Repo Data Model

Contents

[Data Model 1](#_Toc63862408)

[Repo Master 1](#_Toc63862409)

[Repo Transaction 1](#_Toc63862410)

[Repo Transaction History 2](#_Toc63862411)

[Events 3](#_Toc63862412)

[New Repo Trade 3](#_Toc63862413)

[Add Repo Master 3](#_Toc63862414)

[Add Repo Transaction 4](#_Toc63862415)

[Add Repo Transaction History 5](#_Toc63862416)

[Close Repo Trade 5](#_Toc63862417)

[Update Repo Transactions 5](#_Toc63862418)

[Add Repo Transaction History 6](#_Toc63862419)

[Cancel Repo Trade 6](#_Toc63862420)

[Update Repo Transactions 6](#_Toc63862421)

[Add Repo Transaction History 7](#_Toc63862422)

[Rerate Repo Trade 7](#_Toc63862423)

[Update Repo Transactions 7](#_Toc63862424)

[Add Repo Transaction History 7](#_Toc63862425)

[API 8](#_Toc63862426)

[getRepo 8](#_Toc63862427)

[Return Value 8](#_Toc63862428)

[Parameter Logic 9](#_Toc63862429)

[addRepoMaster 10](#_Toc63862430)

[Parameters 10](#_Toc63862431)

[Program Logic 11](#_Toc63862432)

[addRepoTransaction 11](#_Toc63862433)

[Parameters 11](#_Toc63862434)

[Program Logic 11](#_Toc63862435)

[closeRepoTransaction 11](#_Toc63862436)

[Parameters 11](#_Toc63862437)

[Program Logic 11](#_Toc63862438)

[cancelRepoTransaction 11](#_Toc63862439)

[Parameters 11](#_Toc63862440)

[Program Logic 11](#_Toc63862441)

[initializeDatastore 11](#_Toc63862442)

[Parameters 11](#_Toc63862443)

[Program Logic 11](#_Toc63862444)

[clearRepoData 11](#_Toc63862445)

[Program Logic 11](#_Toc63862446)

Here is the data model to for repo transactions.

# Data Model

The data model consists of two parts:

1. Repo master;

2. Repo transactions. Each repo transaction refers to one repo master, multiple repo transactions may refer to the same or different repo master.

## Repo Master

A repo master entry should contain the following fields:

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Code | String, unique id of the repo master | MMRPE420BS |
| Currency | String, loan currency | USD |
| Day Count | String, describing the how to calculate interest | ACT/360 |

## Repo Transaction

A repo transaction entry should contain the following fields:

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | String, unique id of the transaction. | “300766” |
| Type | String, repo transaction type. “RP” for repo (collateral in, money out) and “RR” for reverse repo (collateral out, money in). | RP, RR |
| Portfolio | AIM portfolio that carries out the transaction. | 60001, 40017-B |
| Custodian | custodian account of the collateral. | BOCHK, BOCHK-LIQ |
| Collateral ID Type | Collateral security id type | ISIN |
| Collateral ID | Collateral security id | XS1234567890 |
| Collateral Global ID | Collateral global security id | B000001 |
| Trade Date | String (yyyy-mm-dd), trade date of the repo transaction | 2021-05-23 |
| Settle Date | String (yyyy-mm-dd), settle date of the repo transaction | 2021-05-27 |
| Is Open Repo | Boolean, True if it is open repo, False otherwise | True, False |
| Maturity Date | Empty String if “Is Open Repo” is True, else String (yyyy-mm-dd), maturity date of the repo | 2021-06-30 |
| Quantity | Float number, collateral quantity | 300000 |
| Currency | String, local currency of the collateral. | USD |
| Price | Float number, price of the collateral when the repo transaction is made | USD |
| Collateral Value | Float number, market value of the collateral when the repo transaction is made | 1818000 |
| Repo Name | String, code of the repo master that the repo trade refers to | MMRPE420BS |
| Interest Rate | Float number, repo interest rate | 0.95 |
| Loan Amount | Float number, repo loan amount | 256900.30 |
| Broker | String, repo trade broker | BNP-REPO |
| Haircut | Float number, initialized to zero. It’s not available from Bloomberg repo transaction XML, but from third party repo reports. It need to be populated through another batch process. | 15 |
| Status | the status of the repo transaction. Only 3 status are allowed: “open”, “closed”, and “canceled”.  When a new repo comes in, status = “open”; when it is closed explicitly via a close transaction, status = “closed”; when it is canceled via a cancel transaction, status = “canceled”.  Note that when a repo matures, the status will remain “open” because there will be no external event to update it. | open, closed, canceled |

## Repo Transaction History

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | String, unique id of the transaction. | “300766” |
| Action | String, action performed on the repo. Only 4 actions allowed: “open”, “close”, “cancel”, “rerate” | open, close, cancel, rerate |
| Date | String (yyyy-mm-dd), date of the action | 2021-01-28 |
| Interest Rate | Float number. If the action is “open” or “rerate”, the value is the interest rate on the action day; 0 otherwise. | 0, 0.85 |
| Time Stamp | String (yyyy-mm-dd HH:MM:SS), time when this transaction is entered into the database. | 2020-12-31 17:30:18 |

# Events

These are the events that update the data model:

1. New repo trade;

2. Close repo trade;

3. Cancel repo trade;

4. Rerate repo trade.

## New Repo Trade

When a new repo trade event happens, there will be two set of data as below. In this event, we add the repo master data, repo transaction data, as well as repo transaction history data to the data store.

1. repo master, for repo masters;

2. repo trade, for repo transactions.

### Add Repo Master

Here is the logic to add a repo master record from repo master XML data:

|  |  |  |
| --- | --- | --- |
| Field | Logic | Sample |
| Code | Tag “Code”, ERROR if there is already an entry with the same “Code” value. | MMRPE420BS |
| Currency | Tag “BifurcationCurrency” | USD |
| Day Count | Tag “AccrualDaysPerMonth”: if it is a number, keep it unchanged; if it is a string “Actual”, then map it to “ACT”; if it is any other value, ERROR.  Tag “AccrualDaysPerYear”: same logic as above.  Combine the mapped values of “AccrualDaysPerMonth” and “AccrualDaysPerYear”, with a “/” in between. | AccrualDaysPerMonth = Actual  AccrualDaysPerYear = 360  Result is:  ACT/360 |

### Add Repo Transaction

To a new repo transaction record from repo transaction XML data:

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String. ERROR if there is already an entry with the same “Transaction Id” value. | “300766” |
| Type | There are 2 possible values: “RP” or “RR”  “RR” If the XML element is “ReverseRepo\_InsertUpdate”, “RP” if the XML element is “Repo\_InsertUpdate”, ERROR otherwise. | RP, RR |
| Portfolio | Tag “Portfolio” | 60001, 40017-B |
| Custodian | Tag “LocationAccount” | BOCHK, BOCHK-LIQ |
| Collateral ID Type | Tag “Investment”, it looks like:  Isin=XS1234567890  The value on the left-hand side of “=” is the collateral id type, convert it to all upper case.  The value on the right-hand side of “=” is the collateral id | ISIN |
| Collateral ID | See above | XS1234567890 |
| Collateral Global ID | Initialized to empty String |  |
| Trade Date | Tag “EventDate”, ignore the “T00:00:00” part. For example, 2018-08-27T00:00:00 is converted to 2018-08-27 | 2018-08-27 |
| Settle Date | Tag “SettleDate”, logic similar to “Trade Date” | 2021-05-27 |
| Is Open Repo | True if tag “OpenEnded” is there and its value is “CALC”, False otherwise. | True, False |
| Maturity Date | Empty String if tag “ActualSettleDate” value is “CALC” and “Is Open Repo” is True; ERROR if tag “ActualSettleDate” value is “CALC” and “Is Open Repo” is False; Convert to String otherwise (same logic as “Trade Date”) | “”, “2021-06-30” |
| Quantity | Tag “Quantity” | 300000 |
| Currency | Tag “CounterInvestment” | USD |
| Price | Tag “Price” | 101 |
| Collateral Value | Tag “NetCounterAmount” | 1818000 |
| Repo Name | Tag “RepoName”  Make sure there is a security master record whose “Code” value is the same as the “Repo Name” value. | MMRPE420BS |
| Interest Rate | Tag “Coupon” | 0.95 |
| Loan Amount | Tag “LoanAmount” | 256900.30 |
| Broker | Tag “Broker” | BNP-REPO |
| Haircut | Initialized to zero. | 0 |
| Status | Initialized to “open” | open |

### Add Repo Transaction History

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String. | “300766” |
| Action | “open” | open |
| Date | Tag “SettleDate”, convert to yyyy-mm-dd String | 2021-01-28 |
| Interest Rate | Tag “Coupon” | 0.95 |
| Time Stamp | String (yyyy-mm-dd HH:MM:SS), time when this transaction is entered into the database. | 2020-12-31 17:30:18 |

## Close Repo Trade

When a close repo trade event happens, there will be one set of data:

1. repo transactions;

### Update Repo Transactions

In this event, no new repo transactions will be added, instead we update the existing repo transactions from the data store.

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String | “300766” |

1. Find the repo transaction record from the data store, using “Transaction Id” to match, ERROR if not found;

2. If found, change its “Status” field value to “closed”. Make sure the repo “Status” is “open” before the update, ERROR otherwise.

### Add Repo Transaction History

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String. | “300766” |
| Action | “close” | close |
| Date | Tag “ActualSettleDate”, convert to yyyy-mm-dd String | 2021-01-28 |
| Interest Rate | 0 | 0 |
| Time Stamp | String (yyyy-mm-dd HH:MM:SS), time when this transaction is entered into the database. | 2020-12-31 17:30:18 |

## Cancel Repo Trade

When a cancel repo trade event happens, there will be one set of data:

1. repo transactions.

### Update Repo Transactions

In this event, no new repo transactions will be added, instead we update the existing repo transactions from the data store.

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String | “300766” |

1. Find the repo transaction record from the data store, using “Transaction Id” to match, ERROR if not found;

2. If found, change its “Status” field value to “canceled”.

### Add Repo Transaction History

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String. | “300766” |
| Action | “cancel” | cancel |
| Date | String (yyyy-mm-dd), date when this transaction is entered into the database. | 2020-12-31 |
| Interest Rate | 0 | 0 |
| Time Stamp | String (yyyy-mm-dd HH:MM:SS), time when this transaction is entered into the database. | 2020-12-31 17:30:18 |

## Rerate Repo Trade

When a rerate repo trade event happens, there will be one set of data:

1. repo transactions.

### Update Repo Transactions

In this event, no new repo transactions will be added, instead we update the existing repo transactions from the data store.

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “Loan”, it looks like:  UserTranId1=Ref000001  If on the left-hand side of “=”, the string is “UserTranId1”, then take the value on the right-hand side of “=”; ERROR otherwise. | “300766” |
| Interest Rate | Tag “RateTable”, sub tag “Rate”, convert it to Float | 1.5 |

1. Find the repo transaction record from the data store, using “Transaction Id” to match, ERROR if not found;

2. If found, change its “Interest Rate” field value to the value of “Interest Rate” from the trade data.

### Add Repo Transaction History

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “Loan”, follow the same logic as “update repo transactions” above. | “300766” |
| Action | “rerate” | rerate |
| Date | Tag “RateDate”, convert to “yyyy-mm-dd” String | 2020-12-31 |
| Interest Rate | Tag “Rate”, follow the same logic as “update repo transactions” above. | 1.5 |
| Time Stamp | String (yyyy-mm-dd HH:MM:SS), time when this transaction is entered into the database. | 2020-12-31 17:30:18 |

# API

Here are the functions we want to implement.

## getRepo

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Type | Is Mandatory | Default Value |
| date | String (yyyy-mm-dd) | Yes | No default |
| status | String | No | “open” |
| portfolio | String | No | “all” |
| custodian | String | No | “all” |
| repoName | String | No | “all” |
| broker | String | No | “all” |
| hasHairCut | String | No | “all” |

Sample:

getRepo(

date

, status=”open”, portfolio=”all”, custodian=”all”, repoName=”all”, broker=”all”, hasHairCut=”all”

)

### Return Value

A list of repo objects, where each object is represented by a Dictionary object that contains the following fields:

|  |  |
| --- | --- |
| Field | Value |
| Transaction Id | From repo transaction |
| Type | From repo transaction |
| Portfolio | From repo transaction |
| Custodian | From repo transaction |
| Collateral ID Type | From repo transaction |
| Collateral ID | From repo transaction |
| Collateral Global ID | From repo transaction |
| Trade Date | From repo transaction |
| Settle Date | From repo transaction |
| Is Open Repo | From repo transaction |
| Maturity Date | From repo transaction |
| Quantity | From repo transaction |
| Currency | From repo transaction |
| Price | From repo transaction |
| Collateral Value | From repo transaction |
| Repo Name | From repo transaction |
| Interest Rate | From repo transaction |
| Loan Amount | From repo transaction |
| Broker | From repo transaction |
| Haircut | From repo transaction |
| Status | From repo transaction |
| Day Count | From repo master |

### Parameter Logic

Parameter values are useful to filter repo transactions we need from all the repo transactions in the datastore. Here is the logic.

If parameter’s value is “all”, that means it will not be used in the filtering process.

date: String (yyyy-mm-dd), a mandatory parameter. This parameter is useful together with parameter “status”.

status: String. It can only take 4 values: “all”, “open”, “closed”, and “canceled”

|  |  |
| --- | --- |
| default | open |
| open | Filter repo transactions that,  1. status is “open”, and  2. “Is Open Repo” is True or “Maturity Date” > date. |
| closed | Filter repo transactions that,  1. status is “closed”, or  2. “Is Open Repo” is False and “Maturity Date” <= date and status is not “canceled”. |
| canceled | Filter repo transactions whose status is “canceled” |
| all | Get all repo positions no matter what the status is |

portfolio: String

|  |  |
| --- | --- |
| default | all |
| Other value | Filter repo transactions whose “Portfolio” == portfolio |

custodian: String.

|  |  |
| --- | --- |
| default | all |
| Other value | Filter repo transactions whose “Custodian Account” == custodian |

repoName: String.

|  |  |
| --- | --- |
| default | all |
| Other value | Filter repo transactions whose “Repo Name” == repoName |

broker: String.

|  |  |
| --- | --- |
| default | all |
| Other value | Filter repo transactions whose “Broker” == broker |

hasHairCut: Bool

|  |  |
| --- | --- |
| default | all |
| True | Filter repo transactions whose “Haircut” != 0 |
| False | Filter repo transactions whose “Haircut” == 0 |

## addRepoMaster

Add a repo master record to the datastore.

### Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Value |
| master | Dictionary | A dictionary object containing repo master information from the XML |

Data fields of parameter “master”

|  |  |  |
| --- | --- | --- |
| Field | Type | Sample |
| Code | String | MMRPE420BS |
| BifurcationCurrency | String | USD |
| AccrualDaysPerMonth | String | Actual, “30” |
| AccrualDaysPerYear | String | Actual, “360” |

### Program Logic

|  |  |
| --- | --- |
| Condition | Action |
| Master Code already exist in data store | ERROR: RepoMasterAlreadyExist  Error log: repo master code |
| Else | Add repo master to data store |

## getRepoTransactionHistory

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Type | Is Mandatory | Default Value |
| userTranId | String | Yes | N/A |

### Return Value

A list of Dictionary objects that contains the following fields:

|  |  |  |
| --- | --- | --- |
| Field | Value | Sample |
| Transaction Id | Tag “UserTranId1”, convert it to String. | “300766” |
| Action | “cancel” | cancel |
| Date | String (yyyy-mm-dd), date when this transaction is entered into the database. | 2020-12-31 |
| Interest Rate | A float number | 0.75 |
| Time Stamp | String (yyyy-mm-dd HH:MM:SS), time when this transaction is entered into the database. | 2020-12-31 17:30:18 |

### Program Logic

Return a list of transaction histories whose “UserTrandId1” field is equal to parameter userTranId. If no records are found, return an empty list. Make sure the returned list is sorted by the time stamp, earlier records come first.

## getUserTranIdsFromRepoName

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Type | Is Mandatory | Default Value |
| repoName | String | Yes | N/A |

### Return Value

A list of string that represents the “UserTranId1” field of all repo transactions associated with the repo name.

### Program Logic

Each time when we add a repo transaction, there is a “UserTranId1” field and a “RepoName” field. So find all repo transactions whose “RepoName” is equal to parameter repoName. Then return their “UserTranId1” field as a list.

If nothing is found, return an empty list.

## addRepoTransaction

Add a repo transaction to the datastore

### Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Value |
| transaction | Dictionary | A dictionary object containing repo transaction information from the XML |

Data fields of parameter “transaction”

|  |  |  |
| --- | --- | --- |
| Field | Type | Sample |
| TransactionType | String | “Repo\_InsertUpdate” or “ReverseRepo\_InsertUpdate” |
| UserTranId1 | String | “300734” |
| Portfolio | String | “12734” |
| LocationAccount | String | BOCHK |
| Investment | String | Isin=XS1234567890 |
| EventDate | String | 2018-08-27T00:00:00 |
| SettleDate | String | 2018-08-27T00:00:00 |
| ActualSettleDate | String | 2020-03-10T00:00:00, CALC |
| OpenEnded | String | This field is optional, if it exists, its value is String “CALC” |
| Quantity | Float | 300000 |
| CounterInvestment | String | USD |
| Price | Float | 95.23 |
| NetCounterAmount | Float | 1818234 |
| RepoName | String | MMRPE420BS |
| Coupon | Float | 0.95 |
| LoanAmount | Float | 1818234 |
| Broker | String | BNP-REPO |

### Program Logic

|  |  |
| --- | --- |
| Condition | Action |
| TransactionType is not of the proper values | ERROR: InvalidRepoTransactionType  Error log: Repo transaction type |
| Repo transaction id already exists in datastore | ERROR: RepoTransactionAlreadyExist  Error log: Repo transaction id |
| Repo master code not exists in datastore | ERROR: RepoMasterNotExist  Error log: Repo transaction id, repo name |
| Else | Add repo transaction, update repo transaction history |

## closeRepoTransaction

Close a repo transaction in the datastore.

### Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Value |
| transaction | Dictionary | A dictionary object containing repo transaction information from the XML |

Data fields of parameter “transaction”

|  |  |  |
| --- | --- | --- |
| Field | Type | Sample |
| UserTranId1 | String | “734567” |
| ActualSettleDate | String | 2018-08-31T00:00:00 |

### Program Logic

|  |  |
| --- | --- |
| Condition | Action |
| Repo transaction id not exists in datastore | ERROR: RepoTransactionNotExist  Error log: Repo transaction id |
| Repo transaction id exists, but status is “canceled” | ERROR: CloseCanceledRepoTransaction  Error log: Repo transaction id |
| Else | Update repo transaction, update repo transaction history |

## cancelRepoTransaction

Cancel a repo transaction in the datastore.

### Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Value |
| transaction | Dictionary | A dictionary object containing repo transaction information from the XML |

Data fields of parameter “transaction”

|  |  |  |
| --- | --- | --- |
| Field | Type | Sample |
| UserTranId1 | String | “734567” |

### Program Logic

|  |  |
| --- | --- |
| Condition | Action |
| Repo transaction id not exists in datastore | ERROR: RepoTransactionNotExist  Error log: Repo transaction id |
| Else | Update repo transaction, update repo transaction history |

## initializeDatastore

Initialize the datastore in two mode: test or production. Only after calling this function, all the above repo transaction functions can be called.

### Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Type | Value |
| mode | String | Indicate the datastore mode |

### Program Logic

|  |  |
| --- | --- |
| Condition | Action |
| mode = “production” | Connect to production database |
| Else | Connect to test database |

## clearRepoData

This function is meant to help testing only. It clears all the data in the repo master, repo transactions and repo transaction history. This function works ONLY when connecting to the test database.

### Program Logic

|  |  |
| --- | --- |
| Condition | Action |
| Database is in production mode | ERROR: NoDataClearingInProuctionMode |
| Else | Clears all data in the repo master, repo transactions and repo transaction history |