Yugandhar Open Master Data Management (MDM) Hub

API and Transaction Reference Guide

Yugandhar Open MDM Hub Release – V2.0.0
Date – 11/06/2018

Contents

About Yugandhar Open MDM Hub Project

Master Data Management came a long way in last decade or so. There are currently more than 20 MDM solutions catering to various specializations of MDM like Customer Data Integration (CDI), Product Information Management (PIM), vendor and supplier management etc. However most of these solutions come with licensing costs amounting to thousands of dollar. To offer a completely free solution which would be made available through Apache 2.0 license, A Project is started in 2017 under the name 'Yugandhar Open MDM Project' to build Open Source MDM solutions catering to CDI, PIM and Data Governance Capabilities. Yugandhar in Sanskrit means Ever Lasting and the strongest of its time. Our vision is to build the strongest, Open Source, Multi Domain, Cross Industry and completely free MDM Solution.

We are happy to announce that the first release of the Yugandhar MDM Hub catering to CDI solution is built with Open source technologies like Spring and Hibernate etc, inbuilt data Model, 400+ ready to use services and having incredible Out of the Box capabilities is currently being distributed. We aim to make the current CDI offering the strongest and Planning to bring Data Stewardship and PIM solutions in upcoming years.

About this document

This document covers the Out of the box (OOTB) API and services provided by MDM Hub which can be invoked from out side the MDM Hub or can be utilized to build a new composite service having custom logic.

Invoking Transactions on Yugandhar MDM Hub

- ### REST Webservice

Testing the Yugandhar Open MDM Hub REST webservices is covered in Development Environment Setup guide section 'TEST With SOAPUI'.

- ### MQ interface

MQ messages can be invoked using MQ client. You may build your own java class to send message to request queue by taking reference from YugJMSMessageSender class or use The response will be sent to default response queue. Also take help from activemq site http://activemq.apache.org/how-to-unit-test-jms-code.html.

We will be using SOAPUI based testing approach in this document.

The JSON Message layout

Common structure for invoking all transactions

All the transactions must follow a predefined format to invoke transaction on MDM hub. The request and response have Tranaction Transfer Object (TxnTransferObj). The transfer object encapsulates Header and Payload objects. The response may additionally have error response object inside transfer object.

Request Message Layout

- **Header Object**

The header object has below attributes

- requesterLanguage: The language of the request. It is referred from CONFIG_LANGUAGE_CODE table.
- requesterLocale: The locale of the requester
- userName: The username of the requester. This attribute is used for authorization
- userRole: The role of the requester. This attribute is used for authorization.
- txnCorrelationId: The correlation id of the message. This is sent back in the response as it is.
- txnMessageId: The message id of the transaction. The value of this attribute is used to update the UPDATED_TXN_REF_ID during persistent transaction. i.e. for create / update transaction all the tables getting updated will have UPDATED_TXN_REF_ID updated with this this id which can be used to trace the objects changed in one transaction.
- requestOriginSource: This attribute is used to define the request origin of the message. This attribute can be used to perform any additional logic while executing transaction.
- requesterSystem: This attribute is used to define the requesting system of the message. This attribute can be used to perform any additional logic while executing transaction.
- transactionResponseCode: This is the response code of the transaction which can either be SUCCESS or FAIL.
- requestTimeStamp: The timestamp when the request is received.
- responseTimeStamp: The timestamp when the response is sent back.
- TotalExecutionTimeMillies: The total execution time of the transaction in milliseconds.
- transactionServiceName: The name of the transaction being invoked on MDM hub. E.g. createLegalEntity.

Sample header Object -

```
    "txnHeader": {
        "requesterLanguage": "1",
        "requesterLocale": null,
        "userName": "yugandhar",
        "userRole": "admin",
        "accessToken": null,
        "txnCorrelationId": null,
        "txnMessageId": "",
        "requestOriginSource": null,
        "requesterSystem": null,
         "transactionServiceName": "createLegalEntity"
       }
```

- **TxnPayload Object**

The txnPayload is the default object encapsulating all the data objects. Also this object has the pagination related properties as well.

Refer Javadoc for the list of DOs and attributes of this object.

- **Primary Key Auto generation Vs providing in the request**

Snippet: Create legal entity transaction with user provided idpk

```
          "transactionServiceName": "createLegalEntity"
       },
       "txnPayload": {
                       "legalentityDO":      {
                      "primaryKeyDO": {
                      "idPk": "77776663666AAAA211"
                      },
                      "displayName": "JOHN MCLEAN",
```

Snippet: Create legal entity transaction with auto generated idpk

```
          "transactionServiceName": "createLegalEntity"
       },
       "txnPayload": {
                       "legalentityDO":      {
                      "idPk": null
                      "displayName": "JOHN MCLEAN",
```

Refer section ‘Pluggable primary keys’ section of ‘Development and Customization guide’ for more details.

Response Message Layout

- **Header Object**

Same as request object. Additionally ‘totalExecutionTimeMillies’ is sent in the response which denotes the time in milliseconds consumed for processing the request.

- **TxnPayload Object**

Same as request object. Additionally the response would have responseCode as ‘SUCCESS’ or ‘FAIL’ which denotes if the transaction is successful or failed in MDM Hub.

- **Error Response**

Sample error response

```
{
  "responseCode": "FAIL",
  "txnHeader":   {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "0909cd3c-b4c1-495e-8ce1-6d8f89a7c9ec",
    "transactionServiceName": "createLegalEntity"
  },
  "txnPayload": {"errorResponseObj":   {
    "validationResult": false,
    "errorCode": "10085",
    "errorMessage": "lePersonDO.genderRefkey  is needed in the request",
    "additionalErrorMessage": "lePersonDO.genderRefkey is needed in the request"
  }}
}
```

Sample Request and Response Message

Sample Request

```json
{
"txnHeader": {
        "requesterLanguage": "1",
        "requesterLocale": null,
        "userName": "yugandhar",
        "userRole": "admin",
        "accessToken": null,
        "txnCorrelationId": null,
        "txnMessageId": "",
        "requestOriginSource": null,
        "requesterSystem": null,
        "paginationStartIndex": null,
        "paginationEndIndex": null,
        "paginationTotalResultCount": null,
        "transactionServiceName": "createLegalEntity"
        },
"txnPayload": {
        "legalentityDO": {
                "primaryKeyDO": {
                "idPk": "11116663666777799d1n"
                },
                "displayName": "SACHIN",
                "entityObjectTypeRefkey": "1",
                "classificationCodeRefkey": "1",
                "importanceTypeRefkey": "1",
                "leRatingRefkey": "1",
                "statusTypeRefkey": "1",
                "sourceSystemRefkey": "1",
                "onboardingDate": "2015-10-23",
                "offboardingDate": null,
                "kycVerificationFlag": "Y",
                "description": "NO Issues",
                "lePersonDO": {
                "legalentityIdpk": null,
                "version": null,
                "personTypeRefkey": null,
                "genderRefkey": "1",
                "dateOfBirth": "1989-10-23",
                "countryOfBirthRefkey": "356",
                "countryCitizenshipRefkey": "356",
```

Sample Response

```json
{
"responseCode": "SUCCESS",
"txnHeader": {
        "requesterLanguage": "1",
        "userName": "yugandhar",
        "userRole": "admin",
        "txnMessageId": "326dfd40-157b-4921-b896-55679a46cfa0",
        "transactionServiceName": "createLegalEntity",
        "totalExecutionTimeMillies": "1221"
        },
"txnPayload": {
        "legalentityDO": {
                "idPk": "11116663666777799d1n",
                "version": 0,
                "createdTs": "2018-01-08T09.39.24.041+0000",
                "updatedTs": "2018-01-08T09.39.24.041+0000",
                "updatedByUser": "rakesh",
                "updatedByTxnId": "326dfd40-157b-4921-b896-55679a46cfa0",
                "displayName": "SACHIN",
                "entityObjectTypeRefkey": "1",
                "entityObjectTypeRefValue": "LEPERSON",
                "classificationCodeRefkey": "1",
                "classificationCodeRefValue": "CUSTOMER",
                "importanceTypeRefkey": "1",
                "importanceTypeRefValue": "HIGH",
                "leRatingRefkey": "1",
                "leRatingRefValue": "SINGLE STAR",
                "statusTypeRefkey": "1",
                "statusTypeRefValue": "ACTIVE",
                "sourceSystemRefkey": "1",
                "sourceSystemRefValue": "LE SOURCE SYSTEM 1",
                "onboardingDate": "2015-10-23",
                "kycVerificationFlag": "Y",
                "description": "NO Issues",
                "phoneticDisplayName": "SACAN",
                "lePersonDO": {
                "legalentityIdpk": "11116663666777799d1n",
                "version": 0,
                "createdTs": "2018-01-08T09.39.24.100+0000",
                "updatedTs": "2018-01-08T09.39.24.100+0000",
        "countryOfDomicileRefkey": "356",
        "preferredLanguageRefkey": "1",
        "maritalStatus": null,
        "highestEduQualRefkey": null,
        "isDeceasedFlag": "Y",
        "deceasedDate": null,
        "isHandicappedFlag": null,
        "handicappedSinceDate": null,
        "numberOfDependents": null,
        "numberOfChildren": null,
        "prefereedLanguageRefkey": null,
        "personnamesDOList": [{
            "idPk": null,
            "version": null,
            "legalentityIdpk": null,
            "personnameTypeRefkey": "1",
            "prefixNameRefkey": null,
            "prefixMisc": null,
            "nameOne": "SACHIN",
            "nameTwo": "RAMESH",
            "nameThree": null,
            "nameFour": null,
            "lastName": "TENDULKAR",
            "nickName": "Master Blaster",
            "popularName": "Master Blaster",
            "suffixNameRefkey": 1,
            "suffixMisc": null,
            "nameStandardisedFlag": "N",
            "sourceSystemRefkey": "1"
        }]
        },
        "leSystemKeysRegistryDOList": [{
        "sourceSystemRefkey": "1",
        "referenceId": "112233112",
        "legalentityIdpk": null,
        "statusInSourceRefkey": "1",
        "description": "Sample"
        }]
    }
    }
}
```

```
        "updatedByUser": "rakesh",
        "updatedByTxnId": "326dfd40-157b-4921-b896-55679a46cfa0",
        "genderRefkey": "1",
        "genderRefValue": "MALE",
        "dateOfBirth": "1989-10-23",
        "countryOfBirthRefkey": "356",
        "countryOfBirthRefValue": "IND",
        "countryCitizenshipRefkey": "356",
        "countryCitizenshipRefValue": "IND",
        "countryOfDomicileRefkey": "356",
        "countryOfDomicileRefValue": "IND",
        "isDeceasedFlag": "Y",
        "preferredLanguageRefkey": "1",
        "preferredLanguageRefValue": "ENGLISH",
        "personnamesDOList": [{
            "idPk": "45e1840d-50f6-4e3a-9cac-b3a0662c6ad4",
            "version": 0,
            "createdTs": "2018-01-08T09.39.24.422+0000",
            "updatedTs": "2018-01-08T09.39.24.422+0000",
            "updatedByUser": "rakesh",
            "updatedByTxnId": "326dfd40-157b-4921-b896-
55679a46cfa0",
            "legalentityIdpk": "11116663666777799d1n",
            "personnameTypeRefkey": "1",
            "personnameTypeRefValue": "REGISTERED NAME",
            "nameOne": "SACHIN",
            "nameTwo": "RAMESH",
            "lastName": "TENDULKAR",
            "nickName": "Master Blaster",
            "popularName": "Master Blaster",
            "suffixNameRefkey": "1",
            "suffixNameRefValue": "Sr",
            "nameStandardisedFlag": "N",
            "sourceSystemRefkey": "1",
            "sourceSystemRefValue": "LE SOURCE SYSTEM 1",
            "phoneticNameOne": "SACAN",
            "phoneticNameTwo": "RAN",
            "phoneticLastName": "TANDAL"
        }]
        },
        "leSystemKeysRegistryDOList": [{
        "idPk": "350eb0d0-3531-4a09-843b-bbbfa61604e9",
        "version": 0,
        "createdTs": "2018-01-08T09.39.24.622+0000",
                        "updatedTs": "2018-01-08T09.39.24.622+0000",
                        "updatedByUser": "rakesh",
                        "updatedByTxnId": "326dfd40-157b-4921-b896-55679a46cfa0",
                        "sourceSystemRefkey": "1",
                        "sourceSystemRefValue": "LE SOURCE SYSTEM 1",
                        "referenceId": "112233112",
                        "legalentityIdpk": "11116663666777799d1n",
                        "statusInSourceRefkey": "1",
                        "statusInSourceRefValue": "ACTIVE",
                        "description": "Sample"
                        }]
            }
            }
}
```

Master repository of sample messages

Sample JSON and XML messages are available at githubrepo/resources/Testing location.

Out of the Box Transactions

Base entity services

Base entity services perform CRUD (create, retrieve, update and delete) operations on single database entity.

Composite Services

Composite entity services perform CRUD (create, retrieve, update and delete) operations on multiple database entities as single transaction. Also composite services can have composition of create/ retrieve/ update or delete in single services. e.g. updateLegalEntity service update the legal entity record along with person, corporation, address and identifiers etc. also you may associate new address or identifier through this service itself. Also, search services can also considered as composite services.

Application Configuration Management services

1. createConfigAppPropertiesBase

Description: create a record in CONFIG_APP_PROPERTIES table

Transaction Service Name: createConfigAppPropertiesBase

Request DO: ConfigAppPropertiesDO

Response DO: ConfigAppPropertiesDO

Mandatory Input: key, value

2. createConfigErrorcodeRegistryBase

Description:create a record in CONFIG_ERRORCODE_REGISTRY table

Transaction Service Name: createConfigErrorcodeRegistryBase

Request DO: ConfigErrorcodeRegistryDO

Response DO: ConfigErrorcodeRegistryDO

Mandatory Input: configLanguageCodeKey, errorCode, errorMessage

3. createConfigInquiryLevelsBase

Description: create a record in CONFIG_INQUIRY_LEVELS table.

Transaction Service Name: createConfigInquiryLevelsBase

Request DO: ConfigInquiryLevelsDO

Response DO: ConfigInquiryLevelsDO

Mandatory Input: inquiryLevel, applicableDobj, childDobj

4. createConfigLanguageCodeBase

Description:create a record in CONFIG_LANGUAGE_CODE table.

Transaction Service Name: createConfigLanguageCodeBase

Request DO: ConfigLanguageCodeDO

Response DO: ConfigLanguageCodeDO

Mandatory Input: key, value

5. createConfigTxnRegistryBase

Description: create a record in CONFIG_TXN_REGISTRY table

Transaction Service Name: createConfigTxnRegistryBase

Request DO: ConfigTxnRegistryDO

Response DO: ConfigTxnRegistryDO

Mandatory Input: txnserviceName, txnserviceClass,txnserviceClassmethod

6. findAllConfigErrorcodeRegistryByLanguageCodeBase

Description:find all records in CONFIG_ERRORCODE_REGISTRY table for given language code

Transaction Service Name: findAllConfigErrorcodeRegistryByLanguageCodeBase

Request DO: ConfigErrorcodeRegistryDO

Response DO: ConfigErrorcodeRegistryDO

Mandatory Input: configLanguageCodeKey

inquiryFilter: *ACTIVE/INACTIVE/ALL*

7. findAllConfigInquiryLevelsBase

Description: find all records in CONFIG_INQUIRY_LEVELS table

Transaction Service Name: findAllConfigInquiryLevelsBase

Request DO: ConfigInquiryLevelsDO

Response DO: ConfigInquiryLevelsDO

Mandatory Input: N/A

inquiryFilter: *ACTIVE/INACTIVE/ALL*

8. findAllConfigInquiryLevelsByBusinessKeyBase

Description: find all records in CONFIG_INQUIRY_LEVELS table based on inquiryLevel and applicableDobj

Transaction Service Name: findAllConfigInquiryLevelsByBusinessKeyBase

Request DO: ConfigInquiryLevelsDO

Response DO: ConfigInquiryLevelsDO

Mandatory Input: inquiryLevel and applicableDobj

inquiryFilter: *ACTIVE/INACTIVE/ALL*

9. findAllConfigLanguageCodesBase

Description:find all records in CONFIG_LANGUAGE_CODE table based on idpk

Transaction Service Name: findAllConfigLanguageCodesBase

Request DO: ConfigLanguageCodeDO

Response DO: ConfigLanguageCodeDO

Mandatory Input: N/A

inquiryFilter: *ACTIVE/INACTIVE/ALL*

10. findConfigAppPropertiesByBusinessKeyBase

Description: find a record in CONFIG_APP_PROPERTIES table based on key

Transaction Service Name: findConfigAppPropertiesByBusinessKeyBase

Request DO: ConfigAppPropertiesDO

Response DO: ConfigAppPropertiesDO

Mandatory Input: key

inquiryFilter: *ACTIVE/INACTIVE/ALL*

11. findConfigErrorcodeRegistryByBusinessKeyBase

Description:find a record in CONFIG_ERRORCODE_REGISTRY table based on business key

Transaction Service Name: findConfigErrorcodeRegistryByBusinessKeyBase

Request DO: ConfigErrorcodeRegistryDO

Response DO: ConfigErrorcodeRegistryDO

Mandatory Input: configLanguageCodeKey, errorCode

12. findConfigLanguageCodeByBusinessKeyBase

Description:find a record in CONFIG_LANGUAGE_CODE table based on business key

Transaction Service Name: findConfigLanguageCodeByBusinessKeyBase

Request DO: ConfigLanguageCodeDO

Response DO: ConfigLanguageCodeDO

Mandatory Input: key

13. findConfigTxnRegistryByBusinessKeyBase

Description: find a record in CONFIG_TXN_REGISTRY table based on business key

Transaction Service Name: findConfigTxnRegistryByBusinessKeyBase

Request DO: ConfigTxnRegistryDO

Response DO: ConfigTxnRegistryDO

Mandatory Input: txnserviceName

14. retrieveConfigAppPropertiesBase

Description: retrieve a record in CONFIG_APP_PROPERTIES table based on idpk

Transaction Service Name: retrieveConfigAppPropertiesBase

Request DO: ConfigAppPropertiesDO

Response DO: ConfigAppPropertiesDO

Mandatory Input: idPk

15. retrieveConfigErrorcodeRegistryBase

Description:retrieve a record in CONFIG_ERRORCODE_REGISTRY table based on idpk

Transaction Service Name: retrieveConfigErrorcodeRegistryBase

Request DO: ConfigErrorcodeRegistryDO

Response DO: ConfigErrorcodeRegistryDO

Mandatory Input: idPk

16. retrieveConfigInquiryLevelsBase

Description: retrievea record in CONFIG_INQUIRY_LEVELS table based on idpk

Transaction Service Name: retrieveConfigInquiryLevelsBase

Request DO: ConfigInquiryLevelsDO

Response DO: ConfigInquiryLevelsDO

Mandatory Input: idPk

17. retrieveConfigLanguageCodeBase

Description: retrieve a record in CONFIG_LANGUAGE_CODE table based on idpk

Transaction Service Name: retrieveConfigLanguageCodeBase

Request DO: ConfigLanguageCodeDO

Response DO: ConfigLanguageCodeDO

Mandatory Input: idPk

18. retrieveConfigTxnRegistryBase

Description: retrieve a record in CONFIG_TXN_REGISTRY table based on idpk

Transaction Service Name: retrieveConfigTxnRegistryBase

Request DO: ConfigTxnRegistryDO

Response DO: ConfigTxnRegistryDO

Mandatory Input: idPk

19. updateConfigAppPropertiesBase

Description*:* update a record in CONFIG_APP_PROPERTIES table based on idpk

Transaction Service Name: updateConfigAppPropertiesBase

Request DO: ConfigAppPropertiesDO

Response DO: ConfigAppPropertiesDO

Mandatory Input: idPk and version

20. updateConfigErrorcodeRegistryBase

Description: update a record in CONFIG_ERRORCODE_REGISTRY table based on idpk

Transaction Service Name: updateConfigErrorcodeRegistryBase

Request DO: ConfigErrorcodeRegistryDO

Response DO: ConfigErrorcodeRegistryDO

Mandatory Input: idPk and version

21. updateConfigInquiryLevelsBase

Description: update a record in CONFIG_INQUIRY_LEVELS table based on idpk

Transaction Service Name: updateConfigInquiryLevelsBase

Request DO: ConfigInquiryLevelsDO

Response DO: ConfigInquiryLevelsDO

Mandatory Input: idPk, version

22. updateConfigLanguageCodeBase

Description: update a record in CONFIG_LANGUAGE_CODE table based on idpk

Transaction Service Name: updateConfigLanguageCodeBase

Request DO: ConfigLanguageCodeDO

Response DO: ConfigLanguageCodeDO

Mandatory Input: idPk, version

23. updateConfigTxnRegistryBase

Description: update a record in CONFIG_TXN_REGISTRY table based on idpk

Transaction Service Name: updateConfigTxnRegistryBase

Request DO: ConfigTxnRegistryDO

Response DO: ConfigTxnRegistryDO

Mandatory Input: idPk, version

Authorization Framework services

1. createAuthRolesRegistryBase

Description: create a record in AUTH_ROLES_REGISTRY table

Transaction Service Name: createAuthRolesRegistryBase

Request DO: AuthRolesRegistryDO

Response DO: AuthRolesRegistryDO

Mandatory Input: roleName

2. updateAuthRolesRegistryBase

Description: update a record in AUTH_ROLES_REGISTRY based on idPk

Transaction Service Name: updateAuthRolesRegistryBase

Request DO: AuthRolesRegistryDO

Response DO: AuthRolesRegistryDO

Mandatory Input: idPk, version, roleName

3. retrieveAuthRolesRegistryBase

Description: retrieve a record fromAUTH_ROLES_REGISTRY based on idPk

Transaction Service Name: retrieveAuthRolesRegistryBase

Request DO: AuthRolesRegistryDO

Response DO: AuthRolesRegistryDO

Mandatory Input: idPk

4. findAuthRolesRegistryByBusinessKeyBase

Description: find a record from AUTH_ROLES_REGISTRY based on business key

Transaction Service Name: findAuthRolesRegistryByBusinessKeyBase

Request DO: authRolesRegistryDO

Response DO: authRolesRegistryDO

Mandatory Input: roleName

5. createAuthUserRegistryBase

Description: create a record in AUTH_USER_REGISTRY table

Transaction Service Name: createAuthUserRegistryBase

Request DO: authUserRegistryDO

Response DO: authUserRegistryDO

Mandatory Input: userName

6. updateAuthUserRegistryBase

Description: update a record in AUTH_USER_REGISTRY based on idPk

Transaction Service Name: updateAuthUserRegistryBase

Request DO: authUserRegistryDO

Response DO: authUserRegistryDO

Mandatory Input: idPk, version, userName

7. retrieveAuthUserRegistryBase

Description: retrieve a record from AUTH_USER_REGISTRY based on idPk

Transaction Service Name: retrieveAuthUserRegistryBase

Request DO: authUserRegistryDO

Response DO: authUserRegistryDO

Mandatory Input: idPk

8. findAuthUserRegistryByBusinessKeyBase

Description: find a record from AUTH_USER_REGISTRY based business key

Transaction Service Name: findAuthUserRegistryByBusinessKeyBase

Request DO: authUserRegistryDO

Response DO: authUserRegistryDO

Mandatory Input: userName

9. createAuthUserroleAccesscontrolBase

Description: create a record in AUTH_USERROLE_ACCESSCONTROL table

Transaction Service Name: createAuthUserroleAccesscontrolBase

Request DO: authUserroleAccesscontrolDO

Response DO: authUserroleAccesscontrolDO

Mandatory Input: profileType, authUserRoleRegistryIdpk, configTxnRegistryIdpk

10. updateAuthUserroleAccesscontrolBase

Description: update a record in AUTH_USERROLE_ACCESSCONTROL table based on idPk

Transaction Service Name: updateAuthUserroleAccesscontrolBase

Request DO: authUserroleAccesscontrolDO

Response DO: authUserroleAccesscontrolDO

Mandatory Input: idPk, version, profileType, authUserRoleRegistryIdpk, configTxnRegistryIdpk

11. retrieveAuthUserroleAccesscontrolBase

Description: retrieve a record from AUTH_USERROLE_ACCESSCONTROL based on idPk

Transaction Service Name: retrieveAuthUserroleAccesscontrolBase

Request DO: authUserroleAccesscontrolDO

Response DO: authUserroleAccesscontrolDO

Mandatory Input: idPk

12. findAuthUserroleAccesscontrolByBusinessKeyBase

Description: find a record from AUTH_USERROLE_ACCESSCONTROL table based on business key

Transaction Service Name: findAuthUserroleAccesscontrolByBusinessKeyBase

Request DO: authUserroleAccesscontrolDO

Response DO: authUserroleAccesscontrolDO

Mandatory Input: profileType, authUserRoleRegistryIdpk, configTxnRegistryIdpk

13. findAuthUserroleAccesscontrolByRegistryIdpkBase

Description: find a record from AUTH_USERROLE_ACCESSCONTROL based on business key

Transaction Service Name: findAuthUserroleAccesscontrolByRegistryIdpkBase

Request DO: authUserroleAccesscontrolDO

Response DO: *authUserroleAccesscontrolDO*

Mandatory Input: profileType, authUserRoleRegistryIdpk

14. createAuthUserRoleAssocBase

Description: create a record in AUTH_USER_ROLE_ASSOC table

Transaction Service Name: createAuthUserRoleAssocBase

Request DO: authUserRoleAssocDO

Response DO: authUserRoleAssocDO

Mandatory Input: authRolesRegistryIdpk, authUserRegistryIdpk

15. updateAuthUserRoleAssocBase

Description: update a record in AUTH_USER_ROLE_ASSOC based on idPk

Transaction Service Name: *updateAuthUserRoleAssocBase*

Request DO: *authUserRoleAssocDO*

Response DO: *authUserRoleAssocDO*

Mandatory Input: *idPk, version , authRolesRegistryIdpk, authUserRegistryIdpk*

16. retrieveAuthUserRoleAssocBase

Description: retrieve a record from AUTH_USER_ROLE_ASSOC based on idPk

Transaction Service Name: retrieveAuthUserRoleAssocBase

Request DO: authUserRoleAssocDO

Response DO: authUserRoleAssocDO

Mandatory Input: idPk

17. findAuthUserRoleAssocByBusinessKeyBase

Description: find a record from AUTH_USER_ROLE_ASSOC based on business key

Transaction Service Name: findAuthUserRoleAssocByBusinessKeyBase

Request DO: authUserRoleAssocDO

Response DO: authUserRoleAssocDO

Mandatory Input: authRolesRegistryIdpk and authUserRegistryIdpk

18. findAllRecordsByAuthUserRegistryIdpkBase

Description: find AUTH_USER_ROLE_ASSOC based on authUserRegistryIdpk

Transaction Service Name: findAllRecordsByAuthUserRegistryIdpkBase

Request DO: authUserRoleAssocDO

Response DO: authUserRoleAssocDO

Mandatory Input: authUserRegistryIdpk

19. findAllRecordsByAuthRolesRegistryIdpkBase

Description: find all records from AUTH_USER_ROLE_ASSOC based on authRolesRegistryIdpk

Transaction Service Name: findAllRecordsByAuthRolesRegistryIdpkBase

Request DO: authUserRoleAssocDO

Response DO: authUserRoleAssocDO

Mandatory Input: authRolesRegistryIdpk

Batch Framework Management services

1. createBatchEntityToProcessBase

Description: create a record in BATCH_ENTITY_TO_PROCESS table

Transaction Service Name: createBatchEntityToProcessBase

Request DO: batchEntityToProcessDO

Response DO: batchEntityToProcessDO

Mandatory Input: entityObjectTypeRefkey, entityIdpk, batchProposedActionRefkey, batchActionStatusRefkey, entryMadeBySubsystemName

2. updateBatchEntityToProcessBase

Description: update a record in BATCH_ENTITY_TO_PROCESS table based on idPk

Transaction Service Name: updateBatchEntityToProcessBase

Request DO: batchEntityToProcessDO

Response DO: batchEntityToProcessDO

Mandatory Input: idPk, version

3. retrieveBatchEntityToProcessBase

Description: retrieve a record from BATCH_ENTITY_TO_PROCESS table based on idPk

Transaction Service Name: retrieveBatchEntityToProcessBase

Request DO: batchEntityToProcessDO

Response DO: batchEntityToProcessDO

Mandatory Input: idPk

Match And Merge Framework Management services

1. **createInactiveLeRegistryBase**

   Description: create a record in INACTIVE_LE_REGISTRY table

   Transaction Service Name: createInactiveLeRegistryBase

   Request DO: inactiveLeRegistryDO

   Response DO: inactiveLeRegistryDO

   Mandatory Input: legalentityIdpk, inactivatedTs, inactivationReasonRefkey

2. **createMatchCandidateLeRegistryBase**

   Description: create a record in MATCH_CANDIDATE_LE_REGISTRY table

   Transaction Service Name: createMatchCandidateLeRegistryBase

   Request DO: matchCandidateLeRegistryDO

   Response DO: matchCandidateLeRegistryDO

   Mandatory Input: legalentityIdpk, candidateLegalentityidpk, matchPattern, matchProposedActionRefkey, matchActionstatusRefkey

3. **createMatchMergedLeAssocBase**

   Description: create a record in MATCH_MERGED_LE_ASSOC table

   Transaction Service Name: createMatchMergedLeAssocBase

   Request DO: matchMergedLeAssocDO

   Response DO: matchMergedLeAssocDO

   Mandatory Input: survivorLegalentityIdpk, mergedLegalentityIdpk, mergeReasonRefkey

1. **createRefMatchScoreBase**

   Description: create a record in REF_MATCH_SCORE reference table.

   Transaction Service Name: createRefMatchScoreBase

   Request DO: refMatchScoreDO

   Response DO:refMatchScoreDO

   Mandatory Input: matchEntityObjectName, matchAttrPattern, matchResultRefkey, matchProposedActionRefkey

4. **createRefMatchThresholdBase**

   Description: create a record in REF_MATCH_THRESHOLD table.

   Transaction Service Name: createRefMatchThresholdBase

   Request DO: refMatchThresholdDO

   Response DO: refMatchThresholdDO

   Mandatory Input: attrBlockName, matchThreshold, decayThresholdInDays, decayPercentage, maxDecayPercentage

5. **findAllRefMatchScoreByMatchEntityObjectName**

   Description: retrieve all records in REF_MATCH_SCORE reference table based on entity object name.

   Transaction Service Name: findAllRefMatchScoreByMatchEntityObjectName

   Request DO: refMatchScoreDO

   Response DO: refMatchScoreDO

   Mandatory Input: matchEntityObjectName

6. **findAllRefMatchThresholdBase**

   Description: find all records from REF_MATCH_THRESHOLD table.

   Transaction Service Name: createRefMatchThresholdBase

   Request DO: refMatchThresholdDO

   Response DO: refMatchThresholdDOList

   Mandatory Input: NA

7. **findRefMatchScoreByBusinessKeyBase**

   Description: find a record in REF_MATCH_SCORE reference table based on business key

   Transaction Service Name: findRefMatchScoreByBusinessKeyBase

   Request DO: refMatchScoreDO

   Response DO: refMatchScoreDO

   Mandatory Input: matchEntityObjectName, matchAttrPattern

8. **findRefMatchThresholdByBusinessKeyBase**

   Description: find a record from REF_MATCH_THRESHOLD table based on business key.

   Transaction Service Name: findRefMatchThresholdByBusinessKeyBase

   Request DO: refMatchThresholdDO

   Response DO: refMatchThresholdDO

   Mandatory Input: attrBlockName

9. **retrieveInactiveLeRegistryBase**

   Description: retrieve a record in INACTIVE_LE_REGISTRY table based on idPk

   Transaction Service Name: retrieveInactiveLeRegistryBase

   Request DO: inactiveLeRegistryDO

   Response DO: inactiveLeRegistryDO

   Mandatory Input: idPk

10. **retrieveMatchCandidateLeRegistryBase**

   Description: retrieve a record in MATCH_CANDIDATE_LE_REGISTRY table based on idPk

   Transaction Service Name: retrieveMatchCandidateLeRegistryBase

   Request DO: matchCandidateLeRegistryDO

   Response DO: matchCandidateLeRegistryDO

   Mandatory Input: idPk

11. **retrieveMatchMergedLeAssocBase**

   Description: retrieve a record in MATCH_MERGED_LE_ASSOC table based on idPk

   Transaction Service Name: retrieveMatchMergedLeAssocBase

   Request DO: matchMergedLeAssocDO

   Response DO: matchMergedLeAssocDO

   Mandatory Input: idPk

12. **retrieveRefMatchScoreBase**

Description: retrieve a record in REF_MATCH_SCORE reference table based on idpk

Transaction Service Name: retrieveRefMatchScoreBase

Request DO: refMatchScoreDO

Response DO: refMatchScoreDO

Mandatory Input: idPk

13. **retrieveRefMatchThresholdBase**

Description: retrieve a record in REF_MATCH_THRESHOLD table based on idPk

Transaction Service Name: retrieveRefMatchThresholdBase

Request DO: refMatchThresholdDO

Response DO: refMatchThresholdDO

Mandatory Input: idPk

14. **updateInactiveLeRegistryBase**

Description: update a record in INACTIVE_LE_REGISTRY table based on idPk

Transaction Service Name: updateInactiveLeRegistryBase

Request DO: inactiveLeRegistryDO

Response DO: inactiveLeRegistryDO

Mandatory Input: idPk, version

15. **updateMatchCandidateLeRegistryBase**

    Description: update a record in MATCH_CANDIDATE_LE_REGISTRY table based on idPk

    Transaction Service Name: updateMatchCandidateLeRegistryBase

    Request DO: matchCandidateLeRegistryDO

    Response DO: matchCandidateLeRegistryDO

    Mandatory Input: idPk

16. **updateMatchMergedLeAssocBase**

    Description: update a record in MATCH_MERGED_LE_ASSOC table based on idPk

    Transaction Service Name: updateMatchMergedLeAssocBase

    Request DO: matchMergedLeAssocDO

    Response DO: matchMergedLeAssocDO

    Mandatory Input: idPk, version

17. **updateRefMatchScoreBase**

    Description: update a record in REF_MATCH_SCORE reference table based on idpk

    Transaction Service Name: updateRefMatchScoreBase

    Request DO: refMatchScoreDO

    Response DO: refMatchScoreDO

    Mandatory Input: idPk

18. **updateRefMatchThresholdBase**

Description: retrieve a record in REF_MATCH_THRESHOLD table based on idPk

Transaction Service Name: updateRefMatchThresholdBase

Request DO: refMatchThresholdDO

Response DO: refMatchThresholdDO

Mandatory Input: idPk

Data Table Management Services (Base)

1. createAccountAddressAssocBase

Description: create a record in ACCOUNT_ADDRESS_ASSOC table

Transaction Service Name: createAccountAddressAssocBase

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDO

Mandatory Input: accountIdpk, addressIdpk, addressTypeRefkey, addressSubtypeRefkey

2. createAccountBase

Description: create a record in ACCOUNT table

Transaction Service Name: createAccountBase

Request DO: accountDO

Response DO: accountDO

Mandatory Input: accountName

3. createAccountPhoneAssocBase

Description: create a record in ACCOUNT_PHONE_ASSOC table

Transaction Service Name: createAccountPhoneAssocBase

Request DO: AccountPhoneAssocDO

Response DO: AccountPhoneAssocDO

Mandatory Input: accountIdpk, phoneTypeRefkey, phoneNumber

4. createAddressBase

Description: create a record in ADDRESS table

Transaction Service Name: createAddressBase

Request DO: AddressDO

Response DO: AddressDO

Mandatory Input: addressLineOne, countryRefkey, city

5. createCorporationnamesBase

Description: create a record in CORPORATIONNAMES table

Transaction Service Name: createCorporationnamesBase

Request DO: corporationnamesDO

Response DO: corporationnamesDO

Mandatory Input: legalEntityIdPk, corporationNameTypeRefkey, corporationName

6. createEntityGroupAssocBase

Description: create a record in ENTITY_GROUP_ASSOC table

Transaction Service Name: createEntityGroupAssocBase

Request DO: entityGroupAssocDO

Response DO: entityGroupAssocDO

Mandatory Input: entityObjectTypeRefkey, entityIdpk, entityGroupIdpk, assocTypeRefkey

7. createEntityGroupBase

Description: create a record in ENTITY_GROUP table

Transaction Service Name: createEntityGroupBase

Request DO: entityGroupDO

Response DO: entityGroupDO

Mandatory Input: groupTypeRefkey, groupName

8. createLeAccountAssocBase

Description: create a record in LE_ACCOUNT_ASSOC table.

Transaction Service Name: createLeAccountAssocBase

Request DO: leAccountAssocDO

Response DO: leAccountAssocDO

Mandatory Input: legalEntityIdPk, roletypeRefkey, accountIdpk

9. createLeAddressAssocBase

Description: create a record in LE_ADDRESS_ASSOC table

Transaction Service Name: createLeAddressAssocBase

Request DO: leAddressAssocDO

Response DO: leAddressAssocDO

Mandatory Input: legalEntityIdPk, addressIdpk, addressTypeRefkey, addressSubtypeRefkey

10. createLeCorporationBase

Description: create a record in LE_CORPORATION table

Transaction Service Name: createLeCorporationBase

Request DO: LeCorporationDO

Response DO: LeCorporationDO

Mandatory Input: legalentityIdpk

11. createLeIdentifierKycRegistryBase

Description: create a record in LE_IDENTIFIER_KYC_REGISTRY table

Transaction Service Name: createLeIdentifierKycRegistryBase

Request DO: leIdentifierKycRegistryDO

Response DO: leIdentifierKycRegistryDO

Mandatory Input: legalEntityIdPk, identificationTypeRefkey, identificationNumber

12. createLePersonBase

Description: create a record in LE_PERSON table

Transaction Service Name: createLePersonBase

Request DO: lePersonDO

Response DO: lePersonDO

Mandatory Input: genderRefkey

13. createLePhoneAssocBase

Description: create a record in LE_PHONE_ASSOC table

Transaction Service Name: createLePhoneAssocBase

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDO

Mandatory Input: LegalEntityIdPk, phoneTypeRefkey, phoneNumber

14. createLePreferencesBase

Description: create a record in LE_PREFERENCES table

Transaction Service Name: createLePreferencesBase

Request DO: lePreferencesDO

Response DO: lePreferencesDO

Mandatory Input: legalEntityIdPk, preferenceTypeRefkey

15. createLePropertyAssocBase

Description: create a record in LE_PROPERTY_ASSOC table

Transaction Service Name: createLePropertyAssocBase

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input: legalEntityIdPk, propertyIdpk, propertyLeReltypeRefkey

16. createLeSystemKeysRegistryBase

Description: create a record in LE_SYSTEM_KEYS_REGISTRY table

Transaction Service Name: createLeSystemKeysRegistryBase

Request DO: leSystemKeysRegistryDO

Response DO: leSystemKeysRegistryDO

Mandatory Input: LegalEntityIdPk, sourceSystemRefkey, referenceId

17. createLeToLeRelationshipBase

Description: create a record in LE_TO_LE_RELATIONSHIP table

Transaction Service Name: createLeToLeRelationshipBase

Request DO: leToLeRelationship

Response DO: leToLeRelationship

Mandatory Input: fromLegalentityIdpk, toLegalentityIdpk, leRelationshipTypeRefkey

18. createLeVehicleAssocBase

Description: create a record in LE_VEHICLE_ASSOC table

Transaction Service Name: createLeVehicleAssocBase

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input: legalEntityIdPk, leRoletypeRefkey

19. createLegalentityBase

Description: create a record in LEGALENTITY table

Transaction Service Name: createLegalentityBase

Request DO: legalentityDO

Response DO: legalentityDO

Mandatory Input: displayName, entityObjectTypeRefkey

20. createMiscellaneousInfoBase

Description: create a record in MISCELLANEOUS_INFO table

Transaction Service Name: createMiscellaneousInfoBase

Request DO: miscellaneousInfoDO

Response DO: miscellaneousInfoDO

Mandatory Input: entityObjectTypeRefkey, entityIdpk, name1, value1

21. createPersonnamesBase

Description: create a record in PERSONNAMES table

Transaction Service Name: createPersonnamesBase

Request DO: personnamesDO

Response DO: personnamesDO

Mandatory Input: LegalEntityIdPk, personnameTypeRefkey, nameOne

22. createPhoneStandardizedBase

Description: create a record in PHONE_STANDARDIZED table

Transaction Service Name: createPhoneStandardizedBase

Request DO: phoneStandardizedDO

Response DO: phoneStandardizedDO

Mandatory Input: N/A

23. createPropertyBase

Description: create a record in PROPERTY table

Transaction Service Name: createPropertyBase

Request DO: propertyDO

Response DO: propertyDO

Mandatory Input: propertyName

24. createVehicleBase

Description: create a record in VEHICLE table

Transaction Service Name: createVehicleBase

Request DO: vehicleDO

Response DO: vehicleDO

Mandatory Input: vinNumber

25. findAllAccountAddressAssocByAccountIdPkBase

Description: find all records in ACCOUNT_ADDRESS_ASSOC table based on account idPk

Transaction Service Name: findAllAccountAddressAssocByAccountIdPkBase

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDOList

Mandatory Input: accountIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

26. findAllAccountPhoneAssocByAccountIdPkBase

Description: create a record in ACCOUNT_PHONE_ASSOC table

Transaction Service Name: findAllAccountPhoneAssocByAccountIdPkBase

Request DO: accountPhoneAssocDO

Response DO: accountPhoneAssocDOList

Mandatory Input: accountIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

27. findAllMiscellaneousInfoByEntityIdPkBase

Description: find all records in MISCELLANEOUS_INFO table based on entity id

Transaction Service Name: findAllMiscellaneousInfoByEntityIdPkBase

Request DO: miscellaneousInfoDO

Response DO: miscellaneousInfoDOList

Mandatory Input: entityIdpk, entityObjectTypeRefkey

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

28. findCorporationnamesByLegalEntityIdBase

Description: find all records in CORPORATIONNAMES table based on legal entity Idpk

Transaction Service Name: findCorporationnamesByLegalEntityIdBase

Request DO: corporationnamesDO

Response DO: corporationnamesDOList

Mandatory Input: legalentityIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

29. findEntityGroupAssocByEntityGroupIdpkBase

Description: find all records in ENTITY_GROUP_ASSOC table based on entity group idPk

Transaction Service Name: findEntityGroupAssocByEntityGroupIdpkBase

Request DO: entityGroupAssocDO

Response DO: entityGroupAssocDOList

Mandatory Input: entityGroupIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

30. findEntityGroupAssocByEntityIdPkBase

Description: find all records in ENTITY_GROUP_ASSOC table based on entity idPk

Transaction Service Name: findEntityGroupAssocByEntityIdPkBase

Request DO: entityGroupAssocDO

Response DO: entityGroupAssocDOList

Mandatory Input: entityIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

31. findLeAccountAssocByAccountIdPkBase

Description: find all records in LE_ACCOUNT_ASSOC table based on account idPk

Transaction Service Name: findLeAccountAssocByAccountIdPkBase

Request DO: leAccountAssocDO

Response DO: leAccountAssocDOList

Mandatory Input: accountIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

32. findLeAccountAssocByLegalEntityIdPkBase

Description: find all records in LE_ACCOUNT_ASSOC table based on legal entity idPk

Transaction Service Name: findLeAccountAssocByAccountIdPkBase

Request DO: leAccountAssocDO

Response DO: leAccountAssocDOList

Mandatory Input: legalEntityIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

33. findLeAddressByLegalEntityIdPkBase

Description: find all records in LE_ADDRESS_ASSOC table based on legal entity idPk

Transaction Service Name: findLeAddressByLegalEntityIdPkBase

Request DO: leAddressAssocDO

Response DO: leAddressAssocDOList

Mandatory Input: legalentityIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

34. findLeIdentifierKycRegistryByLegalEntityIdPkBase

Description: find all records in LE_IDENTIFIER_KYC_REGISTRY table based on legal entity idPk

Transaction Service Name: findLeIdentifierKycRegistryByLegalEntityIdPkBase

Request DO: leIdentifierKycRegistryDO

Response DO: leIdentifierKycRegistryDOList

Mandatory Input: legalentityIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

35. findLePhoneAssocByLegalEntityIdPkBase

Description: find all records in LE_PHONE_ASSOC table based on legal entity idPk

Transaction Service Name: findLePhoneAssocByLegalEntityIdPkBase

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDOList

Mandatory Input: idPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

36. findLePreferencesByLegalEntityIdPkBase

Description: find all records from LE_PREFERENCES table based on legal entity idPk

Transaction Service Name: findLePreferencesByLegalEntityIdPkBase

Request DO: lePreferencesDO

Response DO: lePreferencesDOList

Mandatory Input: legalEntityIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

37. findLePropertyAssocByLegalEntityIdPkBase

Description: find all records in LE_PROPERTY_ASSOC table based legal entity idPk

Transaction Service Name: indLePropertyAssocByLegalEntityIdPkBase

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDOList

Mandatory Input: legalEntityIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

38. findLeSystemKeysRegistryByLegalEntityIdPkBase

Description: : find all records in LE_SYSTEM_KEYS_REGISTRY table based legal entity idPk

Transaction Service Name: findLeSystemKeysRegistryByLegalEntityIdPkBase

Request DO: leSystemKeysRegistryDO

Response DO: leSystemKeysRegistryDOList

Mandatory Input: legalEntityIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

39. findLeToLeRelationshipByLegalentityIdpkBase

Description: : find all records in LE_TO_LE_RELATIONSHIP table based legal entity idPk. The legal entity needs to be provided in reqLeToLeRelationshipDO.fromLegalentityIdpk attribute. However the search happens on both the attributes fromLegalentityIdpk or toLegalentityIdpk in database.

Transaction Service Name: findLeToLeRelationshipByLegalentityIdpkBase

Request DO: leToLeRelationship

Response DO: leToLeRelationshipList

Mandatory Input: fromLegalentityIdpk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

40. findLeVehicleAssocByLegalEntityIdPkBase

Description: find all records in LE_VEHICLE_ASSOC table based legal entity idPk

Transaction Service Name: findLeVehicleAssocByLegalEntityIdPkBase

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDOList

Mandatory Input: legalEntityIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

41. findPersonnamesByLegalEntityIdBase

Description: find all records in PERSONNAMES table based legal entity idPk

Transaction Service Name: findPersonnamesByLegalEntityIdBase

Request DO: personnamesDO

Response DO: personnamesDOList

Mandatory Input: legalEntityIdPk

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

42. retrieveAccountAddressAssocBase

Description: retrieve a record from ACCOUNT_ADDRESS_ASSOC table based idPk

Transaction Service Name: retrieveAccountAddressAssocBase

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDO

Mandatory Input: Idpk

43. retrieveAccountBase

Description: *retrieve* a record in ACCOUNT table *based on idPk*

Transaction Service Name: *retrieve*AccountBase

Request DO: *accountDO*

Response DO: *accountDO*

Mandatory Input: *idPk*

44. retrieveAccountPhoneAssocBase

Description: create a record in ACCOUNT_PHONE_ASSOC table based on idPk

Transaction Service Name: retrieveAccountPhoneAssocBase

Request DO: AccountPhoneAssocDO

Response DO: AccountPhoneAssocDO

Mandatory Input: idPk

45. retrieveAddressBase

Description: retrieve a record from ADDRESS table based on idPk

Transaction Service Name: retrieveAddressBase

Request DO: AddressDO

Response DO: AddressDO

Mandatory Input: idPk

46. retrieveCorporationnamesBase

Description: retrieve a record in CORPORATIONNAMES table based on idPk

Transaction Service Name: retrieveCorporationnamesBase

Request DO: corporationnamesDO

Response DO: corporationnamesDO

Mandatory Input: idPk

47. retrieveEntityGroupAssocBase

Description: retrieve a record in ENTITY_GROUP_ASSOC table based on idPk

Transaction Service Name: retrieveEntityGroupAssocBase

Request DO: entityGroupAssocDO

Response DO: entityGroupAssocDO

Mandatory Input: idPk

48. **retrieveEntityGroupBase**

Description: retrieve a record in ENTITY_GROUP table based on idPk

Transaction Service Name: retrieveEntityGroupBase

Request DO: entityGroupDO

Response DO: entityGroupDO

Mandatory Input: idPk

49. **retrieveLeAccountAssocBase**

Description: retrieve a record from LE_ACCOUNT_ASSOC table based on idPk

Transaction Service Name: retrieveLeAccountAssocBase

Request DO: leAccountAssocDO

Response DO: leAccountAssocDO

Mandatory Input: idPk

50. **retrieveLeAddressAssocBase**

Description: retrieve a record from LE_ADDRESS_ASSOC table based on idPk

Transaction Service Name: retrieveLeAddressAssocBase

Request DO: leAddressAssocDO

Response DO: leAddressAssocDO

Mandatory Input: idPk

51. **retrieveLeCorporationBase**

Description: retrieve a record from LE_CORPORATION table based on idPk

Transaction Service Name: retrieveLeCorporationBase

Request DO: LeCorporationDO

Response DO: LeCorporationDO

Mandatory Input: idPk

52. retrieveLeIdentifierKycRegistryBase

Description: retrieve a record from LE_IDENTIFIER_KYC_REGISTRY table based on idPk

Transaction Service Name: retrieveLeIdentifierKycRegistryBase

Request DO: leIdentifierKycRegistryDO

Response DO: leIdentifierKycRegistryDO

Mandatory Input: idPk

53. retrieveLePersonBase

Description: retrieve a record from LE_PERSON table based on idPk

Transaction Service Name: retrieveLePersonBase

Request DO: lePersonDO

Response DO: lePersonDO

Mandatory Input: idPk

54. retrieveLePhoneAssocBase

Description: retrieve a record in LE_PHONE_ASSOC table based on idPk

Transaction Service Name: retrieveLePhoneAssocBase

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDO

Mandatory Input: idPk

55. retrieveLePreferencesBase

Description: retrieve a record in LE_PREFERENCES table based on idPk

Transaction Service Name: retrieveLePreferencesBase

Request DO: lePreferencesDO

Response DO: lePreferencesDO

Mandatory Input: idPk

56. retrieveLePropertyAssocBase

Description: retrieve a record from LE_PROPERTY_ASSOC table based idPk

Transaction Service Name: retrieveLePropertyAssocBase

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input: idPk

57. retrieveLeSystemKeysRegistryBase

Description: retrieve a record from LE_SYSTEM_KEYS_REGISTRY table based on idPk

Transaction Service Name: retrieveLeSystemKeysRegistryBase

Request DO: leSystemKeysRegistryDO

Response DO: leSystemKeysRegistryDO

Mandatory Input: idPk

58. retrieveLeToLeRelationshipBase

Description: retrieve a record from LE_TO_LE_RELATIONSHIP table based on idPk

Transaction Service Name: retrieveLeToLeRelationshipBase

Request DO: leToLeRelationship

Response DO: leToLeRelationship

Mandatory Input: idPk

59. retrieveLeVehicleAssocBase

Description: retrieve a record from LE_VEHICLE_ASSOC table based on idPk

Transaction Service Name: retrieveLeVehicleAssocBase

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input: idPk

60. retrieveLegalentityBase

Description: retrieve a record from LEGALENTITY table based on idPk

Transaction Service Name: retrieveLegalentityBase

Request DO: legalentityDO

Response DO: legalentityDO

Mandatory Input: idPk

61. retrieveMiscellaneousInfoBase

Description: retrieve a record from MISCELLANEOUS_INFO table based on idPk

Transaction Service Name: retrieveMiscellaneousInfoBase

Request DO: miscellaneousInfoDO

Response DO: miscellaneousInfoDO

Mandatory Input: idPk

62. retrievePersonnamesBase

Description: retrieve a record from PERSONNAMES table based on idPk

Transaction Service Name: retrievePersonnamesBase

Request DO: personnamesDO

Response DO: personnamesDO

Mandatory Input: idPk

63. retrievePhoneStandardizedBase

Description: retrieve a record from PHONE_STANDARDIZED table based on idPk

Transaction Service Name: retrievePhoneStandardizedBase

Request DO: phoneStandardizedDO

Response DO: phoneStandardizedDO

Mandatory Input: idPk

64. retrievePropertyBase

Description: retrieve a record from PROPERTY table based on idPk

Transaction Service Name: retrievePropertyBase

Request DO: propertyDO

Response DO: propertyDO

Mandatory Input: idPk

65. retrieveVehicleBase

Description: retrieve a record from VEHICLE table based on idPk

Transaction Service Name: retrieveVehicleBase

Request DO: vehicleDO

Response DO: vehicleDO

Mandatory Input: idPk

66. updateAccountAddressAssocBase

Description: update a record from ACCOUNT_ADDRESS_ASSOC table based idPk

Transaction Service Name: updateAccountAddressAssocBase

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDO

Mandatory Input: Idpk, version

67. updateAccountBase

Description: *update* a record in ACCOUNT table *based on idPk*

Transaction Service Name: *update*AccountBase

Request DO: *accountDO*

Response DO: *accountDO*

Mandatory Input: *idPk, version*

68. updateAccountPhoneAssocBase

Description: update a record in ACCOUNT_PHONE_ASSOC table based on idPk

Transaction Service Name: updateAccountPhoneAssocBase

Request DO: AccountPhoneAssocDO

Response DO: AccountPhoneAssocDO

Mandatory Input: idPk, version

69. updateAddressBase

Description: update a record from ADDRESS table based on idPk

Transaction Service Name: updateAddressBase

Request DO: AddressDO

Response DO: AddressDO

Mandatory Input: idPk, version

70. updateCorporationnamesBase

Description: update a record in CORPORATIONNAMES table based on idPk

Transaction Service Name: updateCorporationnamesBase

Request DO: corporationnamesDO

Response DO: corporationnamesDO

Mandatory Input: idPk, version

71. updateEntityGroupAssocBase

Description: update a record in ENTITY_GROUP_ASSOC table based on idPk

Transaction Service Name: updateEntityGroupAssocBase

Request DO: entityGroupAssocDO

Response DO: entityGroupAssocDO

Mandatory Input: idPk, version

72. updateEntityGroupBase

Description: update a record in ENTITY_GROUP table based on idPk

Transaction Service Name: updateEntityGroupBase

Request DO: entityGroupDO

Response DO: entityGroupDO

Mandatory Input: idPk, version

73. updateLeAccountAssocBase

Description: update a record from LE_ACCOUNT_ASSOC table based on idPk

Transaction Service Name: updateLeAccountAssocBase

Request DO: leAccountAssocDO

Response DO: leAccountAssocDO

Mandatory Input: idPk, version

74. updateLeAddressAssocBase

Description: update a record from LE_ADDRESS_ASSOC table based on idPk

Transaction Service Name: updateLeAddressAssocBase

Request DO: leAddressAssocDO

Response DO: leAddressAssocDO

Mandatory Input: idPk, version

75. updateLeCorporationBase

Description: update a record in LE_CORPORATION table based on idPk

Transaction Service Name: updateLeCorporationBase

Request DO: LeCorporationDO

Response DO: LeCorporationDO

Mandatory Input: idPk, version,legalentityIdpk

76. updateLeIdentifierKycRegistryBase

Description: update a record from LE_IDENTIFIER_KYC_REGISTRY table based on idPk

Transaction Service Name: updateLeIdentifierKycRegistryBase

Request DO: leIdentifierKycRegistryDO

Response DO: leIdentifierKycRegistryDO

Mandatory Input: idPk, version

77. updateLePersonBase

Description: update a record from LE_PERSON table based on idPk

Transaction Service Name: updateLePersonBase

Request DO: lePersonDO

Response DO: lePersonDO

Mandatory Input: idPk, version, legalentityIdpk

78. updateLePhoneAssocBase

Description: update a record in LE_PHONE_ASSOC table based on idPk

Transaction Service Name: updateLePhoneAssocBase

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDO

Mandatory Input: idPk, version

79. updateLePreferencesBase

Description: update a record in LE_PREFERENCES table based on idPk

Transaction Service Name: updateLePreferencesBase

Request DO: lePreferencesDO

Response DO: lePreferencesDO

Mandatory Input: idPk, version

80. updateLePropertyAssocBase

Description: update a record from LE_PROPERTY_ASSOC table based idPk

Transaction Service Name: updateLePropertyAssocBase

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input: idPk, version

81. updateLeSystemKeysRegistryBase

Description: update a record in LE_SYSTEM_KEYS_REGISTRY table based on idPk

Transaction Service Name: updateLeSystemKeysRegistryBase

Request DO: leSystemKeysRegistryDO

Response DO: leSystemKeysRegistryDO

Mandatory Input: idPk, version

82. updateLeToLeRelationshipBase

Description: update a record in LE_TO_LE_RELATIONSHIP table based on idPk

Transaction Service Name: updateLeToLeRelationshipBase

Request DO: leToLeRelationship

Response DO: leToLeRelationship

Mandatory Input: idPk, version

83. updateLeVehicleAssocBase

Description: update a record in LE_VEHICLE_ASSOC table based on idPk

Transaction Service Name: updateLeVehicleAssocBase

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input: idPk

84. updateLegalentityBase

Description: update a record in LEGALENTITY table based on idPk

Transaction Service Name: updateLegalentityBase

Request DO: legalentityDO

Response DO: legalentityDO

Mandatory Input: idPk, version

85. updateMiscellaneousInfoBase

Description: update a record in MISCELLANEOUS_INFO table based on idPk

Transaction Service Name: updateMiscellaneousInfoBase

Request DO: miscellaneousInfoDO

Response DO: miscellaneousInfoDO

Mandatory Input: idPk, version

86. updatePersonnamesBase

Description: update a record in PERSONNAMES table based on idPk

Transaction Service Name: updatePersonnamesBase

Request DO: personnamesDO

Response DO: personnamesDO

Mandatory Input: idPk, version

87. updatePhoneStandardizedBase

Description: update a record in PHONE_STANDARDIZED table based on idPk

Transaction Service Name: updatePhoneStandardizedBase

Request DO: phoneStandardizedDO

Response DO: phoneStandardizedDO

Mandatory Input: idPk, version

88. updatePropertyBase

Description: update a record in PROPERTY table based on idPk

Transaction Service Name: updatePropertyBase

Request DO: propertyDO

Response DO: propertyDO

Mandatory Input: idPk, version

89. updateVehicleBase

Description: update a record in VEHICLE table based on idPk

Transaction Service Name: updateVehicleBase

Request DO: vehicleDO

Response DO: vehicleDO

Mandatory Input: idPk, version

Data Table Management Services (Composite)

1. searchAuthRoles

Description: This service returns the roles based on username and/or role name. This transaction search the database as per below condition

1. If username and userRole are present in the request then this transaction will return the result only if below conditions are satisfied
   - ✓ The given user must be present in AUTH_USER_REGISTRY table.
   - ✓ The given role must be present in AUTH_ROLES_REGISTRY table.
   - ✓ The user must be associated with the given role in AUTH_USER_ROLE_ASSOC table.
   - ✓ The result will have the rows from AUTH_ROLES_REGISTRY table for given user name and role name combination.
2. If userRole is present and username is null in the request then this transaction will return the result only if below conditions are satisfied
   - ✓ The given role must be present in AUTH_ROLES_REGISTRY table.
   - ✓ The result will have the rows from AUTH_ROLES_REGISTRY table for given role.
3. If username is present and userRole is null in the request then this transaction will return the result only if below conditions are satisfied
   - ✓ The given user must be present in AUTH_USER_REGISTRY table.
   - ✓ The result will have the rows from AUTH_ROLES_REGISTRY table to which the user is associated in AUTH_USER_ROLE_ASSOC table. i.e. all the roles to which a user is associated with will be returned.

Transaction Service Name: searchAuthRoles

Request DO: searchAuthAccessControlDO

Response DO: authRolesRegistryDOList (empty list if no records found)

Mandatory Input: rolename or username

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

Wildcard Support: Yes. % can be used for wildcard searches

Use Case –

1. To check if given userName and roleName combination is valid
2. To check if roleName is valid
3. To get all the roles associated with a valid user

Special Notes – This service is used to in authorization framework.

2. searchAuthUsers

Description: This service returns the roles based on username and/or role name. This transaction search the database as per below condition

4. If username and userRole are present in the request then this transaction will return the result only if below conditions are satisfied
   - ✓ The given user must be present in AUTH_USER_REGISTRY table.
   - ✓ The given role must be present in AUTH_ROLES_REGISTRY table.
   - ✓ The user must be associated with the given role in AUTH_USER_ROLE_ASSOC table.
   - ✓ The result will have the rows from AUTH_USER_REGISTRY table for given user name and role name combination.
5. If userRole is present and username is null in the request then this transaction will return the result only if below conditions are satisfied
   - ✓ The given role must be present in AUTH_ROLES_REGISTRY table.
   - ✓ The result will have the rows from AUTH_USER_REGISTRY table for given role.
6. If username is present and userRole is null in the request then this transaction will return the result only if below conditions are satisfied
   - ✓ The given user must be present in AUTH_USER_REGISTRY table.
   - ✓ The result will have the rows from AUTH_USER_REGISTRY table to which the user is associated in AUTH_USER_ROLE_ASSOC table. i.e. all the users to which a role is associated with will be returned.

Transaction Service Name: searchAuthRoles

Request DO: searchAuthAccessControlDO

Response DO: authUserRegistryDOList (empty list if no records found)

Mandatory Input: rolename or username

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

Wildcard Support: Yes. % can be used for wildcard searches

Use Case –

1. To check if given userName and roleName combination is valid
2. To check if userName is valid
3. To get all the users associated with a valid role

Special Notes – N/A

3. searchAuthAccessControl

Description: This service is used to get the transaction service name based on userName or roleName

The response is sent back based on below rule

1. If username and userRole are present in the request then authorization if all of the below conditions are satisfied
   - ✓ The given user must be present in AUTH_USER_REGISTRY table.
   - ✓ The given role must be present in AUTH_ROLES_REGISTRY table.
   - ✓ The user must be associated with the given role in AUTH_USER_ROLE_ASSOC table
   - ✓ The response will be sent back having all the rows if txnserviceName have entry in Access control table for given role i.e. AUTH_USERROLE_ACCESSCONTROL have entry for given role and transaction service name.
2. If userRole is present and username is null in the request then role is authorized to perform the given transaction if all the below conditions are satisfied
   - ✓ The given role must be present in AUTH_ROLES_REGISTRY table.
   - ✓ The response will be sent back having all the rows if txnserviceName have entry in Access control table for given role i.e. AUTH_USERROLE_ACCESSCONTROL have entry for given role and transaction service name.
3. If username is present and userRole is null in the request then user is authorized to perform the given transaction if all the below conditions are satisfied
   - ✓ The given user must be present in AUTH_USER_REGISTRY table.
   - ✓ The response will be sent back having all the rows if txnserviceName have entry in Access control table for given user i.e. AUTH_USERROLE_ACCESSCONTROL have entry for given user and transaction service name.

Transaction Service Name: searchAuthAccessControl

Request DO: searchAuthAccessControlDO

Response DO: authUserRegistryDOList (empty list if no records found)

Mandatory Input: txnserviceName and either rolename or userName

Pagination Supported: Yes

Filter: ACTIVE / INACTIVE / ALL

Wildcard Support: NO

Use Case – To check if given user or role is authorized to invoke a given transaction.

Special Notes – This is the primary service used by MDM Hub to perform authorication of the request based on userName/roleName and txnserviceName

4. createAccount

Description: create an account with its address and phone associations

Transaction Service Name: createAccount

Request DO: accountDO

Response DO: accountDO

Mandatory Input: accountDO

Use Case – create a standalone account along with its address and phone associations.

Special Notes – accountDO.accountAddressAssocDOList and accountDO.acountPhoneAssocDOList will be processed if present.

5. createAccountAddressAssociation

Description: Associate existing account with existing or new address. If address Id in the request (txnPayload.accountAddressAssocDO.addressIdpk) is null then new address will be created in ADDRESS table and will be associated with account. If address Id (txnPayload.accountAddressAssocDO.addressIdpk) is present in the request then account and address association will be created.

Transaction Service Name: createAccountAddressAssociation

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDO

Mandatory Input: accountAddressAssocDO, accountAddressAssocDO .accountIdpk

Use Case –

1. Associate existing account with existing address
2. Associate existing account with new address

Special Notes – N/A

6. createAccountPhoneAssociation

Description: associate a phone number with existing account

Transaction Service Name: createAccountPhoneAssociation

Request DO: accountPhoneAssocDO

Response DO: accountPhoneAssocDO

Mandatory Input: accountPhoneAssocDO.accountIdPk

Use Case – associate a phone number with existing account

Special Notes – The standardized phone number phoneStandardizedDO can also be through this service. If existing idPk of phoneStandardizedDO is provided in the accountPhoneAssocDO. phoneStandardizedIdpk then the same will be used.

7. createEntityGroup

Description: create an entry in ENTITY_GROUP and ENTITY_GROUP_ASSOC table

Transaction Service Name: createEntityGroup

Request DO: entityGroupDO

Response DO: entityGroupDO

Mandatory Input: As per createEntityGroupBase and createEntityGroupAssocbase service

Use Case – create grouping of entities

Special Notes – N/A

8. createLeAccountAssociation

Description: This transaction associates an account to existing legal entity. If Account is already present then a new association is added between LE and account. It's also possible to create a new account through this service.

Transaction Service Name: createLeAccountAssociation

Request DO: leAccountAssocDO

Response DO: leAccountAssocDO

Mandatory Input: leAccountAssocDO.legalEntityIdPk, txnPayload.leAccountAssocDO.accountIdpk or txnPayload.leAccountAssocDO.accountDO

Use Case –

1. create legal entity to account association for existing LE and existing account
2. create a new account and associate it with legal entity

Special Notes – N/A

9. createLeAddressAssociation

Description: create an association between le and address. Create association based on AddressDO or AddressIdPk of existing address

Transaction Service Name: createLeAddressAssociation

Request DO: leAddressAssocDO

Response DO: leAddressAssocDO

Mandatory Input: leAddressAssocDO.legalEntityIdPk, leAddressAssocDO.addressIdpk or leAddressAssocDO .addressDO

Use Case –

1. create legal entity to address association for existing LE and existing address
2. create a new address and associate it with legal entity

Special Notes – N/A

10. createLeCorporation

Description: create a legal entity of type Corporation along with corporation names.

Transaction Service Name: createLeCorporation

Request DO: leCorporationDO

Response DO: leCorporationDO

Mandatory Input: leCorporationDO.legalentityIdpk

Use Case – N/A

Special Notes – This method does not create a record in LEGALENTITY table so this service should be used only if LEGALENTITY is already created through different process and there is a need to create records in LE_CORPORATION and CORPORATIONNAMES. Its recommended to use createLegalEntity service to create a legal entity.

11. createLePerson

Description: create a legal entity of type Person along with person names.

Transaction Service Name: createLePerson

Request DO: lePersonDO

Response DO: lePersonDO

Mandatory Input: lePersonDO.legalentityIdpk

Use Case – N/A

Special Notes – This method does not create a record in LEGALENTITY table so this service should be used only if LEGALENTITY is already created through different process and there is a need to create records in LE_PERSON and PERSONNAMES. It's recommended to use createLegalEntity service to create a legal entity.

12. createLePhoneAssociation

Description: create an association between legal entity and phone. It's also possible to create a standardized phone number record through this service.

Transaction Service Name: createLePhoneAssociation

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDO

Mandatory Input: lePhoneAssocDO.legalEntityIdPk

Use Case –

1. Associate legal entity with phone number
2. Associate a legal entity with phone number and also store the phone number in standardized format.

Special Notes – N/A

13. createLePropertyAssociation

Description: create an association between existing legal entity and existing property. It's also possible to create a new property record and associate it with legal entity.

Transaction Service Name: createLePropertyAssociation

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input: lePropertyAssocDO.legalEntityIdPk, lePropertyAssocDO.propertyIdpk or lePropertyAssocDO. propertyDO

Use Case –

1. create legal entity to property association for existing LE and existing property
2. create a new property and associate it with legal entity

Special Notes – N/A

14. createLeVehicleAssociation

Description: create an association between existing legal entity and existing Vehicle. It's also possible to create a new vehicle record and associate it with legal entity.

Transaction Service Name: createLeVehicleAssociation

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input: leVehicleAssocDO.legalEntityIdPk, leVehicleAssocDO.vehicleDO or leVehicleAssocDO.vehicleIdPk

Use Case –

1. create legal entity to property association for existing LE and existing property
2. create a new property and associate it with legal entity

Special Notes – N/A

15. createLegalEntity

Description: create a legal entity along with person or corporation, system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info

Transaction Service Name: createLegalEntity

Request DO: legalentityDO

Response DO: legalentityDO

Mandatory Input: entityObjectTypeRefkey, lePersonDO or leCorporationDO

Use Case –

1. create new legal entity along with all its objects like person or corporation, system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info

Special Notes – N/A

16. findAllAccountAddressAssocByAccountId

Description: This service returns all the address associated with an account.

Transaction Service Name: findAllAccountAddressAssocByAccountId

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDOList

Mandatory Input: accountIdPk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the addresses associated with an account based on account idPk

Special Notes – This service does not return any account details.

17. findAllAccountPhoneAssocByAccountId

Description: This service returns all the phone numbers associated with an account.

Transaction Service Name: findAllAccountPhoneAssocByAccountId

Request DO: accountPhoneAssocDO

Response DO: accountPhoneAssocDOList

Mandatory Input: accountIdPk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the phone numbers associated with an account based on account idPk

Special Notes – This service does not return any account details.

18. findAllLeAccountAssocByLegalEntityId

Description: This service returns all the accounts associated with legal entity

Transaction Service Name: findAllLeAccountAssocByLegalEntityId

Request DO: LeAccountAssocDO

Response DO: LeAccountAssocDOList

Mandatory Input: leAccountAssocDO. legalEntityIdPk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the accounts associated with a legal entity.

Special Notes – Supports account level inquiry levels. Set the leAccountAssocDO.accountInquiryLevel to retrieve as per inquiry level else the default inquiry level as defined in configuration properties would be used. Refer 'Default Inquiry levels for Legal entity and Account – Default configuration' section to know more about inquiry levels available for account level. This service does not return any Legal entity details.

19. findAllLeAddressAssocByLegalEntityId

Description: This service returns all the addresses associated with a legal entity.

Transaction Service Name: findAllLeAddressAssocByLegalEntityId

Request DO: leAddressAssocDO

Response DO: leAddressAssocDOList

Mandatory Input: leAddressAssocDO.legalEntityIdPk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the addresses associated with a legal entity.

Special Notes – This service does not return legal entity details.

20. findAllLePhoneAssocByLegalEntityId

Description: This service returns all the phone numbers associated with a legal entity.

Transaction Service Name: findAllLePhoneAssocByLegalEntityId

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDOList

Mandatory Input: lePhoneAssocDO.legalEntityIdPk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the phone numbers associated with a legal entity.

Special Notes – This service does not return legal entity details.

21. findAllLePropertyAssocByLegalEntityId

Description: This service returns all the property details associated with a legal entity.

Transaction Service Name: findAllLePropertyAssocByLegalEntityId

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input: lePropertyAssocDO. legalentityIdpk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the property details associated with a legal entity.

Special Notes – This service does not return legal entity details.

22. findAllLeVehicleAssocByLegalEntityId

Description: This service returns all the vehicles associated with a legal entity. This service retrieves the legal entity form the database based on input legalentityIdpk and invokes matching rule as defined in the configuration properties.

Transaction Service Name: findAllLeVehicleAssocByLegalEntityId

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input: leVehicleAssocDO. legalentityIdpk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – get all the vehicles associated with a legal entity.

Special Notes – This service does not return legal entity details.

23. performLeMatch

Description: This service is used to perform the match of given legal entity with the LEs in database.

Transaction Service Name: performLeMatch

Request DO: performLeMatchRequestDO

Response DO: performLeMatchRequestDO

Mandatory Input:

- performLeMatchRequestDO.legalentityDO.idpk
- performLeMatchRequestDO.matchEngineType

Use Case – perform the match of given legal entity with the LE present in database.

Special Notes – This service is used by MDM Hub for performing matching. The users who want to customize the logic they must create a new service. The code of this service must not be changed else the default matching may not work properly.

24. retrieveAccountAddressByAccountAddressAssocId

Description: retrieves account to address association based on accountAddressAssocDO.idPk

Transaction Service Name: retrieveAccountAddressByAccountAddressAssocId

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDO

Mandatory Input: accountAddressAssocDO.idPk

Pagination Supported: N/A

Filter: N/A

Wildcard Support: N/A

Use Case – retrieve the address of the account based on accountAddressAssocDO.idPk

Special Notes – N/A

25. retrieveAccountByAccountId

Description: This service retrieves account details along with Addresses, phones and le-account associations.

Transaction Service Name: retrieveAccountByAccountId

Request DO: accountDO

Response DO: accountDO

Mandatory Input: accountDO.IdPk

Pagination Supported: N/A

Filter: N/A

Wildcard Support: N/A

Use Case –

Special Notes – This transaction supports inquiry levels, set the accountDO.inquiryLevel to a valid value as mentioned in 'Default Inquiry levels for Legal entity and Account – Default configuration' section.

26. retrieveAccountPhoneByAccountPhoneAssocId

Description: retrieves an account to phone association based on accountPhoneAssocDO.idPk

Transaction Service Name: retrieveAccountPhoneByAccountPhoneAssocId

Request DO: accountPhoneAssocDO

Response DO: accountPhoneAssocDO

Mandatory Input: accountPhoneAssocDO.idPk

Use Case – retrieve the phone numbers of the account based on accountAddressAssocDO.idPk

Special Notes – N/A

27. retrieveEntityGroupByEntityGroupId

Description: This service retrieves the entire grouping information containing the associated entities. The data from ENTITY_GROUP along with all associated records from ENTITY_GROUP_ASSOC table are returned in this service.

Transaction Service Name: retrieveEntityGroupByEntityGroupId

Request DO: entityGroupDO

Response DO: entityGroupDO

Mandatory Input: entityGroupDO.idPk

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: N/A

Use Case – retrieve the entire grouping information containing the associated entities.

Special Notes – N/A

28. retrieveLeAccountByLeAccountAssocId

Description: This service retrieves account details based on LeAccountAssocId

Transaction Service Name: retrieveLeAccountByLeAccountAssocId

Request DO: leAccountAssocDO

Response DO: leAccountAssocDO

Mandatory Input: leAccountAssocDO.idPk

Pagination Supported: N/A

Filter: N/A

Wildcard Support: N/A

Use Case – retrieve account details based on LeAccountAssocId

Special Notes – This service supports inquiry level for retrieving account details. Provide valid inquiry level in leAccountAssocDO .accountInquiryLevel attribute as mentioned in 'Default Inquiry levels for Legal entity and Account – Default configuration' section.

29. retrieveLeAddressByLeAddressAssocId

Description: retrieves legal entity to address association based on leAddressAssocDO.idPk

Transaction Service Name: retrieveLeAddressByLeAddressAssocId

Request DO: LeAddressAssocDO

Response DO: LeAddressAssocDO

Mandatory Input: leAddressAssocDO.idPk

Use Case – retrieves legal entity to address association based on leAddressAssocDO.idPk

Special Notes – N/A

30. retrieveLeCorporationByLegalEntityId

Description: retrieve legal entity of corporation type along with its names

Transaction Service Name: retrieveLeCorporationByLegalEntityId

Request DO: leCorporationDO

Response DO: leCorporationDO

Mandatory Input: leCorporationDO.legalentityIdpk

Pagination Supported: YES (pagination supported to retrieve names)

Filter: N/A

Wildcard Support: N/A

Use Case – retrieve legal entity of corporation type along with its names

Special Notes – This service does not retrieve a record from LEGALENTITY table so this service should be used only if LEGALENTITY data is needed. It's recommended to use retrieveLegalEntityByLegalEntityId service to retrieve a legal entity.

31. retrieveLePersonByLegalEntityId

Description: retrieve legal entity of person type along with its names

Transaction Service Name: retrieveLePersonByLegalEntityId

Request DO: lePersonDO

Response DO: lePersonDO

Mandatory Input: lePersonDO. legalentityIdpk

Pagination Supported: YES (pagination supported to retrieve names)

Filter: N/A

Wildcard Support: N/A

Use Case – retrieve legal entity of person type along with its names

Special Notes – This service does not retrieve a record from LEGALENTITY table so this service should be used only if LEGALENTITY data is needed. It's recommended to use retrieveLegalEntityByLegalEntityId service to retrieve a legal entity.

32. retrieveLePhoneByLePhoneAssocId

Description: Retrieves an account to phone association based on lePhoneAssocDO.idPk

Transaction Service Name: retrieveLePhoneByLePhoneAssocId

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDO

Mandatory Input: lePhoneAssocDO.idPk

Pagination Supported: N/A

Filter: N/A

Wildcard Support: N/A

Use Case – retrieves an account to phone association based on lePhoneAssocDO.idPk

Special Notes – N/A

33. retrieveLePropertyByLePropertyAssocId

Description: Retrieve the property details associated with legal entity based on LePropertyAssocId

Transaction Service Name: retrieveLePropertyByLePropertyAssocId

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input: lePropertyAssocDO.idPk

Pagination Supported: N/A

Filter: N/A

Wildcard Support: N/A

Use Case – Retrieve le property details based on LePropertyAssocId

Special Notes – N/A

34. retrieveLeVehicleLeVehicleAssocId

Description: Retrieve the vehicle details associated with legal entity based on LeVehicleAssocId

Transaction Service Name: retrieveLeVehicleLeVehicleAssocId

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input: leVehicleAssocDO.idPk

Pagination Supported: N/A

Filter: N/A

Wildcard Support: N/A

Use Case – Retrieve the vehicle details associated with legal entity based on LeVehicleAssocId

Special Notes – N/A

35. retrieveLegalEntityByLegalEntityId

Description: Retrieve legal entity and all its associated person or corporation, system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info

Transaction Service Name: retrieveLegalEntityByLegalEntityId

Request DO: legalentityDO

Response DO: legalentityDO

Mandatory Input: legalentityDO.idpk

Pagination Supported: YES

Filter: N/A

Wildcard Support: N/A

Use Case – retrieve legal entity and all its associated person or corporation, system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info

Special Notes – LE and account inquiry levels are supported for this service. Provide valid value in legalentityDO .inquiryLevel for LE inquiry and legalentityDO.accountInquiryLevel for account inquiry. Account inquiry level will only be used if LE inquiry level retrieves an account else this attribute will be ignored. If LE inquiry level is not provided then le will be retrieved based on default inquiry level as set in configuration properties.

36. searchAccountByAccountAttributes

Description: search the account in database based on account attributes

Transaction Service Name: searchAccountByAccountAttributes

Request DO: searchAccountRequestDO

Response DO: accountDOList

Mandatory Input: at least one attribute from the below

- searchAccountRequestDO.accountidPk
- searchAccountRequestDO.sourceSystemRefkey
- searchAccountRequestDO.sourceAccountId
- searchAccountRequestDO.accountName
- searchAccountRequestDO.accountName2
- searchAccountRequestDO.accountDescription
- searchAccountRequestDO.accountSourceStatusRefkey
- searchAccountRequestDO.accountMdmStatusRefkey

Pagination Supported: YES.

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case – search and account based on account attributes and retrieve the detailed information of the account along with its address, phones in search result.

Special Notes –

- Account will be retrieved as per searchAccountRequestDO.inquiryLevel. If inquiry level is not provided then only the search result with details for AccountDO (ACCOUNT table) will be returned.

- if searchAccountRequestDO.inquiryFilter is null then only ACTIVE accounts will be searched

37. searchAccountByLEAttributes

Description: search the account in database based on associated legal entity attributes. The database will be searched for legal entities matching the search criteria and the accounts associated with these Legal entities will be returned in response.

Transaction Service Name: searchAccountByLEAttributes

Request DO: searchAccountRequestDO

Response DO: searchAccountRequestDO

Mandatory Input: One of the attributes SearchAccountRequestDO.displayName, nameOne, lastName, corporationName is required if systemKeysRegistryReferenceId or identificationNumber or phoneNumber is not provided.

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case – get the accounts associated with searched legal entities

Special Notes –

- Account will be retrieved as per searchAccountRequestDO.inquiryLevel. If inquiry level is not provided then only the search result with details for AccountDO (ACCOUNT table) will be returned.
- if searchAccountRequestDO.inquiryFilter is null then only ACTIVE legal entities and accounts will be searched

38. searchInactiveLeRegistry

Description: search the INACTIVE_LE_REGISTRY table based on its attributes

Transaction Service Name: searchInactiveLeRegistry

Request DO: inactiveLeRegistryDO

Response DO: inactiveLeRegistryDO

Mandatory Input: inactiveLeRegistryDO.legalentityIdpk or inactivationReasonRefkey

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case – search the INACTIVE_LE_REGISTRY table based on its attributes

Special Notes – N/A

39. searchLegalEntityByAccountAttributes

Description: This service search legal entity based on account attributes. All the legal entities associated with the accounts searched will be returned in the response

Transaction Service Name: searchLegalEntityByAccountAttributes

Request DO: searchLegalEntityRequestDO

Response DO: legalentityDOList

Mandatory Input: at least one of the following

- searchLegalEntityRequestDO.displayName
- searchLegalEntityRequestDO.accountidPk
- searchLegalEntityRequestDO.sourceSystemRefkey
- searchLegalEntityRequestDO.sourceAccountId
- searchLegalEntityRequestDO.accountName
- searchLegalEntityRequestDO.accountName2
- searchLegalEntityRequestDO.accountDescription
- searchLegalEntityRequestDO.accountSourceStatusRefkey
- searchLegalEntityRequestDO.accountMdmStatusRefkey

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case – This service search legal entity based on account attributes

Special Notes – The legal entity can be retrieved based on inquiry level. Provide valid value in searchLegalEntityRequestDO.inquiryLevel attribute to retrieve legal entity based on inquiry level. if inquiry level is not provided then the search response will result only the LegalentityDOs without any child objects of respective LegalentityDO.

40. searchLegalEntityByLEAttributes

Description: This service search legal entity based on LE attributes.

Transaction Service Name: searchLegalEntityByLEAttributes

Request DO: searchLegalEntityRequestDO

Response DO: legalentityDOList

Mandatory Input: One of the attributes SearchLegalEntityRequestDO.displayName, nameOne, lastName, corporationName is required if systemKeysRegistryReferenceId or identificationNumber or phoneNumber is not provided

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case – search legal entity based on LE attributes.

Special Notes – The legal entity can be retrieved based on inquiry level. Provide valid value in searchLegalEntityRequestDO.inquiryLevel attribute to retrieve legal entity based on inquiry level. if inquiry level is not provided then the search response will result only the LegalentityDOs without any child objects of respective LegalentityDO.

41. searchMatchCandidateLE

Description: This service search the candidates for given LE. If the

Transaction Service Name: searchMatchCandidateLE

Request DO: searchMatchCandidateLERequestDO

Response DO: legalentityDOList

Mandatory Input:

- searchMatchCandidateLERequestDO.IdentificationTypeRefkeyList
- searchMatchCandidateLERequestDO.identificationNumberList
- One of the below
    - searchMatchCandidateLERequestDO.nameOneList
    - searchMatchCandidateLERequestDO.lastNameList
      OR
    - searchMatchCandidateLERequestDO.corporationNameList

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case –

- search the match candidates for person or corporation type of LE based on exact match
- search the match candidates for person or corporation type of LE based on phonetic match

Special Notes – if phonetic search then set the isPhoneticSearch = true.

42. searchMatchCandidateLeRegistryBase

Description: search the match candidates from the MATCH_CANDIDATE_LE_REGISTRY table. Its considered that the matching is already performed and candidates list is created in the MATCH_CANDIDATE_LE_REGISTRY table.

Transaction Service Name: searchMatchCandidateLeRegistryBase

Request DO: matchCandidateLeRegistryDO

Response DO: matchCandidateLeRegistryDOList

Mandatory Input: one of the attribute matchCandidateLeRegistryDO.legalentityidpk or candidateLegalentityidpk or matchPattern or matchProposedActionRefkey or matchActionstatusRefkey is needed.

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case:

- search the match candidates of particular LE
- search the match candidates of particular match pattern
- search the match candidates of specific proposed action or action status

Special Notes: N/A

43. searchMatchMergedLeAssocBase

Description: This service search the relation of the legal entities merged.

Transaction Service Name: searchMatchMergedLeAssocBase

Request DO: matchMergedLeAssocDO

Response DO: matchMergedLeAssocDOList

Mandatory Input: MatchMergedLeAssocDO.survivorLegalentityIdpk or mergedLegalentityIdpk or mergeReasonRefkey

Pagination Supported: YES

Filter: ACTIVE/INACTIVE/ALL

Wildcard Support: YES

Use Case:

- search the legal entities merged based on survivor legal entity
- search the legal entities merged based on non-survivor (merged) legal entity
- search the legal entities based on merged reason

Special Notes: N/A

44. updateAccount

Description: update an account with its address and phone associations.

Transaction Service Name: updateAccount

Request DO: accountDO

Response DO: accountDO

Mandatory Input: accountDO.idPk, version

Use Case:

- Update existing account attributes and update account to address and/or account to phone associations.
- Update existing account and create new address/phone association.

Special Notes:

- The idPk of each child object alongwith version needs to be present in the request if the same needs to be updated. E.g. if the address of the account needs to be updated then idPk of the accountDO.accountAddressAssocDOList(0).idPk and accountDO.accountAddressAssocDOList(0).addressDO.idPk must be present with respective version attribute.
- If the idPk at the child object level is null then its assumed to be a new associated and a new record will be created in the database. So update transaction can be used to create new address or phone number association with existing account.

45. updateAccountAddressAssociation

Description: update the existing account – address association. If address Id in the request (txnPayload.accountAddressAssocDO.addressIdpk) is null then new address will be created in ADDRESS table and will be associated with account. If address Id (txnPayload.accountAddressAssocDO.addressIdpk) is present in the request then account and address association updated.

Transaction Service Name: updateAccountAddressAssociation

Request DO: accountAddressAssocDO

Response DO: accountAddressAssocDO

Mandatory Input: accountAddressAssocDO.accountgalEntityIdPk

Use Case:

- update existing account – address association record and update the assoc attributes as well as address attributes
- update existing account – address association record and update the assoc attributes and create a new address record in database.

Special Notes – This service does not update any account attributes.

46. updateAccountPhoneAssociation

Description: update the phone number associated with an account

Transaction Service Name: updateAccountPhoneAssociation

Request DO: accountPhoneAssocDO

Response DO: accountPhoneAssocDO

Mandatory Input: accountPhoneAssocDO.accountIdPk

Use Case:

- update existing account – phone number association record and update the phoneStandardizedDO of the phone number.
- update existing account – address association record and update the assoc attributes and create a new phoneStandardizedDO record in database.

Special Notes: N/A

47. updateEntityGroup

Description: This service is used to update existing entity group as well as new entity grouping association.

Transaction Service Name: updateEntityGroup

Request DO: entityGroupDO

Response DO: entityGroupDO

Mandatory Input: entityGroupDO.idPk

Use Case –

- update existing entity group
- update existing entity group association
- add a new association to entity entity group

Special Notes – entityGroupDO contains entityGroupAssocDOList object which can be used to add associations. To update the existing association, idPk and version needs to be provided.

48. updateLeAccountAssociation

Description: Update existing LE Account association

Transaction Service Name: updateLeAccountAssociation

Request DO: leAccountAssocDO

Response DO: leAccountAssocDO

Mandatory Input:

- LeAccountAssocDO.idPk
- LeAccountAssocDO.LegalEntityIdPk
- leAccountAssocDO.AccountIdpk or AccountDO to create new account

Use Case:

- update existing Le account association
- update existing Le account association and update account attributes
- update existing Le account association to replace the existing account by creating new account

Special Notes: This service internally uses update Account so all the account details including address and phone association can be updated through the same.

49. updateLeAddressAssociation

Description: update existing address association along with address details

Transaction Service Name: updateLeAddressAssociation

Request DO: leAddressAssocDO

Response DO: leAddressAssocDO

Mandatory Input:

- LeAddressAssocDO.idPk
- LeAddressAssocDO.LegalEntityIdPk

- leAddressAssocDO.addressIdpk or AddressDO to create new address

Use Case:

- update existing Le address association
- update existing Le address association and update address attributes
- update existing Le address association to replace the existing address by creating new address

Special Notes: N/A

50. updateLeCorporation

Description: update a legal entity of type Corporation along with corporation names.

Transaction Service Name: updateLeCorporation

Request DO: leCorporationDO

Response DO: leCorporationDO

Mandatory Input: leCorporationDO.legalentityIdpk

Use Case –

- update a corporation and update existing name
- update a corporation and add new name

Special Notes – This method does not update a record in LEGALENTITY table.

51. updateLePerson

Description: update a legal entity of type Person along with person names.

Transaction Service Name: updateLePerson

Request DO: lePersonDO

Response DO: lePersonDO

Mandatory Input: lePersonDO.legalentityIdpk

Use Case –

- update a person and update existing name
- update a person and add new name

Special Notes – This method does not create a record in LEGALENTITY.

52. updateLePhoneAssociation

Description: update an existing association between legal entity and phone. It's also possible to create or update a standardized phone number record through this service.

Transaction Service Name: updateLePhoneAssociation

Request DO: lePhoneAssocDO

Response DO: lePhoneAssocDO

Mandatory Input:

- LePhoneAssocDO.idpk
- LePhoneAssocDO.version
- LePhoneAssocDO.LegalEntityIdPk

Use Case –

1. Update existing legal entity to phone association
2. Update existing legal entity to phone association and add standardized phone number.

Special Notes – N/A

53. updateLePropertyAssociation

Description: update an existing association between existing legal entity and existing property. It's also possible to create a new property record and associate it with legal entity.

Transaction Service Name: updateLePropertyAssociation

Request DO: lePropertyAssocDO

Response DO: lePropertyAssocDO

Mandatory Input:

- lePropertyAssocDO.legalEntityIdPk
- lePropertyAssocDO.propertyIdpk or lePropertyAssocDO. propertyDO
- lePropertyAssocDO.version

Use Case –

1. update existing legal entity to property association
2. update existing legal entity to property association by creating new property record.

Special Notes – N/A

54. updateLeVehicleAssociation

Description: update an existing association between existing legal entity and existing Vehicle. It's also possible to create a new vehicle record and associate it with legal entity.

Transaction Service Name: updateLeVehicleAssociation

Request DO: leVehicleAssocDO

Response DO: leVehicleAssocDO

Mandatory Input:

- leVehicleAssocDO.legalEntityIdPk
- leVehicleAssocDO.vehicleDO or leVehicleAssocDO.vehicleIdPk
- leVehicleAssocDO.version

Use Case –

1. update existing legal entity to property association and update existing property record as well
2. update existing legal entity to property association and create new property record

Special Notes – N/A

55. updateLegalEntity

Description: update a legal entity along with person or corporation, system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info

Transaction Service Name: updateLegalEntity

Request DO: legalentityDO

Response DO: legalentityDO

Mandatory Input:

- legalentityDO.IdPk
- legalentityDO.version
- entityObjectTypeRefkey
- lePersonDO or leCorporationDO

Use Case –

1. update existing legal entity along with all its objects like person or corporation, system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info
2. update existing legal entity and create all its objects like system registry keys, preferences, identifiers, address, phone, account, property, vehicle and miscellaneous info

Special Notes – N/A

Reference data Table Management services

All the reference data services follow same structure as follows. The transactionServiceName and txnPayload.xxxxxxxDO will change as per the reference list name.

Create Request - sample JSON message

```
{
        "txnHeader": {
         "requesterLanguage": "1",
    "requesterLocale": null,
    "userName": "admin",
    "userRole": "admin",
    "accessToken": null,
    "txnCorrelationId": null,
    "txnMessageId": "1234567890123",
    "requestOriginSource": null,
    "requesterSystem": null,
       "transactionServiceName": "create<referenceTableName>Base"
       },
       "txnPayload": {
                        "<referenceTableName>DO":       {
                       "idPk": null,
                       "version": null,
                       "createdTs": null,
                       "deletedTs": null,
                       "updatedTs": null,
                       "updatedByUser": null,
                       "updatedByTxnId": null,
                       "configLanguageCodeKey": null,
                       "key": null,
                       "value": null,
                       "description": null,
    }}

}
```

update request - sample JSON message

```
{
        "txnHeader": {
                "requesterLanguage": "1",
    "requesterLocale": null,
    "userName": "admin",
    "userRole": "admin",
    "accessToken": null,
    "txnCorrelationId": null,
    "txnMessageId": "1234567890123",
    "requestOriginSource": null,
    "requesterSystem": null,
      "transactionServiceName": "update<referenceTableName>Base"
      },
      "txnPayload": {
                 "<referenceTableName>DO":      {
                "idPk": null,
                "version": null,
                "createdTs": null,
                "deletedTs": null,
                "updatedTs": null,
                "updatedByUser": null,
                "updatedByTxnId": null,
                "configLanguageCodeKey": null,
                "key": null,
                "value": null,
                "description": null,
    }

      }

}
```

Retrieve Request - sample JSON message

```
{
      "txnHeader": {
           "requesterLanguage": "1",
    "requesterLocale": null,
    "userName": "admin",
    "userRole": "admin",
    "accessToken": null,
    "txnCorrelationId": null,
    "txnMessageId": "1234567890123",
    "requestOriginSource": null,
    "requesterSystem": null,
       "transactionServiceName": "retrieve<referenceTableName>Base"
      },
      "txnPayload": {
                 "<referenceTableName>DO":       {
     "idPk": "1"

    }

      }

}
```

findByBusinessKey request - sample JSON message.

```
{
        "txnHeader": {
                "requesterLanguage": "1",
     "requesterLocale": null,
     "userName": "admin",
     "userRole": "admin",
     "accessToken": null,
     "txnCorrelationId": null,
     "txnMessageId": "1234567890123",
     "requestOriginSource": null,
     "requesterSystem": null,
                "transactionServiceName": "find<referenceTableName>ByBusinessKeyBase"
        },
        "txnPayload": {
                         "<referenceTableName>DO":       {
                "configLanguageCodeKey": "1",
       "key": "ACTIVE"
     }

        }

}
```

findAllRecorfdsByLanguageCode request - sample JSON message.

The response of this message is the list object of request DO. E.g. if the request do is refCountryIsoDO then the response object (in the txnPayload) would be refCountryIsoDOList.

```
{
        "txnHeader": {
                "requesterLanguage": "1",
                "userName": "Rakesh9",
                "userRole": "admin9",
                "txnMessageId": "12312311115999",
                "transactionServiceName":
"findAll<referenceTableName>ByLanguageCodeBase"
        },
        "txnPayload": {
                "paginationIndexOfCurrentSlice": 0,
                "paginationPageSize": 25,
                "<referenceTableName>DO":        {
                "configLanguageCodeKey": "1"
     }

        }
}
```

All the below mentioned services would have same structure as mentioned above so those are not covered in detail independently.

List of Reference data Management services following common request structure

1. *createRefAccountMdmStatusBase*
2. *createRefAccountSourceStatusBase*
3. *createRefAddressSubtypeBase*
4. *createRefAddressTypeBase*
5. *createRefAgreementTypeBase*
6. *createRefAssocTypeBase*
7. *createRefBillingModeTypeBase*
8. *createRefBranchCodeBase*
9. *createRefClassificationCodeBase*
10. *createRefCorporationNameTypeBase*
11. *createRefCorporationTypeBase*
12. *createRefCountryIsoBase*
13. *createRefCurrencyBase*
14. *createRefDeactivationReasonBase*
15. *createRefEntityObjectTypeBase*
16. *createRefGenderBase*
17. *createRefGroupSubtypeBase*
18. *createRefGroupTypeBase*
19. *createRefHighestEduQualBase*
20. *createRefIdentificationTypeBase*
21. *createRefImportanceTypeBase*
22. *createRefIndustryCodeBase*
23. *createRefLanguageCodeBase*
24. *createRefLeRatingBase*
25. *createRefLeRelationshipTypeBase*
26. *createRefLeRoletypeBase*

27. *createRefLobtypeBase*

28. *createRefPersonTypeBase*

29. *createRefPersonnameTypeBase*

30. *createRefPhoneSubtypeBase*

31. *createRefPhoneTypeBase*

32. *createRefPreferenceTypeBase*

33. *createRefPrefixNameBase*

34. *createRefPropertyLeReltypeBase*

35. *createRefRelationshipStatusBase*

36. *createRefSourceSystemBase*

37. *createRefStateProvinceBase*

38. *createRefStatusInSourceBase*

39. *createRefStatusTypeBase*

40. *createRefSuffixNameBase*

41. *createRefTerminationReasonBase*

42. *findAllRefAccountMdmStatusByLanguageCodeBase*

43. *findAllRefAccountSourceStatusByLanguageCodeBase*

44. *findAllRefAddressSubtypeByLanguageCodeBase*

45. *findAllRefAddressTypeByLanguageCodeBase*

46. *findAllRefAgreementTypeByLanguageCodeBase*

47. *findAllRefAssocTypeByLanguageCodeBase*

48. *findAllRefBillingModeTypeByLanguageCodeBase*

49. *findAllRefBranchCodeByLanguageCodeBase*

50. *findAllRefClassificationCodeByLanguageCodeBase*

51. *findAllRefCorporationNameTypeByLanguageCodeBase*

52. *findAllRefCorporationTypeByLanguageCodeBase*

53. *findAllRefCountryIsoByLanguageCodeBase*

54. ***findAllRefCurrencyByLanguageCodeBase***

55. ***findAllRefDeactivationReasonByLanguageCodeBase***

56. ***findAllRefEntityObjectTypeByLanguageCodeBase***

57. ***findAllRefGenderByLanguageCodeBase***

58. ***findAllRefGroupSubtypeByLanguageCodeBase***

59. ***findAllRefGroupTypeByLanguageCodeBase***

60. ***findAllRefHighestEduQualByLanguageCodeBase***

61. ***findAllRefIdentificationTypeByLanguageCodeBase***

62. ***findAllRefImportanceTypeByLanguageCodeBase***

63. ***findAllRefIndustryCodeByLanguageCodeBase***

64. ***findAllRefLanguageCodeByLanguageCodeBase***

65. ***findAllRefLeRatingByLanguageCodeBase***

66. ***findAllRefLeRelationshipTypeByLanguageCodeBase***

67. ***findAllRefLeRoletypeByLanguageCodeBase***

68. ***findAllRefLobtypeByLanguageCodeBase***

69. ***findAllRefPersonTypeByLanguageCodeBase***

70. ***findAllRefPersonnameTypeByLanguageCodeBase***

71. ***findAllRefPhoneSubtypeByLanguageCodeBase***

72. ***findAllRefPhoneTypeByLanguageCodeBase***

73. ***findAllRefPreferenceTypeByLanguageCodeBase***

74. ***findAllRefPrefixNameByLanguageCodeBase***

75. ***findAllRefPropertyLeReltypeByLanguageCodeBase***

76. ***findAllRefRelationshipStatusByLanguageCodeBase***

77. ***findAllRefSourceSystemByLanguageCodeBase***

78. ***findAllRefStateProvinceByLanguageCodeBase***

79. ***findAllRefStatusInSourceByLanguageCodeBase***

80. ***findAllRefStatusTypeByLanguageCodeBase***

81. *findAllRefSuffixNameByLanguageCodeBase*
82. *findAllRefTerminationReasonByLanguageCodeBase*
83. *findRefAccountMdmStatusByBusinessKeyBase*
84. *findRefAccountSourceStatusByBusinessKeyBase*
85. *findRefAddressSubtypeByBusinessKeyBase*
86. *findRefAddressTypeByBusinessKeyBase*
87. *findRefAgreementTypeByBusinessKeyBase*
88. *findRefAssocTypeByBusinessKeyBase*
89. *findRefBillingModeTypeByBusinessKeyBase*
90. *findRefBranchCodeByBusinessKeyBase*
91. *findRefClassificationCodeByBusinessKeyBase*
92. *findRefCorporationNameTypeByBusinessKeyBase*
93. *findRefCorporationTypeByBusinessKeyBase*
94. *findRefCountryIsoByBusinessKeyBase*
95. *findRefCurrencyByBusinessKeyBase*
96. *findRefDeactivationReasonByBusinessKeyBase*
97. *findRefEntityObjectTypeByBusinessKeyBase*
98. *findRefGenderByBusinessKeyBase*
99. *findRefGroupSubtypeByBusinessKeyBase*
100. *findRefGroupTypeByBusinessKeyBase*
101. *findRefHighestEduQualByBusinessKeyBase*
102. *findRefIdentificationTypeByBusinessKeyBase*
103. *findRefImportanceTypeByBusinessKeyBase*
104. *findRefIndustryCodeByBusinessKeyBase*
105. *findRefLanguageCodeByBusinessKeyBase*
106. *findRefLeRatingByBusinessKeyBase*
107. *findRefLeRelationshipTypeByBusinessKeyBase*

108. *findRefLeRoletypeByBusinessKeyBase*

109. *findRefLobtypeByBusinessKeyBase*

110. *findRefPersonTypeByBusinessKeyBase*

111. *findRefPersonnameTypeByBusinessKeyBase*

112. *findRefPhoneSubtypeByBusinessKeyBase*

113. *findRefPhoneTypeByBusinessKeyBase*

114. *findRefPreferenceTypeByBusinessKeyBase*

115. *findRefPrefixNameByBusinessKeyBase*

116. *findRefPropertyLeReltypeByBusinessKeyBase*

117. *findRefRelationshipStatusByBusinessKeyBase*

118. *findRefSourceSystemByBusinessKeyBase*

119. *findRefStateProvinceByBusinessKeyBase*

120. *findRefStatusInSourceByBusinessKeyBase*

121. *findRefStatusTypeByBusinessKeyBase*

122. *findRefSuffixNameByBusinessKeyBase*

123. *findRefTerminationReasonByBusinessKeyBase*

124. *retrieveRefAccountMdmStatusBase*

125. *retrieveRefAccountSourceStatusBase*

126. *retrieveRefAddressSubtypeBase*

127. *retrieveRefAddressTypeBase*

128. *retrieveRefAgreementTypeBase*

129. *retrieveRefAssocTypeBase*

130. *retrieveRefBillingModeTypeBase*

131. *retrieveRefBranchCodeBase*

132. *retrieveRefClassificationCodeBase*

133. *retrieveRefCorporationNameTypeBase*

134. *retrieveRefCorporationTypeBase*

135. ***retrieveRefCountryIsoBase***

136. ***retrieveRefCurrencyBase***

137. ***retrieveRefDeactivationReasonBase***

138. ***retrieveRefEntityObjectTypeBase***

139. ***retrieveRefGenderBase***

140. ***retrieveRefGroupSubtypeBase***

141. ***retrieveRefGroupTypeBase***

142. ***retrieveRefHighestEduQualBase***

143. ***retrieveRefIdentificationTypeBase***

144. ***retrieveRefImportanceTypeBase***

145. ***retrieveRefIndustryCodeBase***

146. ***retrieveRefLanguageCodeBase***

147. ***retrieveRefLeRatingBase***

148. ***retrieveRefLeRelationshipTypeBase***

149. ***retrieveRefLeRoletypeBase***

150. ***retrieveRefLobtypeBase***

151. ***retrieveRefPersonTypeBase***

152. ***retrieveRefPersonnameTypeBase***

153. ***retrieveRefPhoneSubtypeBase***

154. ***retrieveRefPhoneTypeBase***

155. ***retrieveRefPreferenceTypeBase***

156. ***retrieveRefPrefixNameBase***

157. ***retrieveRefPropertyLeReltypeBase***

158. ***retrieveRefRelationshipStatusBase***

159. ***retrieveRefSourceSystemBase***

160. ***retrieveRefStateProvinceBase***

161. ***retrieveRefStatusInSourceBase***

162. ***retrieveRefStatusTypeBase***

163. ***retrieveRefSuffixNameBase***

164. ***retrieveRefTerminationReasonBase***

165. ***updateRefAccountMdmStatusBase***

166. ***updateRefAccountSourceStatusBase***

167. ***updateRefAddressSubtypeBase***

168. ***updateRefAddressTypeBase***

169. ***updateRefAgreementTypeBase***

170. ***updateRefAssocTypeBase***

171. ***updateRefBillingModeTypeBase***

172. ***updateRefBranchCodeBase***

173. ***updateRefClassificationCodeBase***

174. ***updateRefCorporationNameTypeBase***

175. ***updateRefCorporationTypeBase***

176. ***updateRefCountryIsoBase***

177. ***updateRefCurrencyBase***

178. ***updateRefDeactivationReasonBase***

179. ***updateRefEntityObjectTypeBase***

180. ***updateRefGenderBase***

181. ***updateRefGroupSubtypeBase***

182. ***updateRefGroupTypeBase***

183. ***updateRefHighestEduQualBase***

184. ***updateRefIdentificationTypeBase***

185. ***updateRefImportanceTypeBase***

186. ***updateRefIndustryCodeBase***

187. ***updateRefLanguageCodeBase***

188. ***updateRefLeRatingBase***

189. *updateRefLeRelationshipTypeBase*

190. *updateRefLeRoletypeBase*

191. *updateRefLobtypeBase*

192. *updateRefPersonTypeBase*

193. *updateRefPersonnameTypeBase*

194. *updateRefPhoneSubtypeBase*

195. *updateRefPhoneTypeBase*

196. *updateRefPreferenceTypeBase*

197. *updateRefPrefixNameBase*

198. *updateRefPropertyLeReltypeBase*

199. *updateRefRelationshipStatusBase*

200. *updateRefSourceSystemBase*

201. *updateRefStateProvinceBase*

202. *updateRefStatusInSourceBase*

203. *updateRefStatusTypeBase*

204. *updateRefSuffixNameBase*

205. *updateRefTerminationReasonBase*

206. *createRefBatchActionStatusBase*

207. *updateRefBatchActionStatusBase*

208. *retrieveRefBatchActionStatusBase*

209. *findRefBatchActionStatusByBusinessKeyBase*

210. *findAllRefBatchActionStatusByLanguageCodeBase*

211. *createRefBatchProposedActionBase*

212. *updateRefBatchProposedActionBase*

213. *retrieveRefBatchProposedActionBase*

214. *findRefBatchProposedActionByBusinessKeyBase*

215. *findAllRefBatchProposedActionByLanguageCodeBase*

216. ***createRefInactivationReasonBase***

217. ***createRefMatchActionstatusBase***

218. ***createRefMatchProposedActionBase***

219. ***createRefMatchResultBase***

220. ***findAllRefInactivationReasonByLanguageCodeBase***

221. ***findAllRefMatchActionstatusByLanguageCodeBase***

222. ***findAllRefMatchProposedActionByLanguageCodeBase***

223. ***findAllRefMatchResultByLanguageCodeBase***

224. ***findRefMatchResultByBusinessKeyBase***

225. ***findRefInactivationReasonByBusinessKeyBase***

226. ***findRefMatchActionstatusByBusinessKeyBase***

227. ***findRefMatchProposedActionByBusinessKeyBase***

228. ***findAllRefMergeReasonByLanguageCodeBase***

229. ***retrieveRefInactivationReasonBase***

230. ***retrieveRefMatchActionstatusBase***

231. ***retrieveRefMatchProposedActionBase***

232. ***retrieveRefMatchResultBase***

233. ***updateRefInactivationReasonBase***

234. ***updateRefMatchActionstatusBase***

235. ***updateRefMatchProposedActionBase***

236. ***updateRefMatchResultBase***

237. ***createRefMergeReasonBase***

238. ***findRefMergeReasonByBusinessKeyBase***

239. ***retrieveRefMergeReasonBase***

240. ***updateRefMergeReasonBase***

Inquiry levels for Legal entity and Account – Default configuration

Below is the list of inquiry levels configured by default in the CONFIG_INQUIRY_LEVELS table. Currently Account and Legal entity level inquiry levels are configured which can also be customized. Please refer development guide to understand more about defining new inquiry level.

Note – By default, the inquiry level is numeric in nature; Higher the number higher the inquiry level.

Inquiry Level	Applicable DO	Child DO	Comments if any
101	LegalentityDO	LeSystemKeysRegistryDO	
102	LegalentityDO	LeCorporationDO	
102	LegalentityDO	LePersonDO	
102	LegalentityDO	LeSystemKeysRegistryDO	
103	LegalentityDO	LeAddressAssocDO	
103	LegalentityDO	LeCorporationDO	
103	LegalentityDO	LePersonDO	
103	LegalentityDO	LePhoneAssocDO	
103	LegalentityDO	LePreferencesDO	
103	LegalentityDO	LeSystemKeysRegistryDO	
104	LegalentityDO	LeAddressAssocDO	
104	LegalentityDO	LeCorporationDO	
104	LegalentityDO	LeIdentifierKycRegistryDO	
104	LegalentityDO	LePersonDO	
104	LegalentityDO	LePhoneAssocDO	
104	LegalentityDO	LePreferencesDO	
104	LegalentityDO	LeSystemKeysRegistryDO	
104	LegalentityDO	MiscellaneousInfoDO	
105	LegalentityDO	LeAccountAssocDO	
105	LegalentityDO	LeAddressAssocDO	
105	LegalentityDO	LeCorporationDO	
105	LegalentityDO	LeIdentifierKycRegistryDO	
105	LegalentityDO	LePersonDO	
105	LegalentityDO	LePhoneAssocDO	
105	LegalentityDO	LePreferencesDO	
105	LegalentityDO	LeSystemKeysRegistryDO	
105	LegalentityDO	MiscellaneousInfoDO	
106	LegalentityDO	LeAccountAssocDO	
106	LegalentityDO	LeAddressAssocDO	
106	LegalentityDO	LeCorporationDO	
106	LegalentityDO	LeIdentifierKycRegistryDO	
106	LegalentityDO	LePersonDO	
106	LegalentityDO	LePhoneAssocDO	
106	LegalentityDO	LePreferencesDO	
106	LegalentityDO	LePropertyAssocDO	
106	LegalentityDO	LeSystemKeysRegistryDO	

106	LegalentityDO	LeToLeRelationshipDO	
106	LegalentityDO	LeVehicleAssocDO	
106	LegalentityDO	MiscellaneousInfoDO	
107	LegalentityDO	LeAccountAssocDO	
107	LegalentityDO	LeCorporationDO	
107	LegalentityDO	LePersonDO	
107	LegalentityDO	LeSystemKeysRegistryDO	
108	LegalentityDO	LeAddressAssocDO	This inquiry level is used by default Match and Merge framework so must not be customized
108	LegalentityDO	LeCorporationDO	This inquiry level is used by default Match and Merge framework so must not be customized
108	LegalentityDO	LeIdentifierKycRegistryDO	This inquiry level is used by default Match and Merge framework so must not be customized
108	LegalentityDO	LePersonDO	This inquiry level is used by default Match and Merge framework so must not be customized
108	LegalentityDO	LePhoneAssocDO	This inquiry level is used by default Match and Merge framework so must not be customized