Contents

[Liquibase Integration 2](#_Toc64862027)

[Approach 2](#_Toc64862028)

[Pre-requisite: 3](#_Toc64862029)

[Steps 3](#_Toc64862030)

[DB tables in reference to track to Liquibase DB updates 4](#_Toc64862031)

[Liquibase Integration – Configuration and sample setup files 4](#_Toc64862032)

[Liquibase-<env>. properties 4](#_Toc64862033)

[Commonly used Liquibase Commands: 5](#_Toc64862034)

[Changeset 5](#_Toc64862035)

[Master-changelog 5](#_Toc64862036)

[Example - pom.xml file 6](#_Toc64862037)

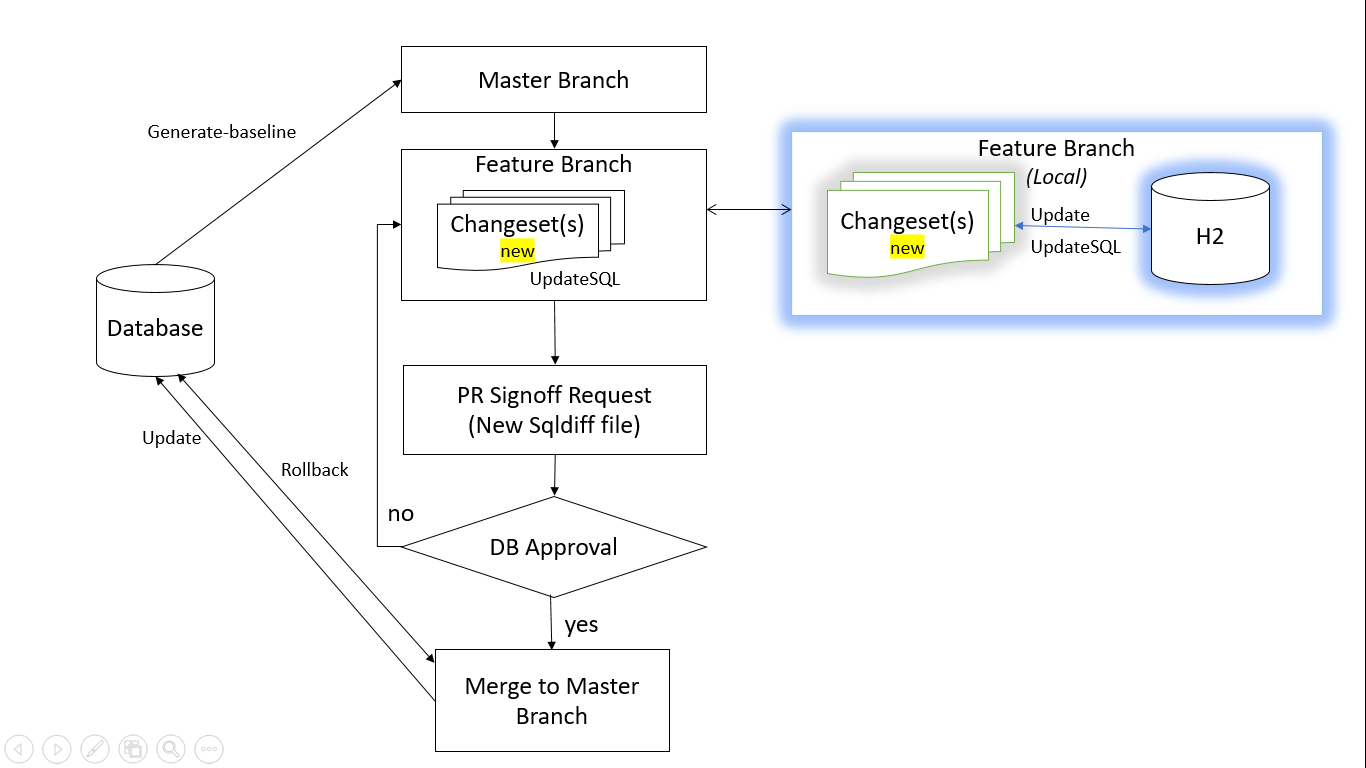
# Liquibase Integration

Liquibase is an automation solution for database change management. To provide synchronization scripts to the Database created along with the MS code in the repository. By integrating the Liquibase DB automation scripts with the code in the version control system, in the continuous integration platform.  it enables us to keep in sync the DB versions with the MS code in the deployment environment. Provide automated approach for DB changes in-line with software releases and promote through environments.

Liquibase uses changeset(s) to create/update/modify the DB instance and add capabilities to rollback any changes on a specific DB instance. Allows us to manage revisions and enables to track DB tasks performed easily.

Approach-

The below approach is put together for a kickstart on implementing the Liquibase Database change management system. To help new users adapt the Liquibase Developer Workflow as quickly as possible.



* Changeset – Database script, allows you to group database change types together and is considered to be a unit of change that Liquibase executes on a database.
* H2 Database – Liquibase uses H2 Databases on Windows, helps the developer to test DB scripts with the service level code developed or updated.
* Generate-baseline, Update, UpdateSQL, Rollback, etc – Are Liquibase commands run on the Database. Baseline command creates xml document with a snapshot of the existing Database schema. While the Update, UpdateSQL, Rollback are Database specific Liquibase commands. Refer the [Liquibase commands tutorial here](https://docs.liquibase.com/commands/community/home.html)

### Pre-requisite:

1. JIRA ticket is opened for requirement for a database change and/or MS code comes in as a User story.
2. Development Team works on the new requirement and communicate the upcoming Database related changes to the Database team
3. Follow the link [H2 Database](https://docs.liquibase.com/workflows/database-setup-tutorials/h2.html?Highlight=H2%20), for setup and configuration information.
4. Development team has verified there exists a snapshot of the Database baseline, else if does not exist create/generate file generate-baseline.xml file via TeamCity job from the latest live/production-equivalent database version.

Steps:

1. Developer makes sure the initial steps are considered.
2. Developer creates a Feature branch from Master.
3. Developer create Database relevant changes in [changeset file](#_Changeset)(s) in a separate folder in the Feature branch.
4. A ‘[**masterchangelog.xml’**](#_Master-changelog) needs to be maintained adding the **generated-baseline.xml** file and add/include the list of changeset files created.
5. Run Liquibase **'updateSql'** command against DEV database to obtain the differential updates. An output file of this command will return Sqldiff sql **‘migrate.sql’** file.
6. Complete MS code changes and conducts Unit testing.
7. The Sqldiff sql file is sent along with PR request to DB team/Dev Lead for reference in the signoff process.
8. Once PR is approved, MS code is merged to Master.
9. Build the MS code.
10. The Liquibase ‘**update’** command is used to apply the new DB changes in the specific environment.
11. Run the ‘**liquibase-tag’** command to tag the version in the Database. This ensures that a version tag is given to the list of ALL necessary Database changes planned for this sprint are committed for that Sprint/Development ready for release to test and deploy to QA/NFT/Prod.
12. Incase of an issue “**rollback**” command for specific **liquibase-tag** version applied to the Database.
13. Deploy the service in Dev environment.
14. Do the sanity in Dev environment.
15. If java coding issue, fix it and deploy it in DEV environment.
16. If sanity test is completed successfully , run Liquibase-update against QA environment in TeamCity
17. If MS deployment to QA is fine, test is completed, we can repeat the above steps for deployments in higher environments in TeamCity.

**Note** – The generated baseline file generated from live Oracle DB may contain token type **‘\*’ for Number field,** which may not be recognized by H2 Database local setup. So it is recommended to set the parameter value for columnSize in the properties file

For eg: In the **generate-baseline.xml** generated from Oracle Database.

*“*<changeSet author="balajpc (generated)" id="1613542393559-7">  
 <createTable tableName="LND\_EDR\_CUST\_MENU\_TB">  
 <column name="EDR\_CUST\_MENU\_SEQ" type="NUMBER(**${columnSize},** 0)">  
 <constraints nullable="false" primaryKey="true" primaryKeyName="PK\_EDRCMT"/>  
 </column>

*“*

Configurations needed for testing -

**Liquibase-h2.properties**

**“**

driver=org.h2.Driver  
url=jdbc:h2:tcp://localhost:8091/~/LND\_APP\_OWNER\_EDR;MODE=Oracle;SCHEMA=LND\_APP\_OWNER;  
username=sa  
**parameter.columnSize=38**

**“**

DB tables in reference to track to Liquibase DB updates **for a specific Microservice -**

1. DATABASECHANGELOG - Database Changelog
2. DATABASECHANGELOGLOCK - Database Changelog Lock

## Liquibase Integration – Configuration and sample setup files

Liquibase-<env>. properties contain connection details

There can be more than 1 maven profiles for running in different modes and against different environments.

Liquibase-<env>. properties :-

*driver=oracle. jdbc. OracleDriver*

*url=jdbc:oracle:thin:@vstdogb01d.server.rbsgrp.mde:1535:DOGBLN1*

*# Below credentials are for reading real dev database, don't have schema owner perms, so can only generate from this*

*defaultSchemaName=APP\_XYZ*

*username=APP\_USERX*

*password=Lnd27example#123*

Sample Profiles, configured specific to environment  :-

\* Liquibase-h2 - local in-memory/file-based database for testing Liquibase. It helps making sure the DB scripts work properly in development environment.

\* Liquibase-dev - Development environment

#### Commonly used Liquibase Commands:

\* Liquibase-generate - scan the selected environment and generate Liquibase baseline

\* Liquibase-sync - create Liquibase changelog tables, making it look like we've always used Liquibase for existing schemas

\* Liquibase-update-generated-only - apply Liquibase baseline to the selected environment

\* Liquibase-update - apply full Liquibase changelogs to the selected environment

\* Liquibase-tag - apply a tag to the current Liquibase state

\* Liquibase-rollback - rollback a Liquibase change or multiple changes

\* Liquibase-updateSQL – used when you want to inspect the raw SQL before running the goal. Recommend this command to be run by developer to check for any changes that may have been added to the changelog file of your database, need to sync up.

#### Changeset

**Empty DB -> Apply Changeset1 -> DB Version 1 -> Apply Changeset2 -> DB Version 2 ... so on**

Sample Changesetfile.xml

*<changeSet author="xyz" id="010-LND-2021-add-customer-address-column">*  
*<addColumn tableName="LND\_EDR\_APPL\_TB">*  
*<column name="Address" type="VARCHAR2"/>*  
*</addColumn>*  
*</changeSet>*

#### Master-changelog

**Master Changelog -> Changelog1 -> Changeset1.a, Changeset1.b, Changeset1.c**

**-> Changelog1 -> Changeset2.a, Changeset2.b**

**... so on**

Master-changelog.xml file :-

<include file="prerequisites.xml" relativeToChangelogFile="true"/>  
<include file="generated-baseline.xml" relativeToChangelogFile="true"/>  
<include file="2021/011-LND-2021-add-customer-table.xml" relativeToChangelogFile="true"/>  
<include file="2021/012-LND-2021-add-newcolumn.xml" relativeToChangelogFile="true"/>  
<include file="2021/013-LND-2021-add-customer-address-column.xml" relativeToChangelogFile="true"/>

#### Example - pom.xml file

Configuring Liquibase attributes for a pom.xml file

...

Liquibase Maven plugin

*<pluginManagement>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<version>3.10.0</version>*

*<configuration>*

*<changeLogDirectory>${project.basedir}/src/main/resources/Liquibase/</changeLogDirectory>*

*<promptOnNonLocalDatabase>false</promptOnNonLocalDatabase>*

*<databaseChangeLogLockTableName>MSVC\_DATABASECHANGELOGLOCK</databaseChangeLogLockTableName>*

*<databaseChangeLogTableName>MSVC\_DATABASECHANGELOG</databaseChangeLogTableName>*

*</configuration>*

*<dependencies>*

*<dependency>*

*<groupId>com.oracle</groupId>*

*<artifactId>ojdbc</artifactId>*

*<version>12.1.0.2</version>*

*</dependency>*

*<dependency>*

*<groupId>com.h2database</groupId>*

*<artifactId>h2</artifactId>*

*<version>1.4.196</version>*

*</dependency>*

*</dependencies>*

*</plugin>*

*</plugins>*

*</pluginManagement>*

***……..***

*<profile>*

*<id>Liquibase-generate</id>*

*<build>*

*<defaultGoal>Liquibase:generateChangeLog</defaultGoal>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<outputTeamCitychangeLogFile>${project.basedir}/src/main/resources/Liquibase/generated-baseline.xml</outputTeamCitychangeLogFile>*

*<diffIncludeObjects>MSVC\_.\*</diffIncludeObjects>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-sync</id>*

*<build>*

*<defaultGoal>Liquibase:changelogSync</defaultGoal>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<changeLogFile>master-changelog.xml</changeLogFile>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-update</id>*

*<build>*

*<defaultGoal>Liquibase:update</defaultGoal>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<changeLogFile>master-changelog.xml</changeLogFile>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-updateSQL</id>*

*<build>*

*<defaultGoal>Liquibase:updateSQL</defaultGoal>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<changeLogFile>master-changelog.xml</changeLogFile>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-rollback</id>*

*<build>*

*<defaultGoal>Liquibase:rollback</defaultGoal>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<changeLogFile>master-changelog.xml</changeLogFile>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-update-generated-only</id>*

*<build>*

*<defaultGoal>Liquibase:update</defaultGoal>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<changeLogFile>generated-baseline.xml</changeLogFile>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-dev</id>*

*<build>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<propertyFile>${project.basedir}/src/main/resources/Liquibase/Liquibase-dev.properties</propertyFile>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

*<profile>*

*<id>Liquibase-h2</id>*

*<build>*

*<plugins>*

*<plugin>*

*<groupId>org.Liquibase</groupId>*

*<artifactId>Liquibase-maven-plugin</artifactId>*

*<configuration>*

*<propertyFile>${project.basedir}/src/main/resources/Liquibase/Liquibase-h2.properties</propertyFile>*

*<dropFirst>false</dropFirst>*

*</configuration>*

*</plugin>*

*</plugins>*

*</build>*

*</profile>*

**…….**