Yugandhar Microservice Platform

Data Model Guide

Yugandhar Microservice Platform Release - V1.0.0
Date – 23/05/2018

Contents

About Yugandhar Project

The Yugandhar Project is the umbrella project focused on building open source cloud ready solutions. The current offerings include Yugandhar Microservice Platform (MSP) and Yugandhar Open Master Data Management (MDM) Hub.

About Yugandhar Microservice Platform

Yugandhar Microservice Platform (also referred as Yugandhar MSP) provides framework for rapid development of your Microservice. This is an architecturally proven Springboot based application having all the basic components needed for a Microservice application to work which just needs to be extended as per your requirements.

Yugandhar Microservice platform codes with code generation templates for rapid development. Just design your data model and generate the code using Yugandhar templates, your base table services will be ready. To create the composite services, you may have to write custom code which would take minimal efforts. Your new Microservice would be ready in just few hours.

About this document

This document covers the data model of Yugandhar Microservice Platform.

Understanding Data Table structure

The Yugandhar Microservice Platform does not provide any data entities but a framework to generate the code as per your own data model. The data entities must follow the below guidelines for optimistic lock and Yugandhar Code generation templates to work.

The data entity table has below attributes in common. All the other attributes are used to store the data related to the entity however below attributes are mandatory. We don't recommend composite primary key for the table rather use an ID_PK column to uniquely identify each row.

Column Name	Primary key	Nullable?	Data Type	Description
ID_PK	YES	No	VARCHAR2 (50 Byte)	ID to uniquely identify an entity in the system
VERSION		No	NUMBER	VERSION attribute used for optimistic lock
CREATED_TS		No	TIMESTAMP(6)	Creation timestamp of the record
DELETED_TS		YES	TIMESTAMP(6)	Soft-delete timestamp of the record
UPDATED_TS		No	TIMESTAMP(6)	The timestamp when the record is last updated
UPDATED_BY_USER		No	VARCHAR2 (50 Byte)	The userid which updated this record last
UPDATED_BY_TXN_ID		YES	VARCHAR2 (100 Byte)	The reference id of the transaction which updated the record
<ADDITIONAL ATTRIBUTES>		X	X	X

Understanding Reference Table structure

The reference data entity tables are meant to store key-value pairs. This is also termed as 'List of Values' (LOV). LOV tables have some soft rules to easily identify them from other tables as below

1. The Name of reference data entity must start with 'REF_' characters
2. The CONFIG_LANGUAGE_CODE_KEY attribute must be present for multilingual support
3. The entity must have KEY and VALUE column.
4. By default, Yugandhar Microservice Platform Reference data entity generated transactions does the search on KEY and CONFIG_LANGUAGE_CODE_KEY attributes. The same should be followed going forward. However if needed the code can be modified to do the searching based on VALUE or any other column as well.
5. The CONFIG_LANGUAGE_CODE_KEY and KEY attribute forms the unique constraint to restrict creating any duplicate values in the reference list.
6. The records of the reference data entities get cached in Yugandhar Microservice Platform using ehcache caching framework. This is done to improve the performance of real time transactions. You have the option to disable the caching either by removing the cache statement or changing the cache expiry time to few milliseconds. However it's not advisable to disable the caching unless there is a strong reason to do the same.

The reference data entity tables have below attributes in common.

Column Name	Primary key	Nullable?	Data Type	Description
ID_PK	Yes	N	VARCHAR2 (50 Byte)	ID to uniquely identify an entity in the system
VERSION		N	NUMBER	VERSION attribute used for optimistic lock
CREATED_TS		N	TIMESTAMP(6)	Creation timestamp of the record
DELETED_TS		Y	TIMESTAMP(6)	Soft-delete timestamp of the record
UPDATED_TS		N	TIMESTAMP(6)	The timestamp when the record is last updated
UPDATED_BY_USER		N	VARCHAR2 (50 Byte)	The userid which updated this record last
UPDATED_BY_TXN_ID		Y	VARCHAR2 (100 Byte)	The reference id of the transaction which updated the record

CONFIG_LANGUAGE_CODE_KEY	Unique Key	N	VARCHAR2 (50 Byte)	The language code which provide multilingual support. This language code is referred from application configuration table CONFIG_LANGUAGE_CODE. Not to be confused with REF_LANGUAGE_CODE, this is used to store the list of languages to be used in conjunction with data like preferred language of the person.
KEY	Unique Key	N	VARCHAR2 (50 Byte)	The key of the key-value pair from list of values
VALUE		N	VARCHAR2 (50 Byte)	The value of the key-value pair from list of values
DESCRIPTION		Y	VARCHAR2 (50 Byte)	Description if any related to key value pair

Understanding Configuration tables

The configuration tables starts with prefix 'CONFIG_<entityname>'. The structure of the entity is similar to the data entities mentioned above. Also most of the configuration entities are cached, yet there is no fixed guideline on the same. The configuration tables are exclusively used by Yugandhar Framework and data structure must not be customized by the user although new entries can be made in the table data.

Column Name	Primary key	Nullable?	Data Type	Description
ID_PK	YES	No	VARCHAR2 (50 Byte)	ID to uniquely identify an entity in the system
VERSION		No	NUMBER	VERSION attribute used for optimistic lock
CREATED_TS		No	TIMESTAMP(6)	Creation timestamp of the record
DELETED_TS		YES	TIMESTAMP(6)	Soft-delete timestamp of the record
UPDATED_TS		No	TIMESTAMP(6)	The timestamp when the record is last updated
UPDATED_BY_USER		No	VARCHAR2 (50 Byte)	The userid which updated this record last
UPDATED_BY_TXN_ID		YES	VARCHAR2 (100 Byte)	The reference id of the transaction which updated the record

Understanding Audit Log table structure

The Audit Log tables store the history Insert, update and delete operations performed on the base database entity. Audit log tables have some soft rules to easily identify them from other tables as below

1. The name of the Audit Log table starts with 'AL_<Name of the base entity>' e.g. the Audit table of the REF_CURRENCY table is AL_ REF_CURRENCY. The name can be changed if needed.
2. The Audit log table must have below mentioned attributes along with all the attributes of the base entity.
3. The AUDITLOG_ACTION_CODE gets updated as per below logic
    a. If a new record is created in base entity (e.g. REF_CURRENCY) then audit log trigger will create a new record in corresponding audit log table (e.g. AL_REF_CURRENCY) with AUDITLOG_ACTION_CODE as 'I'.
    b. If a new record is updated in base entity (e.g. REF_CURRENCY) then audit log trigger will create a new record in corresponding audit log table (e.g. AL_REF_CURRENCY) with AUDITLOG_ACTION_CODE as 'U'.
    c. If a new record is deleted in base entity (e.g. REF_CURRENCY) then audit log trigger will create a new record in corresponding audit log table (e.g. AL_REF_CURRENCY) with AUDITLOG_ACTION_CODE as 'D'.
4. Audit log tables will have multiple rows for each operation performed on every record giving a full audit history.
5. Currently the Audit Log tables do not have any purging policy provided as the purging requirements are highly dissimilar across industries. So your Audit log data size may continue to rise. It is advisable to disable the audit history (by disabling relevant database triggers) if audit information is not needed or define a policy to purge in project design phase itself.

Additional Attributes in Audit log table in addition to the attributes of base table.

Column Name	Primary key	Nullable?	Data Type	Description
AUDITLOG_ID_PK	YES	N	VARCHAR2 (50 Byte)	ID to uniquely identify an entity in the system
AUDITLOG_CREATED_TS		N	TIMESTAMP(6)	The timestamp when audit history record is created
AUDITLOG_ACTION_CODE		N	CHAR (1 Byte)	The action code of the audit log record based on base table

Understanding Application Configuration entities

CONFIG_APP_PROPERTIES - The table stores the application properties, should not be changed unless needed

CONFIG_ERRORCODE_REGISTRY –

This table stores all the error code and error messages being referred in MDM application

CONFIG_LANGUAGE_CODE

Configuration Language Codes supported by Application

CONFIG_TXN_REGISTRY

The table is reference list storing the list of all the services available for Yugandhar MDM

List of Reference Data (LOV) entities

- REF_COUNTRY_ISO
- REF_CURRENCY
- REF_LANGUAGE_CODE

List ofAudit History Logging entities

- AL_AUTH_ROLES_REGISTRY
- AL_AUTH_USER_REGISTRY
- AL_AUTH_USER_ROLE_ASSOC
- AL_AUTH_USERROLE_ACCESSCONTROL
- AL_CONFIG_APP_PROPERTIES
- AL_CONFIG_ERRORCODE_REGISTRY
- AL_CONFIG_LANGUAGE_CODE
- AL_CONFIG_TXN_REGISTRY
- AL_REF_COUNTRY_ISO
- AL_REF_CURRENCY
- AL_REF_LANGUAGE_CODE