DB-DWH docker image repository

Table of Contents

The parts of the project can be divided into several parts, complementing each other. Basically, there are 2 groups of images:

- **parent (base image)** - these are the docker images that are used as base images for the target products. They can only contain the absolutely necessary functionality and content.
- **builder** - they are used to carry content, they should never be used for base image. Its function is in the multi-stage dockerfile, where it is used as driver storage for example.

1. /dockerfile

Detailed description about the docker images.

1.1. liquibase

- Image:
  - `icellmobilsoft/db-base-liquibase`
- Base: `icellmobilsoft/java11jre`

- Goal: to use a central liquibase in every project
- Extra ENVs:
  - **$LIQUIBASE_HOME** - liquibase main directory
  - **$DOCKER_LIQUIBASE_CHANGELOG** - liquibase changelog main directory
  - **$DOCKER_LIQUIBASE_CLASSPATH** - liquibase classpath main directory
- Installed packages:
  - **liquibase**
  - **OJDBC10.JAR files** in /lib

This image extends the java11jre image with a liquibase install. With this, it's possible to run PostgreSQL or Oracle database changelog scripts and install any required objects.

[Liquibase auto install dockerimage.drawio] | *Liquibase-auto-install-dockerimage.drawio.png*

Figure 1. Dockerimage

Use cases

1. Original liquibase image, but user and directory structure is modified to iCell standards. The usage is same as the original Liquibase docs. The ENV parameters help achieving a clear structure.
2. Installer image - As specified by internal iCellmobilsoft standards, allows the running of multiple liquibase commands in a specified order. Developers can make automated installers without complicated arguments.

Installer image

The concept of an installation is different from a simple Liquibase run in that it consists of several Liquibase runs and steps. The steps may have common parameters, but they may also have different parameters. We want to standardize this by using this installer image.

Installer image's goal

- To be all-in-one - as in, contain anything required for the install (default parameters, changelog files, versioning…), without outside requirements.
- Allow the usage of custom ENV paramaters, that is used by the Liquibase changesets (e.g.: schema names, prefixes, etc…)
- If the install consists of multiple steps, then they can be parameterized separately (properties file and ENVs)
- Minimum configuration at runtime (e.g.: url, username, password)

Currently a maximum of 99 steps can be used:

- **STEP_1** - to be used on a fresh, clean DB,with 'admin/root'. For the purposes of:
  - Schema initialization
  - Creation of DB users and privileges

    - Setup and configure DB extensions
- **STEP_2-99** - typically used for commands to be run by a user, who was created in STEP_1 For example:
    - table creation
    - index, constraint management
    - insert, update, delete data

[Liquibase auto install script.drawio] | *Liquibase-auto-install-script.drawio.png*

Figure 2. Installation process

To ensure uniform use, it was necessary to introduce some restrictions:

- **It's not possible to use Liquibase command line parameters**. If any other parameter is required then use Liquibase ENV
- Some of the Liquibase ENV parameters are ignored, and are listed in the table below. Every other ENV parameter works normally. More about this: Liquibase command precedence - Configuration Hierarchy.

Installer image settings

1. `$DOCKER_LIQUIBASE_CHANGELOG` - the directory to be used for Liquibase changelog scripts
2. `$LIQUIBASE_HOME/$AUTO_INSTALL/defaults-step-01.properties` and `$LIQUIBASE_HOME/$AUTO_INSTALL/defaults-step-02.properties` properties files, for the default parameters of STEP_1 and STEP_2. These parameters can be overridden by docker ENV parameters at installation.

Table 1. Control parameters

Docker ENV	Set of Values	Function
$AUTO_INSTALL	custom (postgres, oracle) <*1>	Docker image installation activation
$INSTALL_STEP	STEP_1 , STEP_2, … STEP_99 <*2>	Which step to run

① Its value depends on the directory structure in which the `defaults-step-*.properties` files are placed. For example the default is `$LIQUIBASE_HOME/*postgres/defaults-step-01.properties` and so `postgres`

② Only required for running individual steps. Currently, a maximum of 99 steps are possible, by default every step is run after the other.

[Liquibase auto install param.drawio] | *Liquibase-auto-install-param.drawio.png*

Figure 3. Parameter handling

Table 2. Runtime parmeters

Docker ENV	Step properties file	Disabled Liquibase parameter	Default value
INSTALL_URL or INSTALL_URL_S1 INSTALL_URL_S2	URL	LIQUIBASE_COMMAND_URL	jdbc:postgresql://localhost:5432/postgres
INSTALL_USERNAME or INSTALL_USERNAME_S1 INSTALL_USERNAME_S2	USERNAME	LIQUIBASE_COMMAND_USERNAME	postgres
INSTALL_PASSWORD or INSTALL_PASSWORD_S1 INSTALL_PASSWORD_S2	PASSWORD	LIQUIBASE_COMMAND_PASSWORD	postgres
INSTALL_CHANGELOGFILE or INSTALL_CHANGELOGFILE_S1 INSTALL_CHANGELOGFILE_S2	CHANGELOGFILE	LIQUIBASE_COMMAND_CHANGELOG_FILE	liquibase-install-default.xml
INSTALL_CONTEXTS or INSTALL_CONTEXTS_S1 INSTALL_CONTEXTS_S2	CONTEXTS	LIQUIBASE_COMMAND_CONTEXTS	
INSTALL_LABELS or INSTALL_LABELS_S1 INSTALL_LABELS_S2	LABELS	LIQUIBASE_COMMAND_LABELS	

These runtime parameters are used in related STEP_* The value precedence of the parameters is as follows (higher wins):

4.`INSTALL_URL_S1` / `INSTALL_URL_S2` (ENV parameter)
3.`INSTALL_URL` (ENV pareméter)
2.`URL` (`defaults-step-01.properties` / `defaults-step-02.properties` parameter)
1.default (`jdbc:postgresql://localhost:5432/postgres`)
0.`LIQUIBASE_COMMAND_URL` ignored

Example

```
docker run --rm \
```

```
  -e AUTO_INSTALL=postgresql \
  -e URL=jdbc:postgresql://url:5432/postgres \
  -e INSTALL_URL=jdbc:postgresql://install-url:5432/postgres \
  -e INSTALL_URL_S1=jdbc:postgresql://install-url-s1:5432/postgres \
  -e LIQUIBASE_COMMAND_URL=jdbc:postgresql://liquibase-command-url-s1:5432/postgres \
  icellmobilsoft/db-base-liquibase:$VERSION

# Liquibase install
# STEP_1: `jdbc:postgresql://install-url-s1:5432/postgres`
# STEP_2: `jdbc:postgresql://install-url:5432/postgres`
# URLs are going to be used.
```

Liquibase usage samples

Classic Liquibase

It can be used as usual as the original Liquibase, with properties, command line or ENV parameters.

Changelog attached at runtime

```
docker run --rm \
  -v ./test/liquibase/liquibase-
changelog.xml:/home/icellmobilsoft/liquibase/changelog/liquibase-changelog.xml \
  dockerhub.icellmobilsoft.hu/db-base-liquibase:$VERSION \
  update \
    --changelog-file=liquibase-defaults.properties \
    --url='jdbc:postgresql://localhost:5432/postgres' \
    --username=postgres \
    --password=postgres
```

Liquibase installer image

Sources are attached runtime, defaults are in the properties files

```
docker run --rm \
  -v ./test/liquibase/defaults-step-
01.properties:/home/icellmobilsoft/liquibase/postgresql/defaults-step-01.properties \
  -v ./test/liquibase/defaults-step-
02.properties:/home/icellmobilsoft/liquibase/postgresql/defaults-step-02.properties \
  -v ./test/liquibase:/home/icellmobilsoft/liquibase/changelog \
  -e AUTO_INSTALL=postgresql \
  icellmobilsoft/db-base-liquibase:$VERSION
```

① Here, we assume that in `defaults-step-01.properties` and `defaults-step-02.properties` files, the correct STEP_1 and STEP_2 values for `CHANGELOGFILE` are configured. Other settings Runtime parmeters.

Sources are attached runtime, controlled by ENV variables

```
docker run --rm \
  -v ./test/liquibase/defaults-step-
01.properties:/home/icellmobilsoft/liquibase/postgresql/defaults-step-01.properties \
  -v ./test/liquibase/defaults-step-
02.properties:/home/icellmobilsoft/liquibase/postgresql/defaults-step-02.properties \
  -v ./test/liquibase:/home/icellmobilsoft/liquibase/changelog \
  -e AUTO_INSTALL=postgresql \
  -e INSTALL_URL=jdbc:postgresql://install-url:5432/postgres \
  -e INSTALL_USERNAME_S1=pgadmin \
  -e INSTALL_PASSWORD_S1=pgpass \
  -e INSTALL_USERNAME_S2=schemauser \
  -e INSTALL_PASSWORD_S2=schemapass \
  dockerhub.icellmobilsoft.hu/db-base-liquibase:$VERSION
```

① Here, we assume that in `defaults-step-01.properties` and `defaults-step-02.properties` files, the correct STEP_1 and STEP_2 values for `CHANGELOGFILE` are configured. Other settings Runtime parmeters.

1.2. Postgres

- Image:
  - `icellmobilsoft/db-base-postgres_148`
- Base: `postgres:14.8-bullseye`
- Goal: Postgresql DB setup with CRON scheduler
- Installed packages:
  - **postgresql-14-cron**

1.3. pg_tools

- Image:
  - `icellmobilsoft/db-base-pg_tools`
- Base: `icellmobilsoft/db-base-liquibase`
- Goal: Setup a fixed pg_partman partition manager and then start the pg_cron scheduler in the Postgres image
- This image should be used in every project repository where a Postgres DB is used.
  - Installing: To be used in Step2, after the creation of the projects schema, since it will be installed inside that

2. Release notes

2.1. v0.1.0

- New image:
  - icellmobilsoft/db-base-liquibase - Based on liquibase 4.15.

2.2. v0.2.0

- New image:
  - icellmobilsoft/db-base-liquibase-bigquery - Not available.

2.3. v0.3.0

- New image:
  - icellmobilsoft/db-base-airflow - Not available.

2.4. v0.4.0

- New image:
  - icellmobilsoft/db-base-liquibase-bigquery-gc-cli - Not available.

2.5. v0.5.0

- New image:
  - icellmobilsoft/db-base-postgres - A Postgresql 14 image with Partition manager and a Cron scheduler.
- liquibase upgrade 4.15 → 4.21
- ENV versioning moved to .env file

2.6. v0.6.0

- INSTALL_STEPs are handled dinamically and extended to 99

2.7. v0.7.0

- OJDBC*.jar files are installed in the Liquibase image

2.8. v0.8.0

- In dockerfile/liquibase/bash/auto.install.sh file, the logic of INSTALL_PASSWORD_S* 'override' is now fixed.
- Multiplatform image build

2.9. v0.9.0

- postgresql-14-partman package version fixing to `5.0.0-1.pgdg110+1`, and postgresql-14-cron package version fixing to `1.6.2-1.pgdg110+1`.

IMPORTANT	Running the 5.0.0 partman extension does not create the `custom_time_partitions` table in the partman schema anymore, should be considered when running common liquibase commands.

- This error is fixed below:
- New image:
  - icellmobilsoft/db-base-pg_tools - contains the fixed pg_partman and handles the starting of pg_cron scheduler.
- partman is removed from the postgres image and only contains the CRON scheduler
- this pg_tools should be used in every future projects, where partitioning is required!

2.10. v0.10.0

- New open source dwh_ddl_change_tracker - Not available

2.11. v1.0.0

- Github release
  - icellmobilsoft/db-base-postgres renamed to icellmobilsoft/db-base-postgres_148
  - db-base-liquibase-bigquery,db-base-liquibase-bigquery-gc-cli, db-base-airflow images are removed

2.12. v1.1.0

- Postgresql 17.5 version added (icellmobilsoft/db-base-postgres_175)
- Liquibase 4.31 version added (icellmobilsoft/db-base-liquibase_431)
- Installed pg_jobmon 1.4.1 and pg_partman 5.0.1 extensions in the Postgresql 175 image.
- Upgrade pg_partman extension to 5.3.1 in the Postgresql 175 image.