With CosmWasm

CosmWasm is an advanced smart contracting platform built for the Cosmos ecosystem.Cosm refers to “Cosmos” whileWasm refers to WebAssembly. CosmWasm uniquely enables developers to build multi-chain smart contracts, making use of the InterBlockchain Communication (IBC) Protocol.

CosmWasm contract can update and fetch the prices from Band Standard Dataset using Band's CosmWasm contract, deployed on their network.

The deployed contract addresses can be found inSupported Blockchains Section.

Build

Contract

To compile all contracts, run the following script in the repo root:/scripts/build_artifacts.sh or the command below: The optimized wasm code and its checksums can be found in the/artifacts directory

docker run --rm -v "(pwd)":/code \ --mount type=volume,source="(basename "(pwd)")_cache",target=/code/target \ --mount type=volume,source=registry_cache,target=/usr/local/cargo/registry \ cosmwasm/workspace-optimizer:0.12.7

Schema

To generate the JSON schema files for the contract call, queries and query responses, run the following script in the repo root:/scripts/build_schemas.sh or runcargo schema in the smart contract directory.

Usage

To query the prices from Band Protocol's StdReference contracts, the contract looking to use the price values should query Band Protocol'sstd_reference contract.

QueryMsg

The query messages used to retrieve price data for price data are as follows:

pub

enum

QueryMsg

{ GetReferenceData

{ // Symbol pair to query where: // symbol_pair := (base_symbol, quote_symbol) // e.g. BTC/USD ≡ ("BTC", "USD") symbol_pair :

(String ,

String) , } , GetReferenceDataBulk

{ // Vector of Symbol pair to query // e.g.≡ <("BTC", "USD"), ("ETH", "USD"), ("BAND", "BTC")> symbol_pairs :

Vec < (String ,

String)

, } , }

ReferenceData

ReferenceData is the struct that is returned when querying withGetReferenceData orGetReferenceDataBulk where the bulk variant returnsVec

ReferenceData is defined as:

pub

struct

ReferenceData

{ // Pair rate e.g. rate of BTC/USD pub rate :

Uint256 , // Unix time of when the base asset was last updated. e.g. Last update time of BTC in Unix time pub last_updated_base :

Uint64 , // Unix time of when the quote asset was last updated. e.g. Last update time of USD in Unix time pub last_updated_quote :

Uint64 , }

Examples

Single Query

For example, if we wanted to query the price ofBTC/USD , the demo function below shows how this can be done.

fn

demo (std_ref_addr :

Addr , symbol_pair :

(String ,

String) ,)

->

StdResult < ReferenceData

{ deps . querier . query_wasm_smart (& std_ref_addr , & QueryMsg :: GetReferenceData

{ symbol_pair , } ,) } Where the result fromdemo(std_ref_addr, ("BTC", "USD")) would yield:

ReferenceData(23131270000000000000000, 1659588229, 1659589497) and the results can be interpreted as:

- BTC/USD* rate = 23131.27 BTC/USD
-
    - lastUpdatedBase = 1659588229
-
    - lastUpdatedQuote = 1659589497

Bulk Query

fn

demo (std_ref_addr :

Addr , symbol_pairs :

Vec < String

,)

->

StdResult < ReferenceData

{ deps . querier . query_wasm_smart (& std_ref_addr , & QueryMsg :: GetReferenceDataBulk

{ symbol_pairs , } ,) } Where the result fromdemo(std_ref_addr, [("BTC", "USD"), ("ETH", "BTC")]) would yield:

[ReferenceData(23131270000000000000000, 1659588229, 1659589497), ReferenceData(71601775432131482, 1659588229, 1659588229)] and the results can be interpreted as:

- BTC/USD* rate = 23131.27 BTC/USD
-
    - lastUpdatedBase = 1659588229
-
    - lastUpdatedQuote = 1659589497

- ETH/BTC* rate = 0.07160177543213148 ETH/BTC
-
    - lastUpdatedBase = 1659588229
-
    - lastUpdatedQuote = 1659588229 Previous With Solidity Next On Client Libraries