "key" :

"data_source_hash" , "value" :

"61b369daa5c0918020a52165f6c7662d5b9c1eee915025cb3d2b9947a26e48c7" } , {

"key" :

"external_id" ,

"value" :

"0"

} , {

"key" :

"calldata" ,

"value" :

"BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"62"

} , { "key" :

"data_source_hash" , "value" :

"107048da9dbf7960c79fb20e0585e080bb9be07d42a1ce09c5479bbada8d0289" } , {

"key" :

"external_id" ,

"value" :

"3"

} , {

"key" :

"calldata" ,

"value" :

"BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"60"

} , { "key" :

"data_source_hash" , "value" :

"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac" } , {

"key" :

"external_id" ,

"value" :

"5"

} , {

"key" :

"calldata" ,

"value" :

"huobipro BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"59"

} , { "key" :

"data_source_hash" , "value" :

"5c011454981c473af3bf6ef93c76b36bfb6cc0ce5310a70a1ba569de3fc0c15d" } , {

"key" :

"external_id" ,

"value" :

"2"

} , {

"key" :

"calldata" ,

"value" :

"BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"60"

} , { "key" :

"data_source_hash" , "value" :

"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac" } , {

"key" :

"external_id" ,

"value" :

"4"

} , {

"key" :

"calldata" ,

"value" :

"binance BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"60"

} , { "key" :

"data_source_hash" , "value" :

"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac" } , {

"key" :

"external_id" ,

"value" :

"9"

} , {

"key" :

"calldata" ,

"value" :

"bittrex BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"60"

} , { "key" :

"data_source_hash" , "value" :

"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac" } , {

"key" :

"external_id" ,

"value" :

"7"

} , {

"key" :

"calldata" ,

"value" :

"kraken BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"60"

} , { "key" :

"data_source_hash" , "value" :

"2e588de76a58338125022bc42b460072300aebbcc4acaf55f91755c1c1799bac" } , {

"key" :

"external_id" ,

"value" :

"8"

} , {

"key" :

"calldata" ,

"value" :

"bitfinex BTC ETH"

} , {

"key" :

"fee"

} , {

"key" :

"data_source_id" ,

"value" :

"58"

} , { "key" :

"data_source_hash" , "value" :

"7e6759fade717a06fb643392bfde837bfc3437da2ded54feed706e6cd35de461" } , {

"key" :

"external_id" ,

"value" :

"1"

} , {

"key" :

"calldata" ,

"value" :

"BTC ETH"

} , {

"key" :

"fee"

}] } , { "type" :

"request" , "attributes" :

[{

"key" :

"id" ,

"value" :

"307081"

} , {

"key" :

"client_id" ,

"value" :

"BandProtocol"

} , {

"key" :

"oracle_script_id" ,

"value" :

"37"

} , { "key" :

"calldata" , "value" :

"0000000200000003425443000000034555448000000000000064" } , {

"key" :

"ask_count" ,

"value" :

"4"

} , {

"key" :

"min_count" ,

"value" :

"3"

} , {

"key" :

"gas_used" ,

"value" :

"111048"

} , {

"key" :

"total_fees"

} , { "key" :

"validator" , "value" :

"bandvaloper1l2hchtyawk9tk43zzjrzr2lcd0zyxngcjdsshe" } , { "key" :

"validator" , "value" :

"bandvaloper17n5rmujk78nkgss7tjecg4nfzn6geg4cqtyg3u" } , { "key" :

"validator" , "value" :

"bandvaloper1e9sa38742tzhmandc4gkqve9zy8zc0yremaa3j" } , { "key" :

"validator" , "value" :

"bandvaloper1lm2puy995yt8dh53cnazk3ge3m27t7cay4ndaq" }] }] }] , "gasWanted" :

"2000000" , "gasUsed" :

"566496" } Previous Getting Started Next Data Module