Table of Contents

[Flyway 2](#_Toc106182791)

[How Flyway works 2](#_Toc106182792)

[Creating Database Scripts 3](#_Toc106182793)

[Running Flyway 3](#_Toc106182794)

[Commands 3](#_Toc106182795)

[Flyway + SpringBoot Integration 4](#_Toc106182796)

[Flyway + Maven Plug-in Integration 4](#_Toc106182797)

[Flyway – Commands 6](#_Toc106182798)

[1. migrate 7](#_Toc106182799)

[2. info 7](#_Toc106182800)

[3. validate 8](#_Toc106182801)

[4. repair 8](#_Toc106182802)

[5. clean 9](#_Toc106182803)

[Ref. 9](#_Toc106182804)

# Flyway

Logo, company name

Description automatically generated

* Recreate a database from scratch
* Make it clear at all times what **state** a database is in
* **Migrate** in a deterministic way from your current version of the database to a newer one

## How Flyway works

1. We have an empty database called flywaydb
2. When we run flyway first time, it will create flyway\_schema\_history table.
3. This table will be used to track the state of the database.
4. Flyway will begin **scanning** the filesystem or the classpath of the application for **migrations**.
5. The migrations are then **sorted** based on their **version number** and applied in order:
6. As each migration gets applied, the schema\_history table is updated accordingly

Diagram

Description automatically generated

Simply create a new migration with a version number higher than the current one. When next time Flyway starts, it will find it and upgrade the database accordingly.

Creating Database Scripts

Database Scripts consists of the following parts:

* **Prefix**: V for versioned ([configurable](https://flywaydb.org/documentation/configuration/parameters/sqlMigrationPrefix)), U for undo ([configurable](https://flywaydb.org/documentation/configuration/parameters/undoSqlMigrationPrefix)) and R for repeatable migrations ([configurable](https://flywaydb.org/documentation/configuration/parameters/repeatableSqlMigrationPrefix))
* **Version**: Version with dots or underscores separate as many parts as you like (Not for repeatable migrations)
* **Separator**: \_\_ (two underscores) ([configurable](https://flywaydb.org/documentation/configuration/parameters/sqlMigrationSeparator))
* **Description**: Underscores or spaces separate the words
* **Suffix**: .sql ([configurable](https://flywaydb.org/documentation/configuration/parameters/sqlMigrationSuffixes))

A picture containing graphical user interface

Description automatically generated

Running Flyway

Flyway supports a range of different options to run database migrations:

* via [command line](https://reflectoring.io/#command-line)
* via [Java API](https://reflectoring.io/#java-api),
* via **Maven** and [Gradle](https://reflectoring.io/#gradle-plugin) plugins, and
* via [community plugins and integrations](https://flywaydb.org/documentation/plugins/) including [**Spring Boot**](https://reflectoring.io/#spring-boot-auto-configuration)**.**

Commands

Flyway relies on seven commands to manage database version control.

* migrate**:** Migrates the schema to the latest version. Flyway will create the schema history table automatically if it doesn’t exist.
* clean**:** Drops all objects in the configured schemas.
* info**:** Prints the details and status information about all the migrations.
* validate**:** Validates the applied migrations against the available ones.
* undo**:** Undoes the most recently applied versioned migration.
* baseline**:** Baselines an existing database, excluding all migrations up to and including baselineVersion.
* repair**:** Repairs the schema history table.

## Flyway + SpringBoot Integration

1.We need to Create SpringBoot project & add Flyway depency in pom.xml

<dependency>

<groupId>org.flywaydb</groupId>

<artifactId>flyway-core</artifactId>

</dependency>

2. Create Database migration scripts under resources/db/migration/ folder. We have created 3 scripts.

Graphical user interface, text, application

Description automatically generated

3.Update application.properties with config. Details

spring.datasource.url=jdbc:postgresql://127.0.0.1:5432/flywaydb

spring.datasource.username=postgres

spring.datasource.password=postgres

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

spring.jpa.hibernate.ddl-auto=update

spring.flyway.baseline-on-migrate = true

spring.mvc.view.prefix: /WEB-INF/jsp/

spring.mvc.view.suffix: .jsp

4. Start the application & it will execute scripts order by version numbers & saves history in <DATABASE>\_schema\_history table

Graphical user interface

Description automatically generated

## Flyway + Maven Plug-in Integration

1. First, Disable flyway in SpringBoot

spring.flyway.enabled=false

2. Update pom.xml-file in the build/plugins-section with Flyway Configuration details

<properties>

<java.version>11</java.version>

<database.url>jdbc:postgresql://localhost:5432/flywaydb</database.url>

<database.user>postgres</database.user>

<databese.password>postgres</databese.password>

</properties>

<dependencies>

<dependency>

<groupId>org.flywaydb</groupId>

<artifactId>flyway-core</artifactId>

</dependency>

<dependency>

<groupId>org.postgresql</groupId>

<artifactId>postgresql</artifactId>

<scope>runtime</scope>

</dependency>

</dependencies>

To perform Flyway migration at mvn build process itself, we need configure goals in flyway-maven-plugin. Now whenever we run mvn install command, Flyway migration automactally happen.

<plugin>

<groupId>org.flywaydb</groupId>

<artifactId>flyway-maven-plugin</artifactId>

<executions>

<execution>

<id>echo-intsall-phase</id>

<phase>install</phase>

<goals>

<goal>migrate</goal>

</goals>

</execution>

</executions>

<configuration>

<sqlMigrationSeparator>\_\_</sqlMigrationSeparator>

<locations>

<location>filesystem:src/main/resources/db/migration</location>

</locations>

<url>${database.url}</url>

<user>${database.user}</user>

<password>${databese.password}</password>

</configuration>

<dependencies>

<dependency>

<groupId>org.postgresql</groupId>

<artifactId>postgresql</artifactId>

<version>42.3.6</version>

</dependency>

</dependencies>

</plugin>

3.Run mvn flyway:migrate command to perform Flyway migration.

Or mvn clean install

satyakaveti@Satyas-MacBook-Pro-2 FlywayPOC % mvn flyway:migrate

[INFO] Scanning for projects...

[INFO]

[INFO] -----------------------< com.dpworld:FlywayPOC >------------------------

[INFO] Building FlywayPOC 0.0.1-SNAPSHOT

[INFO] --------------------------------[ jar ]---------------------------------

[INFO]

[INFO] --- flyway-maven-plugin:8.0.5:migrate (default-cli) @ FlywayPOC ---

[INFO] Flyway Community Edition 8.0.5 by Redgate

[INFO] Database: jdbc:postgresql://localhost:5432/flywaydb (PostgreSQL 14.3)

[INFO] Successfully validated 3 migrations (execution time 00:00.010s)

[INFO] Creating Schema History table "public"."flyway\_schema\_history" ...

[INFO] Current version of schema "public": << Empty Schema >>

[INFO] Migrating schema "public" to version "1 - createTable"

[INFO] Migrating schema "public" to version "2 - updateTable"

[INFO] Migrating schema "public" to version "3 - insertData1"

[INFO] Successfully applied 3 migrations to schema "public", now at version v3 (execution time 00:00.042s)

[INFO] Flyway Community Edition 8.0.5 by Redgate

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 1.142 s

[INFO] Finished at: 2022-06-09T12:49:02+05:30

[INFO] ------------------------------------------------------------------------

**Maven Build Commands**

mvn flyway:migrate

mvn clean install -DskipTests=true flyway:migrate

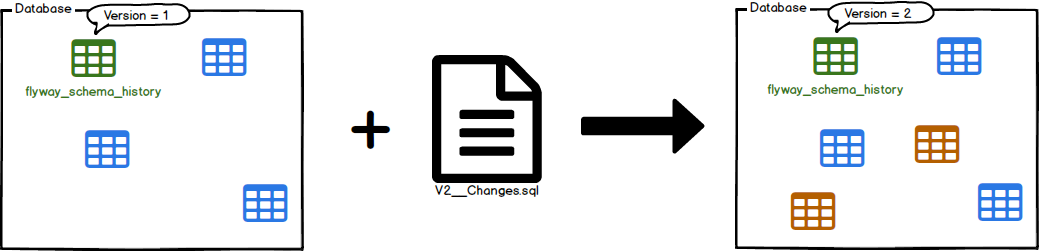
## Flyway – Commands

Flyway relies on seven commands to manage database version control.

* migrate**:** Migrates the schema to the latest version. Flyway will create the schema history table automatically if it doesn’t exist.
* clean**:** Drops all objects in the configured schemas.
* info**:** Prints the details and status information about all the migrations.
* validate**:** Validates the applied migrations against the available ones.
* undo**:** Undoes the most recently applied versioned migration.
* baseline**:** Baselines an existing database, excluding all migrations up to and including baselineVersion.
* repair**:** Repairs the schema history table.

1. migrate

It will scan the filesystem or your classpath for available migrations. It will compare them to the migrations that have been applied to the database. If any difference is found, it will migrate the database to close the gap.



mvn flyway:migrate

Graphical user interface

Description automatically generated

2. info

Prints the details and status information about all the migrations.

mvn flyway:info

A picture containing table

Description automatically generated

3. validate

Validate applied migrations against resolved ones (on the filesystem or classpath) to detect accidental changes that may prevent the schema(s) from being recreated exactly.

Validation fails if

* differences in migration names, types or checksums are found
* versions have been applied that aren’t resolved locally anymore
* versions have been resolved that haven’t been applied yet

mvn flyway:validate

I have deleted 6 version from DB manually. If I validate, it will show validation failed because Scripts inside the code folder & Database versions are not same.

Text

Description automatically generated

4. repair

Repairs the Flyway schema history table. This will perform the following actions:

* Remove any failed migrations on databases without DDL transactions  
  (User objects left behind must still be cleaned up manually)
* Realign the checksums, descriptions and types of the applied migrations with the ones of the available migrations

mvn flyway:repair

Graphical user interface, text

Description automatically generated

5. clean

Drops all objects in the configured schemas. Clean is a great help in development and test. It will effectively give you a fresh start, by wiping your configured schemas completely clean. All objects (tables, views, procedures, …) will be dropped

mvn flyway:clean

Text

Description automatically generated

## Ref.

* <https://dzone.com/articles/flyway-vs-liquibase>
* <https://flywaydb.org/documentation/getstarted/how>
* <https://reflectoring.io/database-migration-spring-boot-flyway/>
* <https://pavankjadda.medium.com/safely-evolving-database-with-liquibase-spring-data-and-spring-boot-9c8d2aab1537>