How to Read Data

Lens contract is all you need. Instead of collecting and calculating addresses here and there to get the data you need, you can look up everything in one place : Lens.

zksyncEra Lens : 0xf55150000aac457eCC88b34dA9291e3F6E7DB165

Linea Lens :

0xaA18cDb16a4DD88a59f4c2f45b5c91d009549e06

ABI file :[Repo here]

There are 3 return struct types : Bribe / Gauge / Pool

These are all you need to get user balances, calculate Apr, pool info, etc. We also use this Lens contract for our frontend.

You could see the original source code at therepo .

As you can see, except for 2 functions marked as 'view', all of them are not view functions.

So you should call them as staticCall to get the result returned.

```
Copy //functions you could use

//type 'Token' here is byte32. look 'How to interact with Velocore' for the details
functionspotPrice(ISwapswap,Tokenbase,Tokenquote,uint256baseAmount)publicreturns(uint256)
functionspotPrice(Tokenbase,Tokenquote,uint256amount)publicreturns(uint256)
functionspotPrice(Tokenquote,Token[]memorytok,uint256[]memoryamount)publicreturns(uint256)
functionuserBalances(addressuser,Token[]calldatats)publicviewreturns(uint256[]memorybalances)
functionwombatGauges(addressuser)externalreturns(GaugeData[]memorygaugeDataArray)
functioncanonicalPools(addressuser,uint256begin,uint256maxLength) externalreturns(GaugeData[]memorygaugeDataArray)
functioncanonicalPoolLength()externalreturns(uint256)
functionqueryGauge(addressgauge,addressuser)externalreturns(GaugeDatamemorypoolData)
functiongetPoolBalance(addresspool,Tokent)externalviewreturns(uint256)
functionqueryPool(addresspool)externalreturns(PoolDatamemoryret)
functionemissionRate(IGaugegauge)externalreturns(uint256)

//additionalpricequeryfunctionaddedonLinea
functionwombatPrice(addresswombatPool,Tokenasset,Tokenunit,uint256amount)publicviewreturns(uint256)
```

For example you could query,

1. Total votes, APR(Average interest Rate per second), staked TVL, Emission rate from 'GaugeData'.
2. Use various functions above to get gauge data.
3. *There are too many variable pools and can cause errors depending on the RPC limit when querying at once, so we have prepared a canonicalPools function with pagenation. Wombat pools are usually few, so querying with wombatGauges() shouldn't cause errors.
4. Pool's TVL, coverage ratio of wombat pools (difference between minted LPTokens and reserves), TokenA/B in specific pool from 'PoolData'
5. Stake position of specific user (using functions like queryGauge(), userBalances())
6. Get relevantToken struct in bytes32
7. to build parameters for other functions.
8.

Last updated8 days ago On this page