DevOps Meetup DUS, 12.07.2017

Continuous Database Integration with Flyway

Sandra Parsick
mail@sandra-parsick.de
@SandraParsick

About me

- Sandra Parsick
- Freelance Sofware Developer in Java environment
- Focus areas:
  - Java enterprise applications
  - agile methods
  - Software Craftmanship
  - Automation of development process
- trainings
- workshops
- Softwerkskammer Ruhrgebiet

- Twitter: @SandraParsick
- Blog: http://blog.sandra-parsick.de
- E-Mail: mail@sandra-parsick.de

Agenda

- Continuous Database Integration (CDBI)
- Flyway
- Pitfalls

Continuous Database Integration

- Definition
- Motivation
- How to set up

Definition

„Continuous Database Integration (CDBI) is the process of rebuilding your database and test data any time a change is applied to a project's version control repository“

(by Continuous Integration by Paul M. Duvall, Steve Matyas und Andrew Glover)

Motivation

- One shared test database for all developer.
- Nobody knows which database migration script was run on which database instance.
- Test database differs from productive database
- Database migration scripts are distributed in ticket system, developer's system etc.

How To Set Up

- Treat database code like a normal source code
  - ➔ Put all database assets (DDL, DML, configurations, test data, stored procedures, functions etc) in your version control system.
  - ➔ Test your database code after every change.
- Give every developer his own database / Make test database being similar to the productiv database.
  - ➔ Set up the database by build scripts.
- All database changes are transparent
  - – Change history

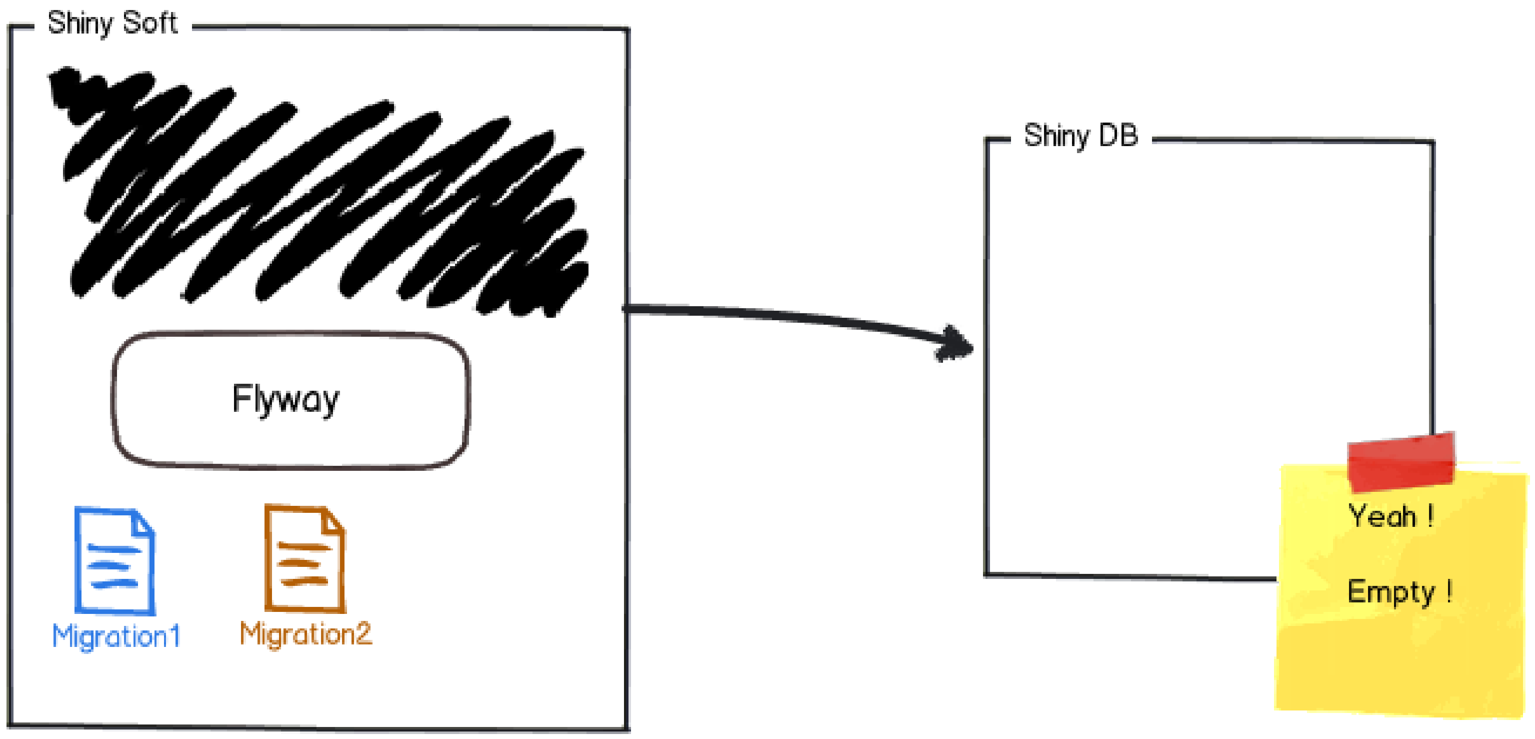
Flyway

- What is Flyway?
- How Does Flyway Work?
- How to write scripts for migration with Flyway?
- What is not possible with Flyway?
- How to use Flyway?

What is Flyway?

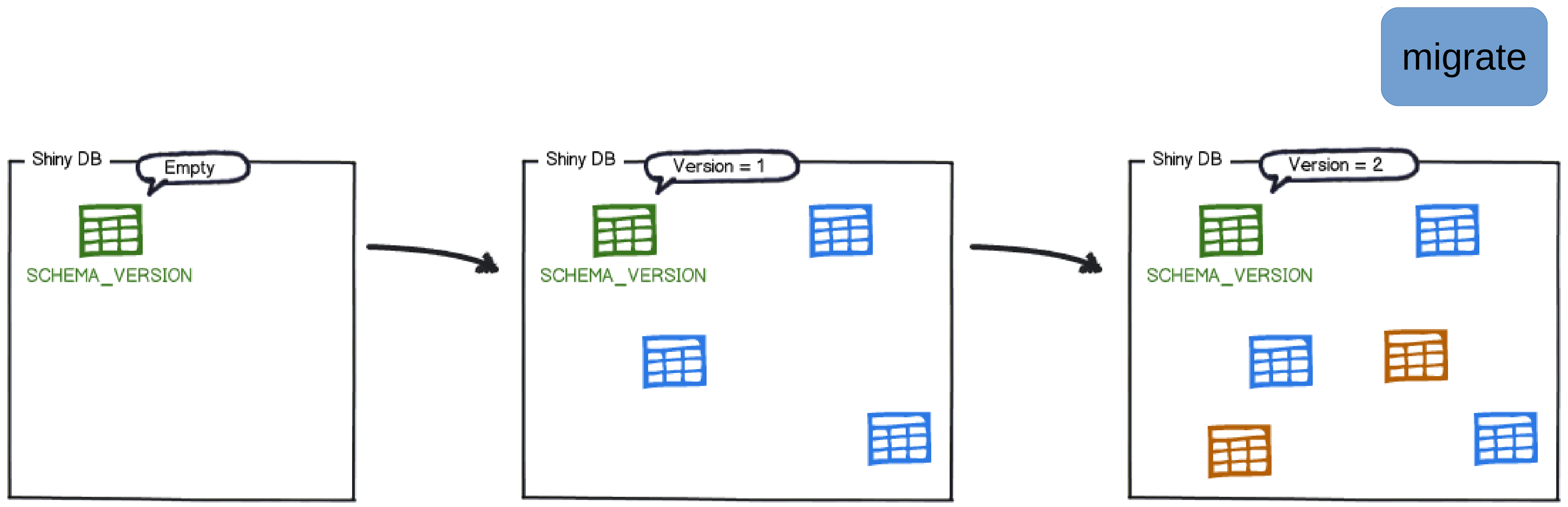
- Database migration framework based on Java
- Recreate a database from scratch
- Make it clear at all times what state a database is in
- Migrate in a deterministic way from your current version of the database to a newer one
- Four migrations:
    - SQL- and Java-based migration
    - Versionbased and repeatable migration
- Current version: 4.2.0
- Homepage: http://flywaydb.org/
- Twitter: @flywaydb

How does Flyway work?



Reference: flywaydb.org

How does Flyway work?



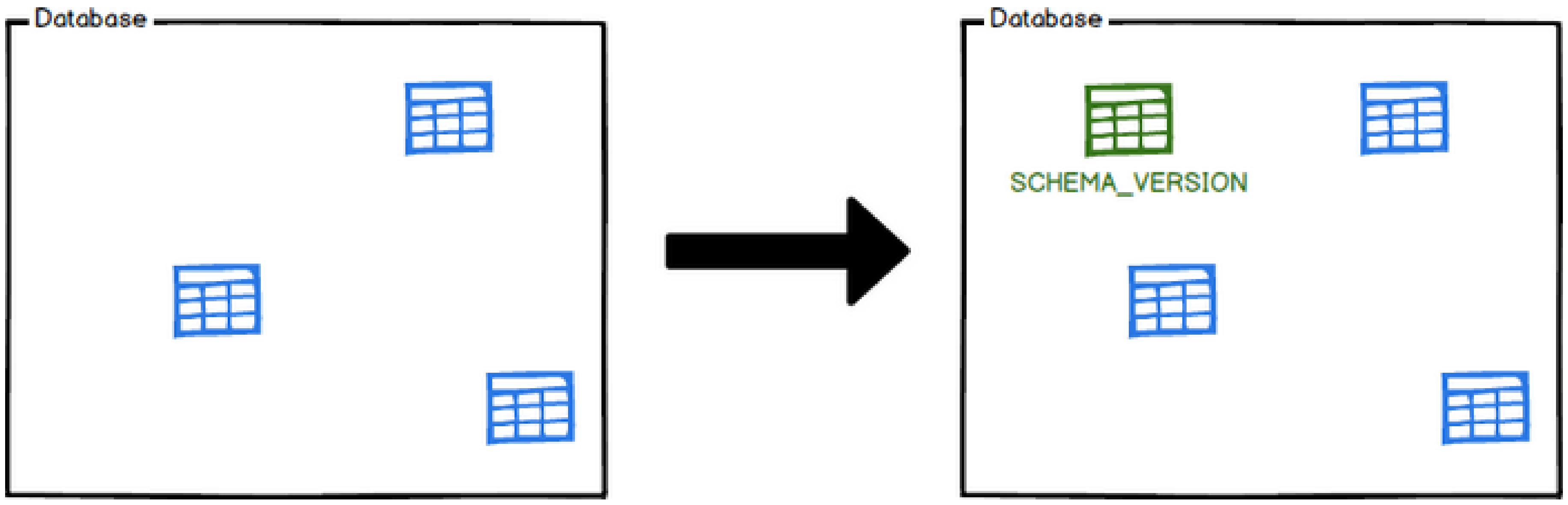
Reference: flywaydb.org

schema_version

installed_rank	version	description	type	script	checksum	installed_by	installed_on	execution_time	success
1	1	Initial Setup	SQL	V1__Initial_Setup.sql	1996767037	axel	2016-02-04 22:23:00.0	546	true
2	2	First Changes	SQL	V2__First_Changes.sql	1279644856	axel	2016-02-06 09:18:00.0	127	true

How does Flyway work?

baseline



Reference: flywaydb.org

Migration Scripts

- Four possibilities

	Version-based	Repeatable
SQL-based	✓	✓
Java-based	✓	✓

Version-based Migration

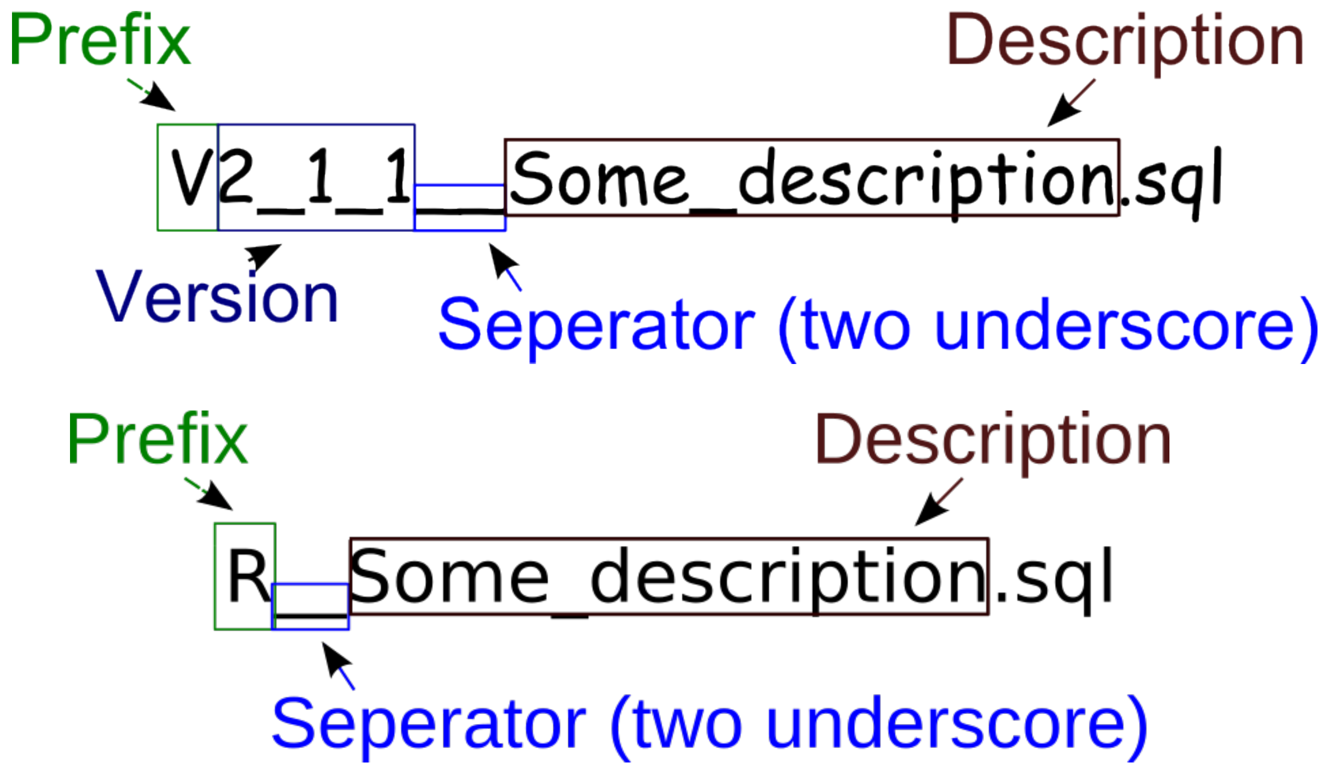
- **Characteristics**
  - Scripts have a unique version
  - They run only once
- **Typical usage**
  - DDL changes (CREATE/ALTER/DROP für TABLES,INDEXES,FOREIGN KEYS,...)
  - Simple data changes

Repeatable Migration

- **Characteristics**
  - Scripts have no version
  - They are executed if their check sum is changed
  - They are executed after all version-based scripts have been executed
- **Typical usage**
  - (Re-) Creation of views / procedures / functions / packages / ...
  - Bulk reimport of master data

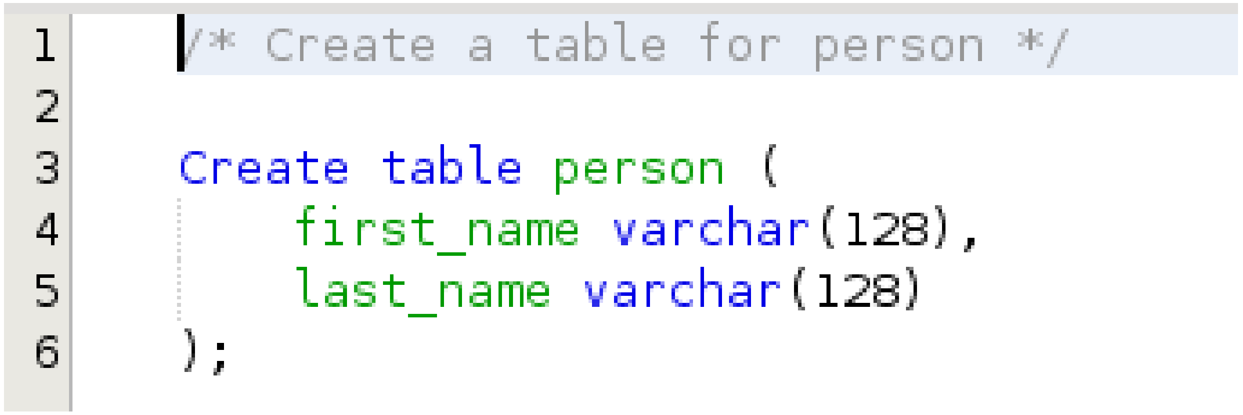
SQL Migration

- **Typical usage**
  - DDL changes (CREATE/ALTER/DROP für TABLES,VIEWS,TRIGGERS,SEQUENCES,...)
  - Simple data changes
- **Naming**


Prefix
Description
V2_1_1__Some_description.sql
Version
Seperator (two underscore)
Prefix
Description
R__Some_description.sql
Seperator (two underscore)

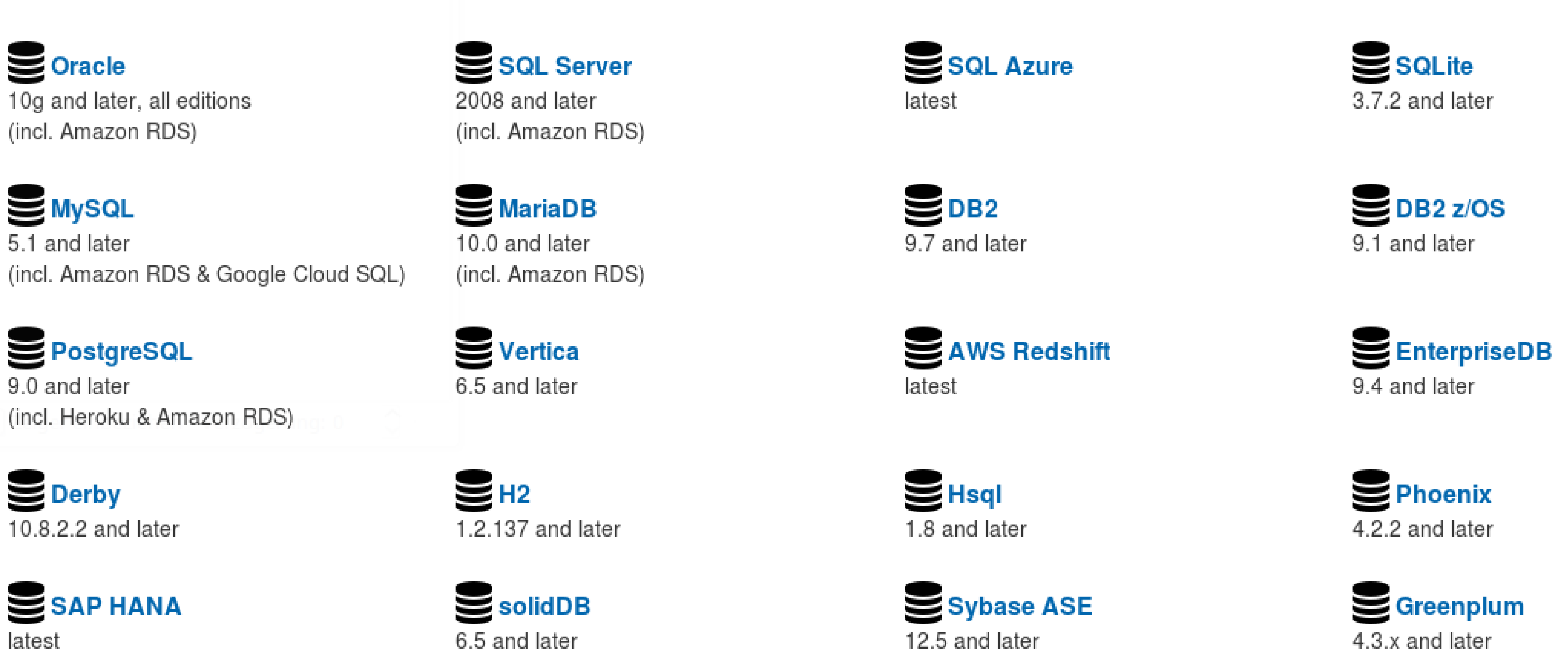
SQL Migration

- **Syntax**
  - Single or multi line statements
  - Flexible placeholder replacement
  - Single (–) or multi line (/* */) comments
  - Database-specific SQL syntax extensions
- **Example**



```sql
/* Create a table for person */

Create table person (
    first_name varchar(128),
    last_name varchar(128)
);
```

Supported Databases



Oracle
10g and later, all editions
(incl. Amazon RDS)

SQL Server
2008 and later
(incl. Amazon RDS)

SQL Azure
latest

SQLite
3.7.2 and later

MySQL
5.1 and later
(incl. Amazon RDS & Google Cloud SQL)

MariaDB
10.0 and later
(incl. Amazon RDS)

DB2
9.7 and later

DB2 z/OS
9.1 and later

PostgreSQL
9.0 and later
(incl. Heroku & Amazon RDS)

Vertica
6.5 and later

AWS Redshift
latest

EnterpriseDB
9.4 and later

Derby
10.8.2.2 and later

H2
1.2.137 and later

Hsql
1.8 and later

Phoenix
4.2.2 and later

SAP HANA
latest

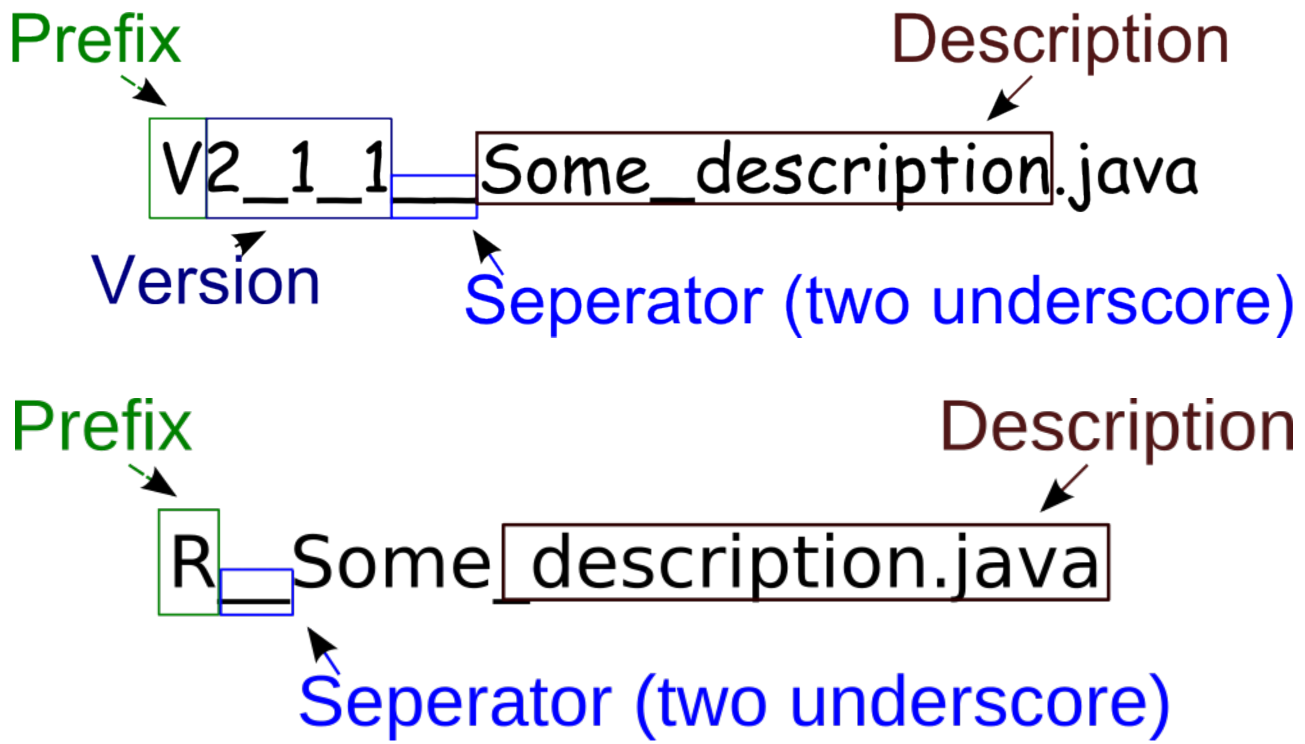
solidDB
6.5 and later

Sybase ASE
12.5 and later

Greenplum
4.3.x and later

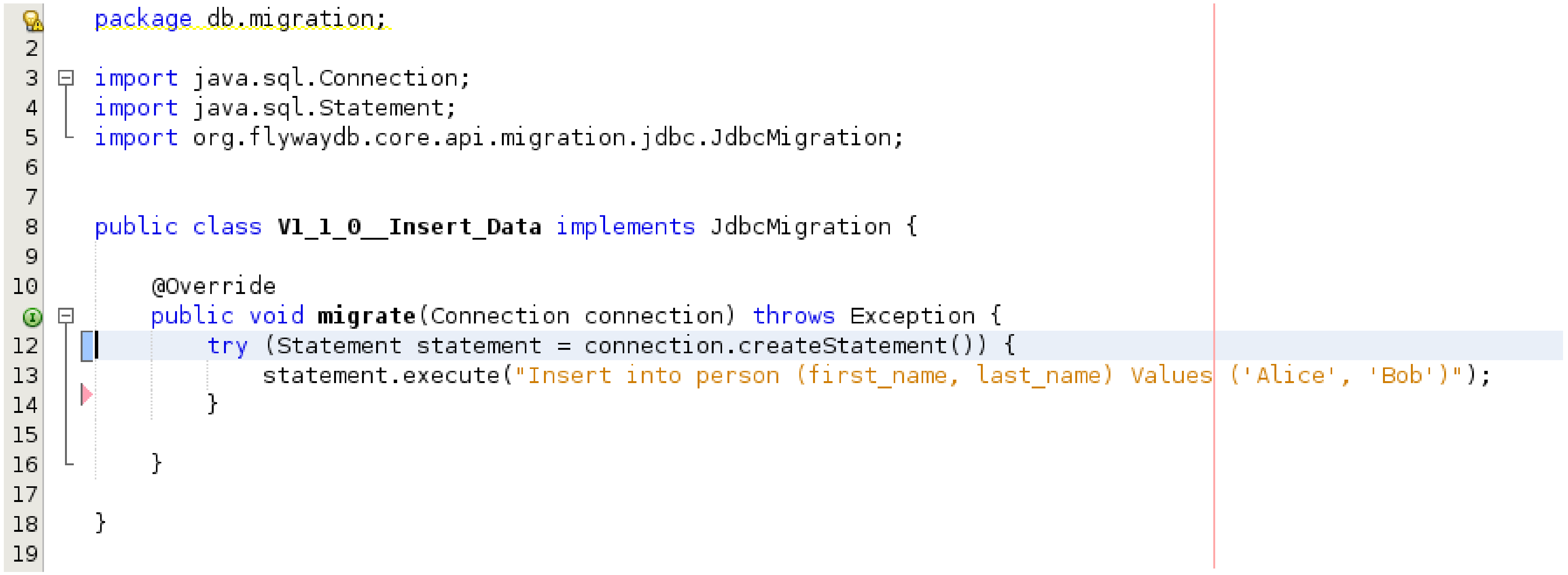
Java Migration

- **Typical usage**
  - BLOB & CLOB changes
  - Advanced bulk data changes (Recalculations, advanced format changes, …)
- **Naming**


Prefix
Description
V2_1_1__Some_description.java
Version
Seperator (two underscore)
Prefix
Description
R__Some_description.java
Seperator (two underscore)

Java Migration

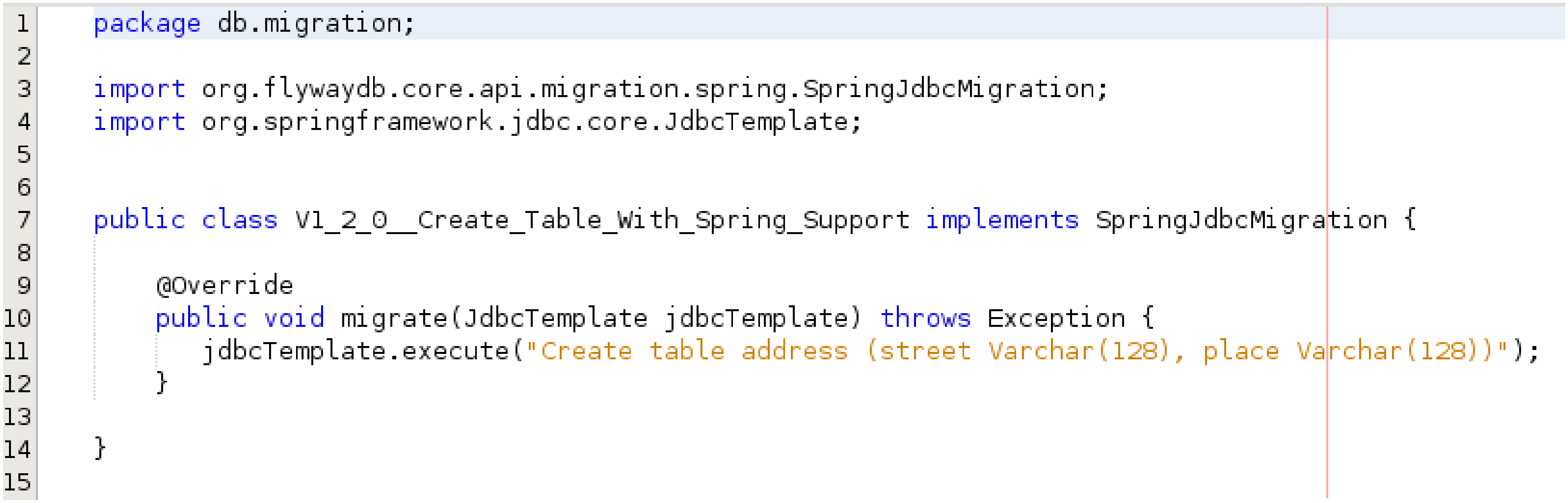
Example



```
package db.migration;

import java.sql.Connection;
import java.sql.Statement;
import org.flywaydb.core.api.migration.jdbc.JdbcMigration;


public class V1_1_0__Insert_Data implements JdbcMigration {

    @Override
    public void migrate(Connection connection) throws Exception {
        try (Statement statement = connection.createStatement()) {
            statement.execute("Insert into person (first_name, last_name) Values ('Alice', 'Bob')");
        }

    }

}
```

Java Migration

Example Spring Support

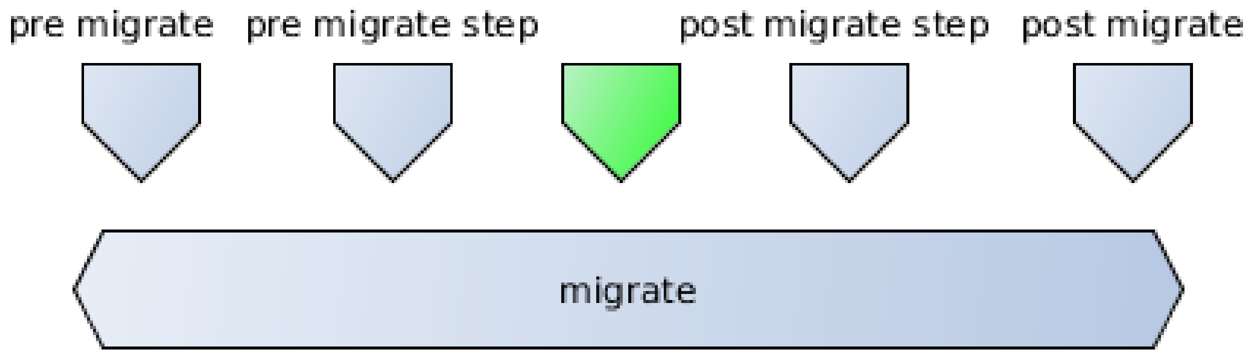


```java
package db.migration;

import org.flywaydb.core.api.migration.spring.SpringJdbcMigration;
import org.springframework.jdbc.core.JdbcTemplate;


public class V1_2_0__Create_Table_With_Spring_Support implements SpringJdbcMigration {

    @Override
    public void migrate(JdbcTemplate jdbcTemplate) throws Exception {
       jdbcTemplate.execute("Create table address (street Varchar(128), place Varchar(128))");
    }

}
```

Advanced Migrations - Callbacks

- **Typical usage**
  - Stored Procedure Compilation
  - Materialized View Update

- **Flyway Lifecycle (Example migrate)**



SQL Callbacks

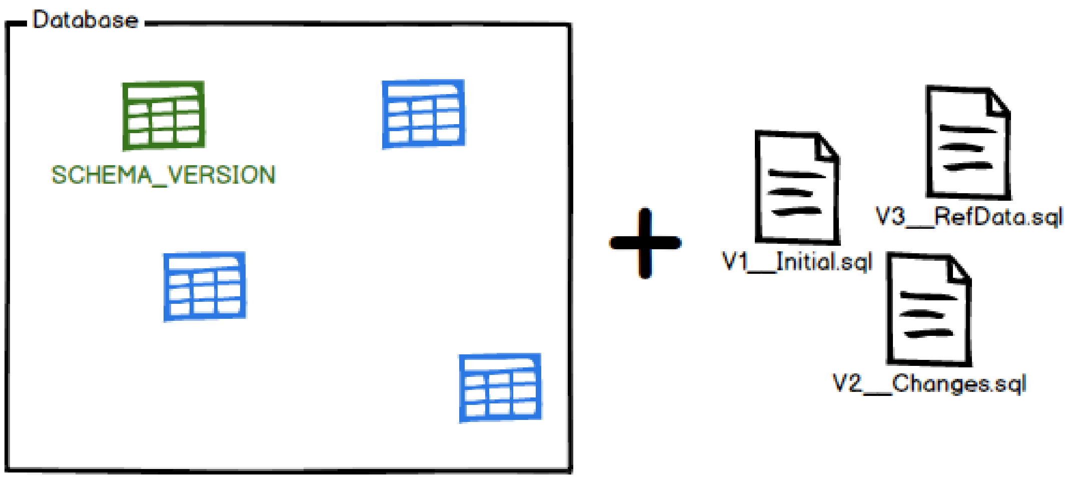
- **Example migrate-Lifecycle:**
  - SQL callback scripts are indicated by naming:
    - BeforeMigrate.sql
    - BeforeEachMigrate.sql
    - AfterEachMigrate.sql
    - AfterMigrate.sql

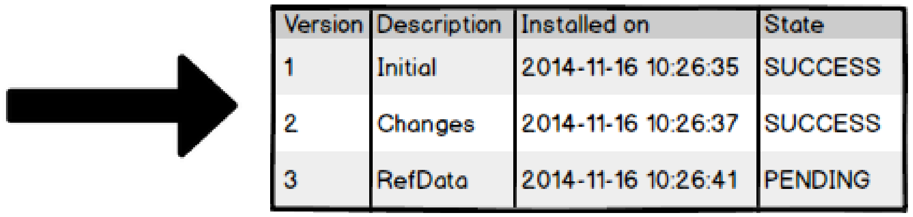
Java Callbacks

```java
public interface FlywayCallback {
	/**
	 * Runs before the clean task executes.
	 *
	 * @param connection A valid connection to the database.
	 */
	void beforeClean(Connection connection);

	/**
	 * Runs after the clean task executes.
	 *
	 * @param connection A valid connection to the database.
	 */
	void afterClean(Connection connection);

	/**
	 * Runs before the migrate task executes.
	 *
	 * @param connection A valid connection to the database.
	 */
	void beforeMigrate(Connection connection);
```

More Flyway Commands

info



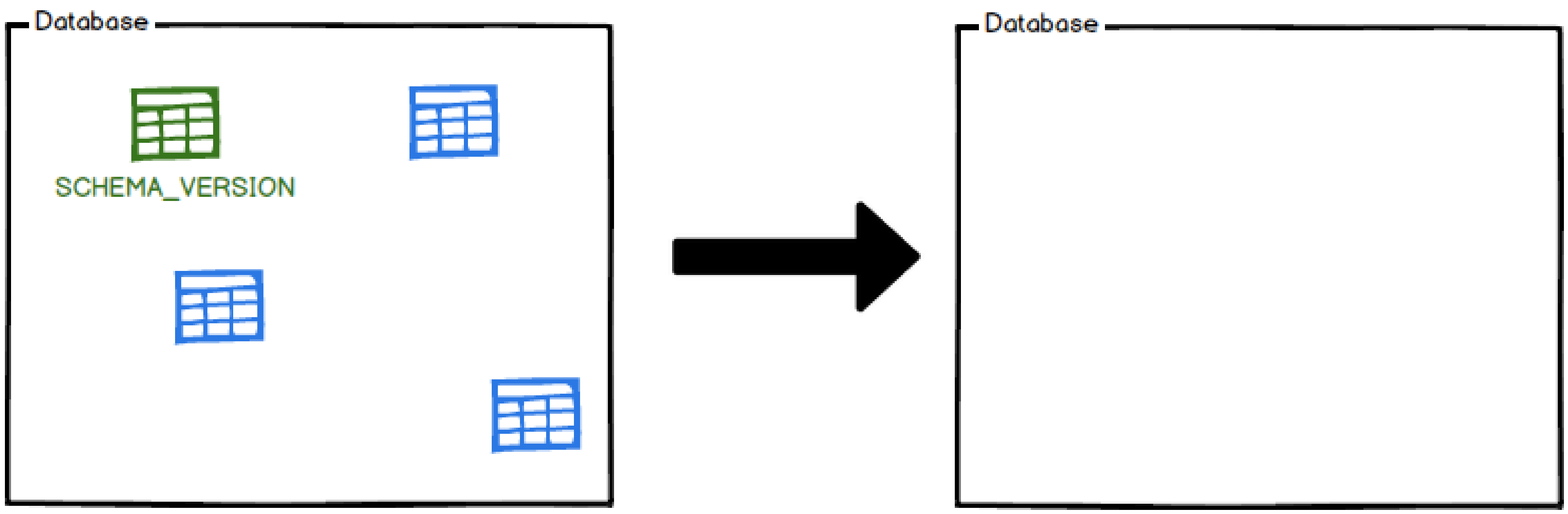


Version	Description	Installed on	State
1	Initial	2014-11-16 10:26:35	SUCCESS
2	Changes	2014-11-16 10:26:37	SUCCESS
3	RefData	2014-11-16 10:26:41	PENDING

Reference: flywaydb.org

More Flyway Commands

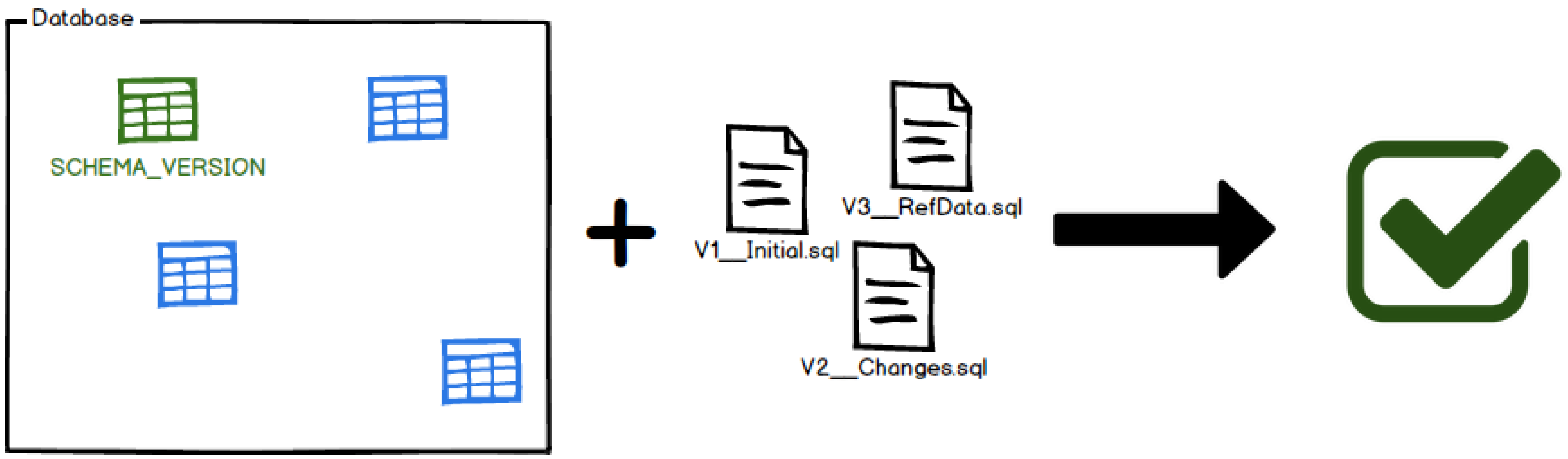
clean



Reference: flywaydb.org

More Flyway Commands

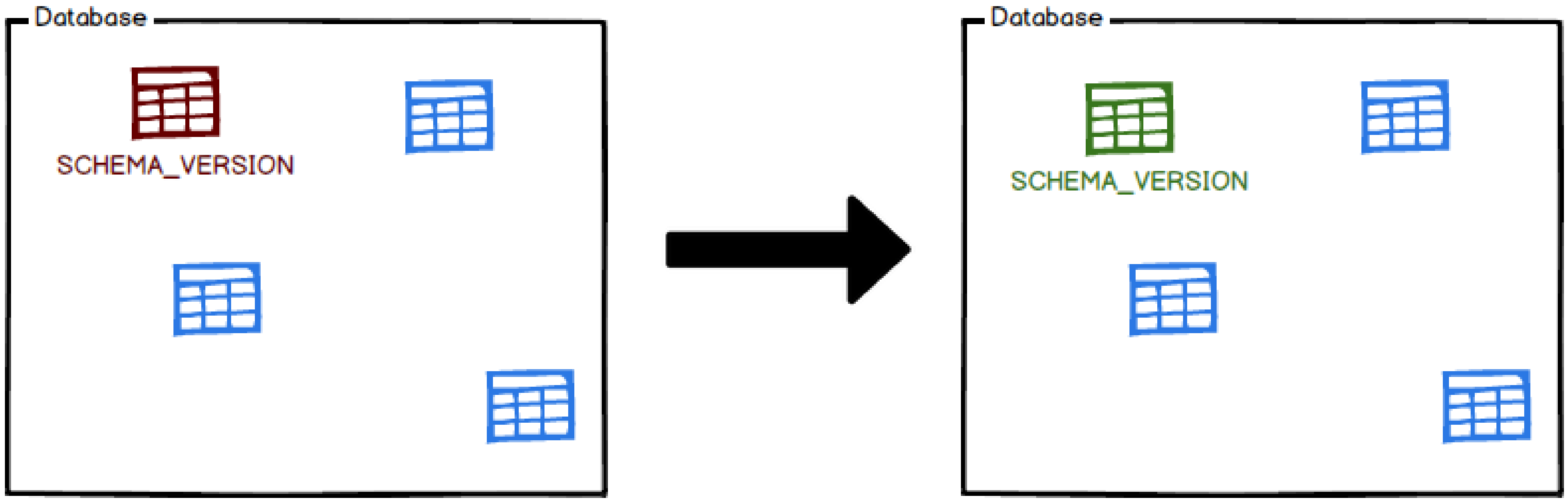
validate



Reference: flywaydb.org

More Flyway Commands

repair



Reference: flywaydb.org

What is not possible with Flyway?

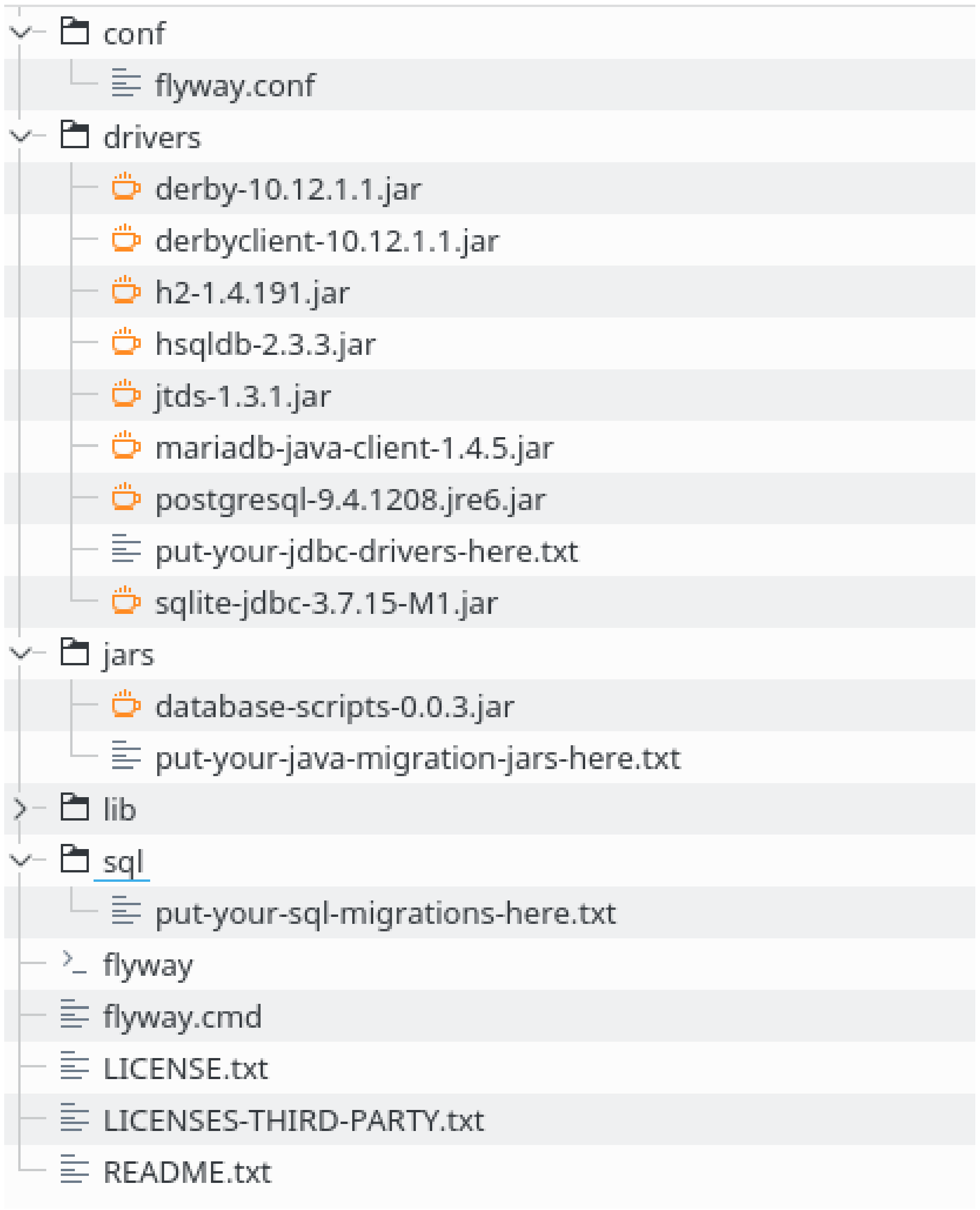
- Rollback scripts execution
- „Write once, run on many database vendors“

How to use Flyway?

- **Flyway Clients:**
  - Java API
  - Maven Plugin
  - Command-line Tool
  - Gradle Plugin
  - SBT Plugin
  - Ant task

- **Third Party Plugins:**
  - Spring Boot
  - Grails
  - Dropwizard
  - Others

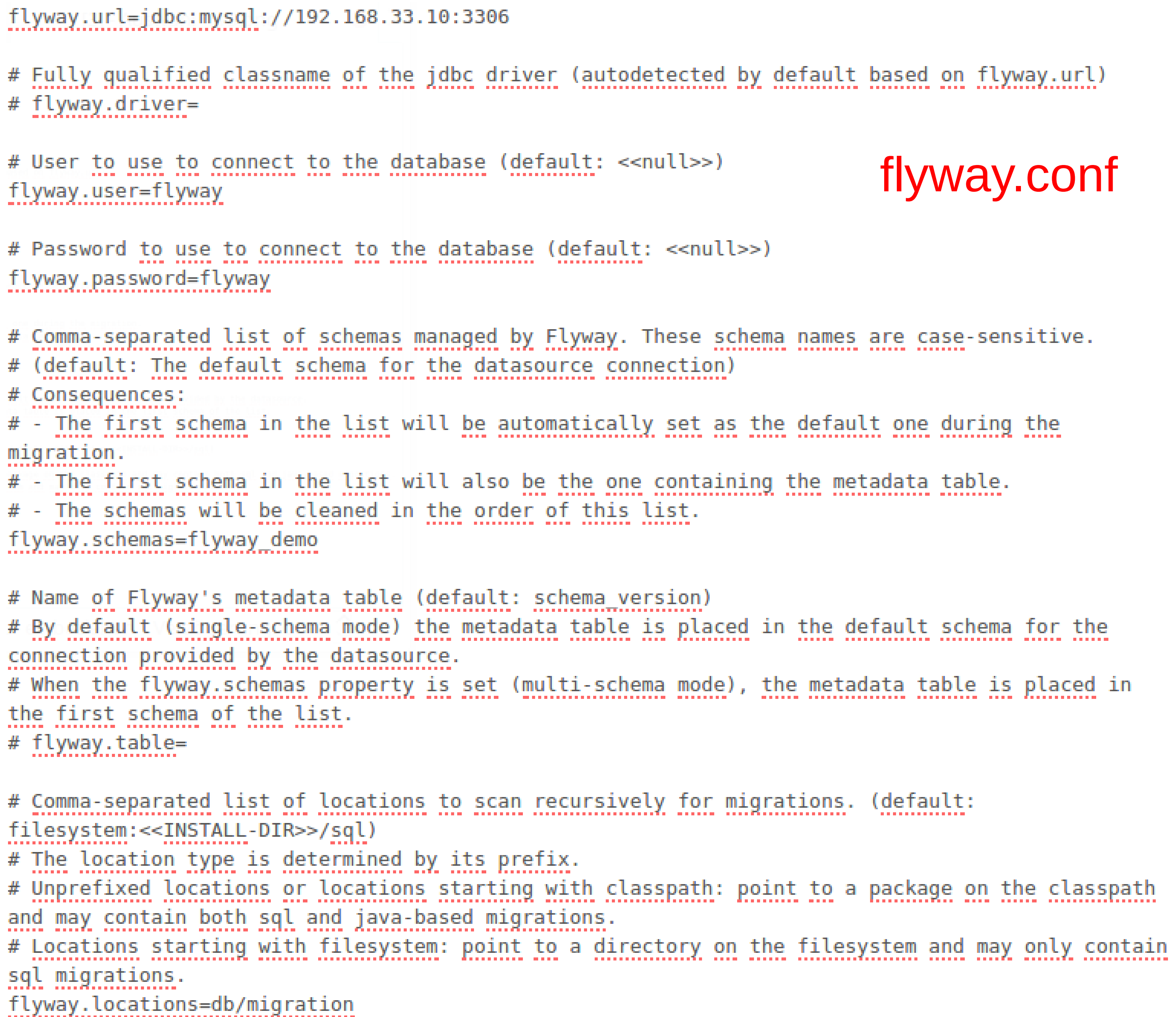
Command-line Tool



```
conf
  flyway.conf
drivers
  derby-10.12.1.1.jar
  derbyclient-10.12.1.1.jar
  h2-1.4.191.jar
  hsqldb-2.3.3.jar
  jtds-1.3.1.jar
  mariadb-java-client-1.4.5.jar
  postgresql-9.4.1208.jre6.jar
  put-your-jdbc-drivers-here.txt
  sqlite-jdbc-3.7.15-M1.jar
jars
  database-scripts-0.0.3.jar
  put-your-java-migration-jars-here.txt
lib
sql
  put-your-sql-migrations-here.txt
flyway
flyway.cmd
LICENSE.txt
LICENSES-THIRD-PARTY.txt
README.txt
```

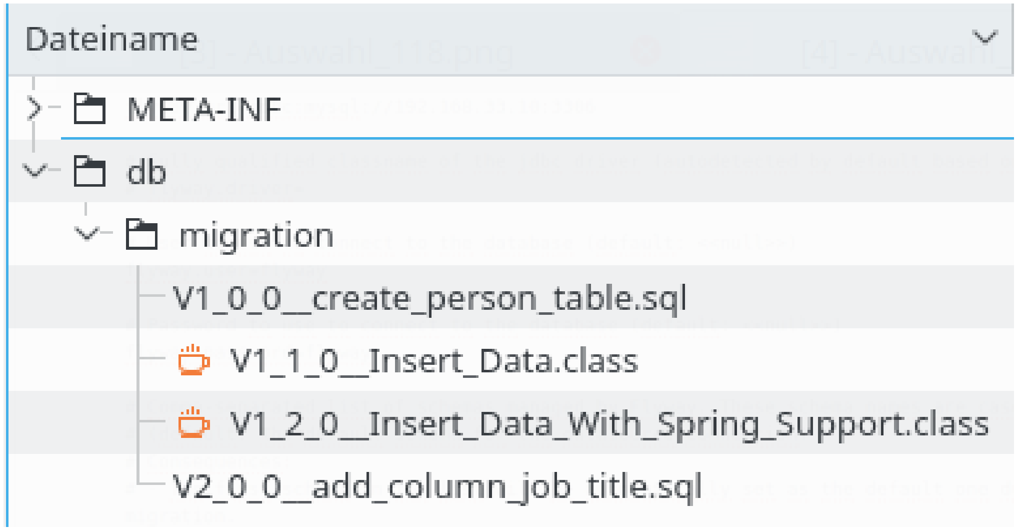
Command-line Tool

flyway.conf



```
flyway.url=jdbc:mysql://192.168.33.10:3306

# Fully qualified classname of the jdbc driver (autodetected by default based on flyway.url)
# flyway.driver=

# User to use to connect to the database (default: <<null>>)
flyway.user=flyway

# Password to use to connect to the database (default: <<null>>)
flyway.password=flyway

# Comma-separated list of schemas managed by Flyway. These schema names are case-sensitive.
# (default: The default schema for the datasource connection)
# Consequences:
# - The first schema in the list will be automatically set as the default one during the
migration.
# - The first schema in the list will also be the one containing the metadata table.
# - The schemas will be cleaned in the order of this list.
flyway.schemas=flyway_demo

# Name of Flyway's metadata table (default: schema_version)
# By default (single-schema mode) the metadata table is placed in the default schema for the
connection provided by the datasource.
# When the flyway.schemas property is set (multi-schema mode), the metadata table is placed in
the first schema of the list.
# flyway.table=

# Comma-separated list of locations to scan recursively for migrations. (default:
filesystem:<<INSTALL-DIR>>/sql)
# The location type is determined by its prefix.
# Unprefixed locations or locations starting with classpath: point to a package on the classpath
and may contain both sql and java-based migrations.
# Locations starting with filesystem: point to a directory on the filesystem and may only contain
sql migrations.
flyway.locations=db/migration
```

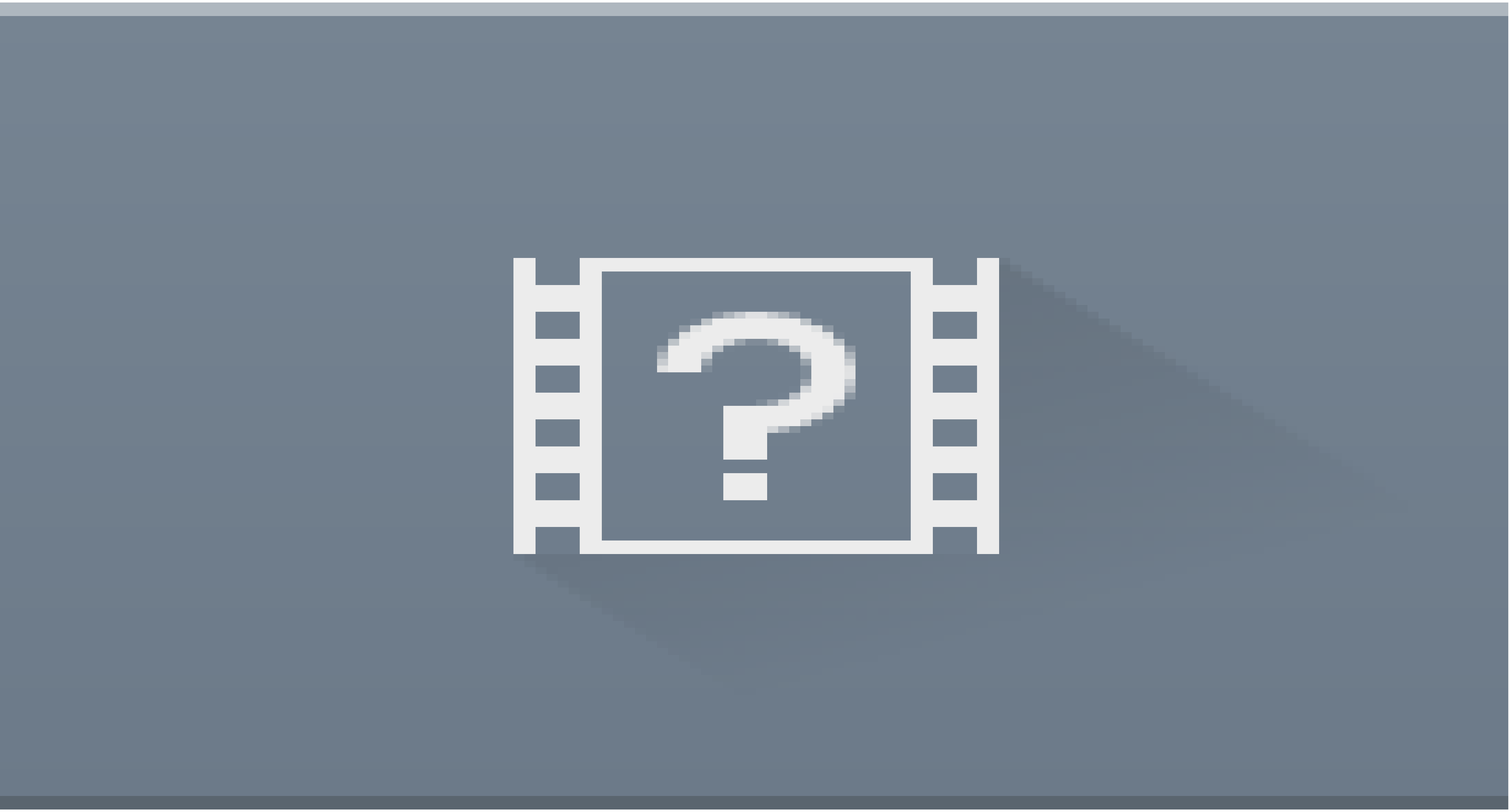
Command-line Tool



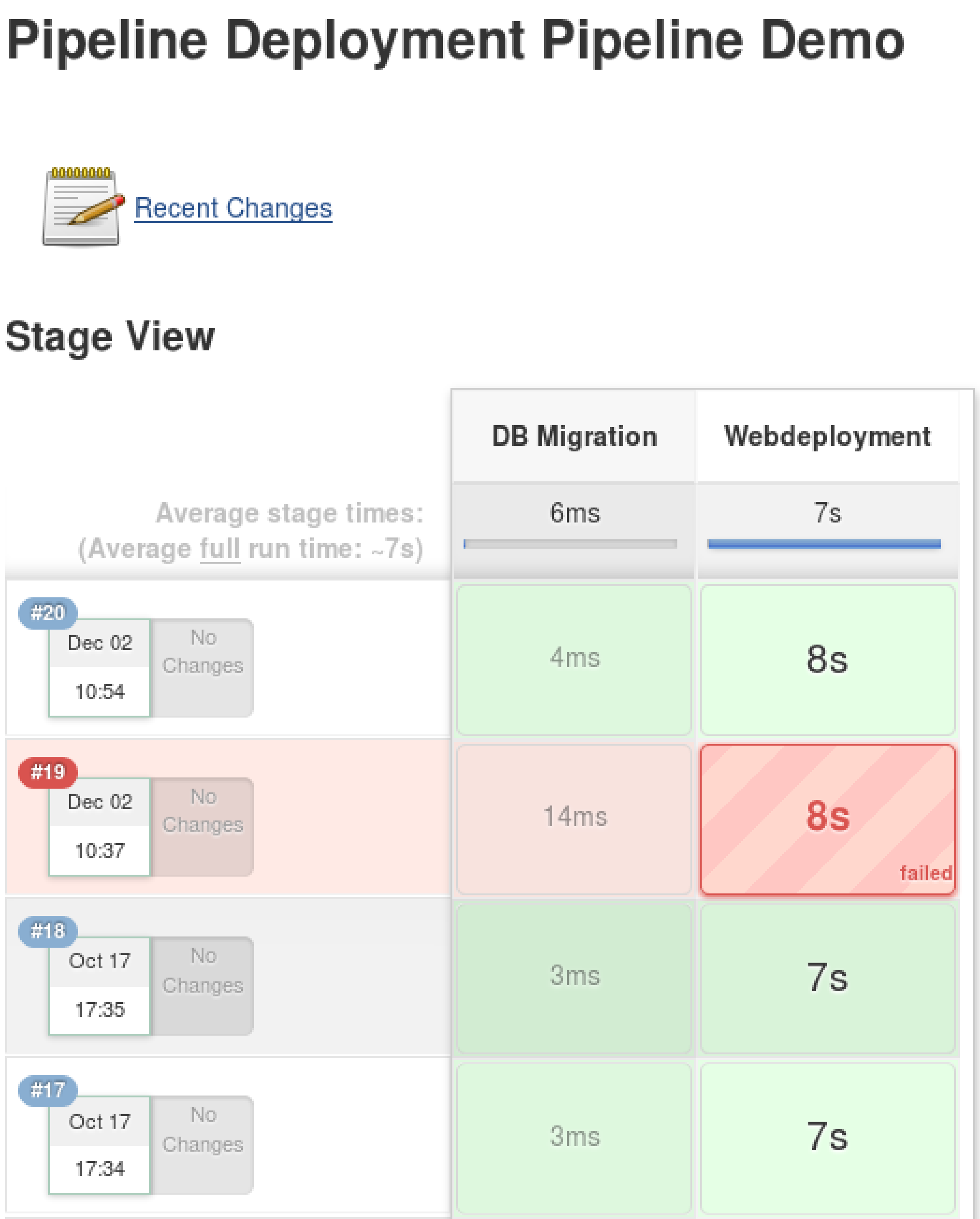
```
Dateiname
> META-INF
v db
  v migration
      V1_0_0__create_person_table.sql
      V1_1_0__Insert_Data.class
      V1_2_0__Insert_Data_With_Spring_Support.class
      V2_0_0__add_column_job_title.sql
```

database-scripts-0.0.3.jar

Command-line Tool



Command-line Tool



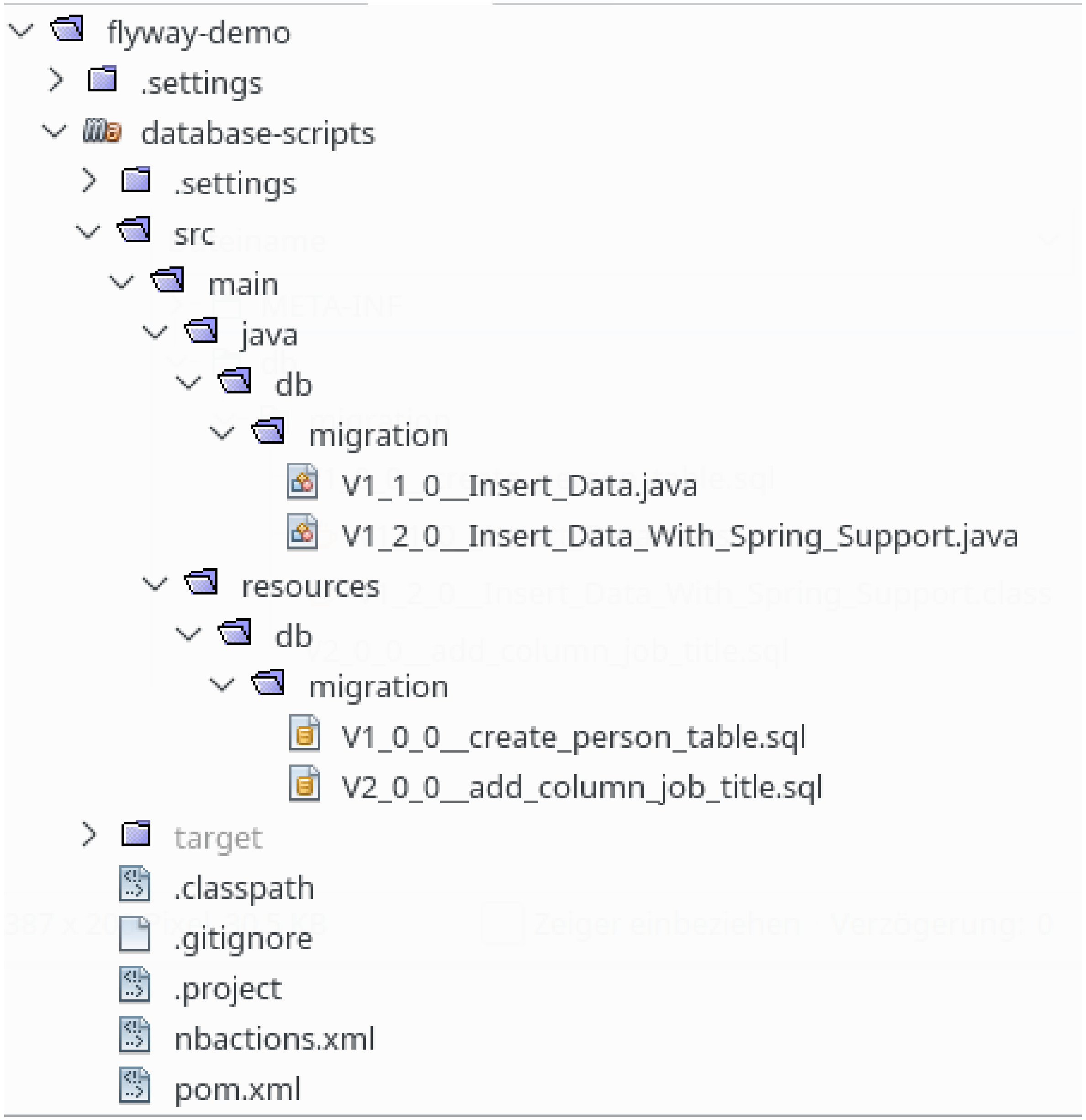
Pipeline Deployment Pipeline Demo

Recent Changes

Stage View

	DB Migration	Webdeployment
Average stage times: (Average full run time: ~7s)	6ms	7s
#20 Dec 02 10:54 No Changes	4ms	8s
#19 Dec 02 10:37 No Changes	14ms	8s failed
#18 Oct 17 17:35 No Changes	3ms	7s
#17 Oct 17 17:34 No Changes	3ms	7s

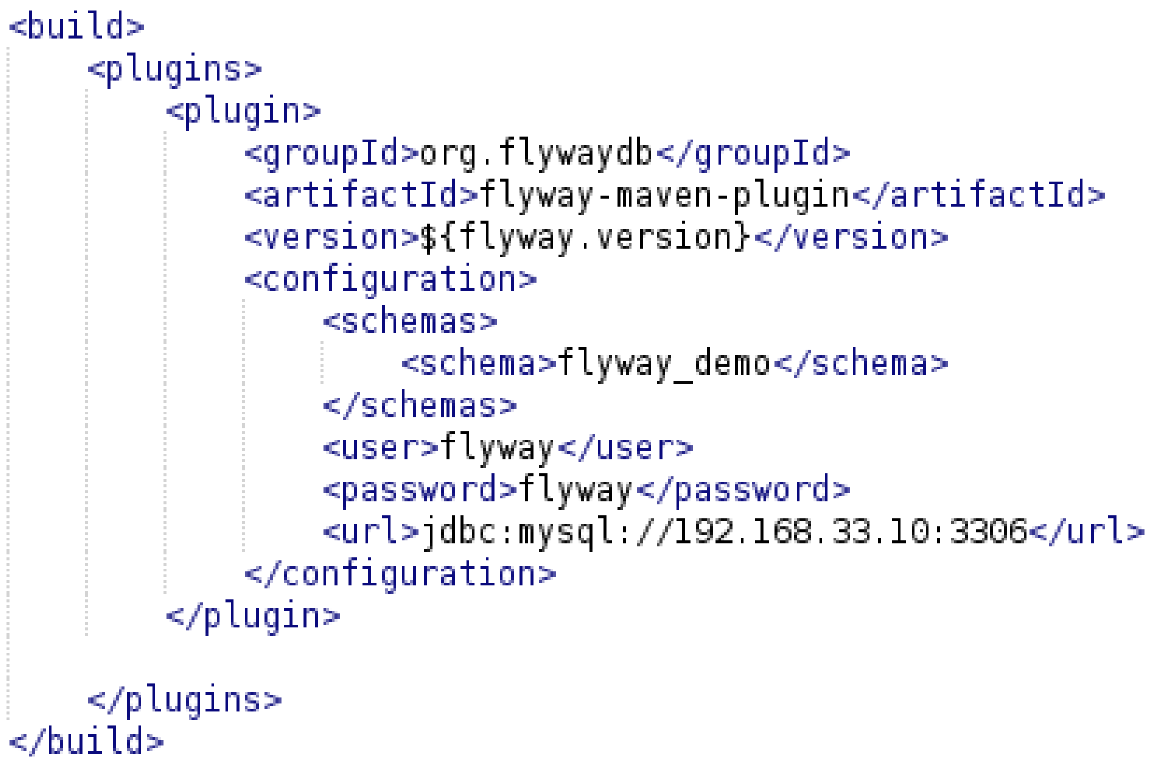
Maven Plugin



```
flyway-demo
  .settings
  database-scripts
    .settings
    src
      main
        java
          db
            migration
              V1_1_0__Insert_Data.java
              V1_2_0__Insert_Data_With_Spring_Support.java
        resources
          db
            migration
              V1_0_0__create_person_table.sql
              V2_0_0__add_column_job_title.sql
    target
    .classpath
    .gitignore
    .project
    nbactions.xml
    pom.xml
```

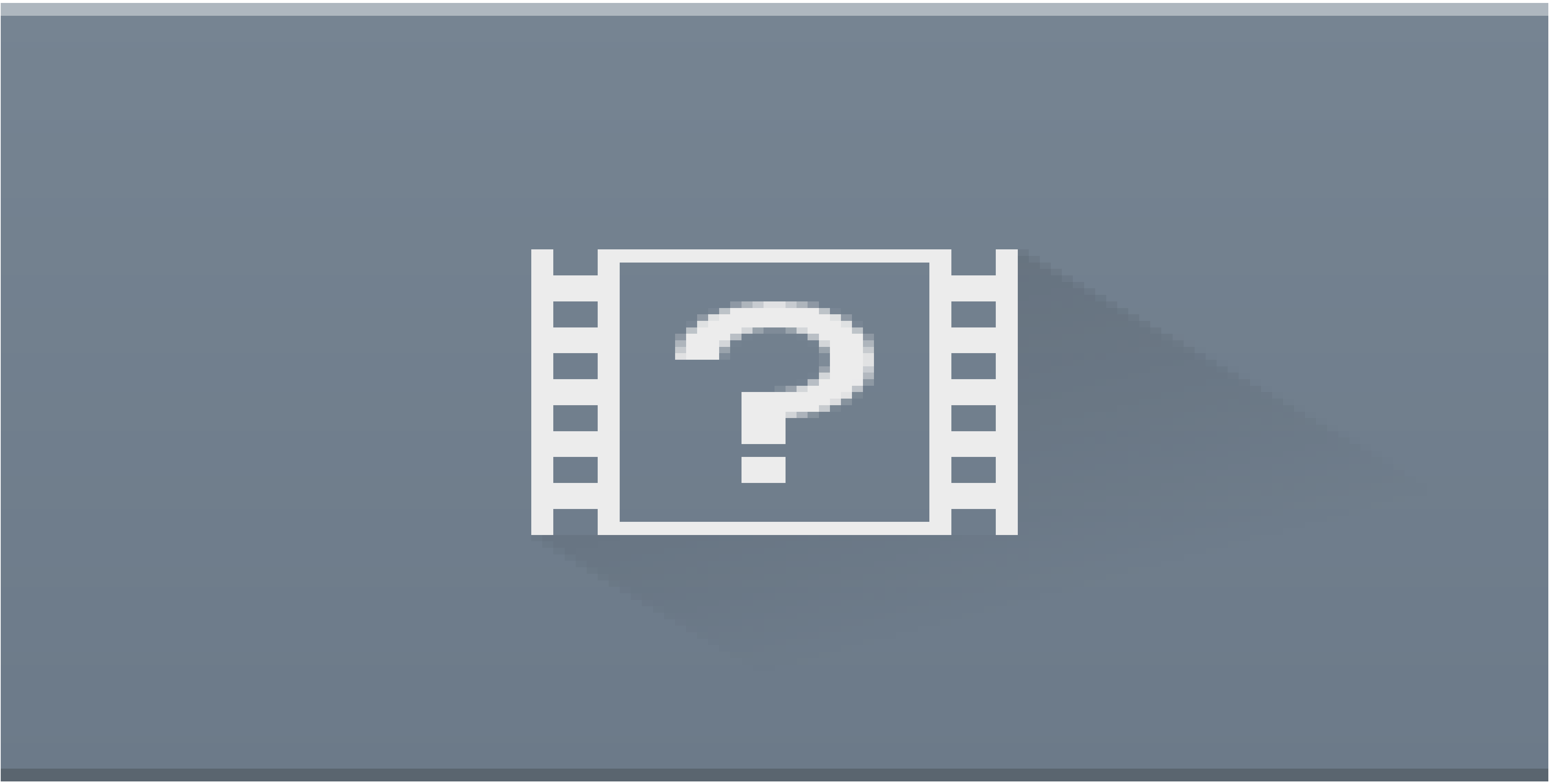
Maven Plugin

pom.xml

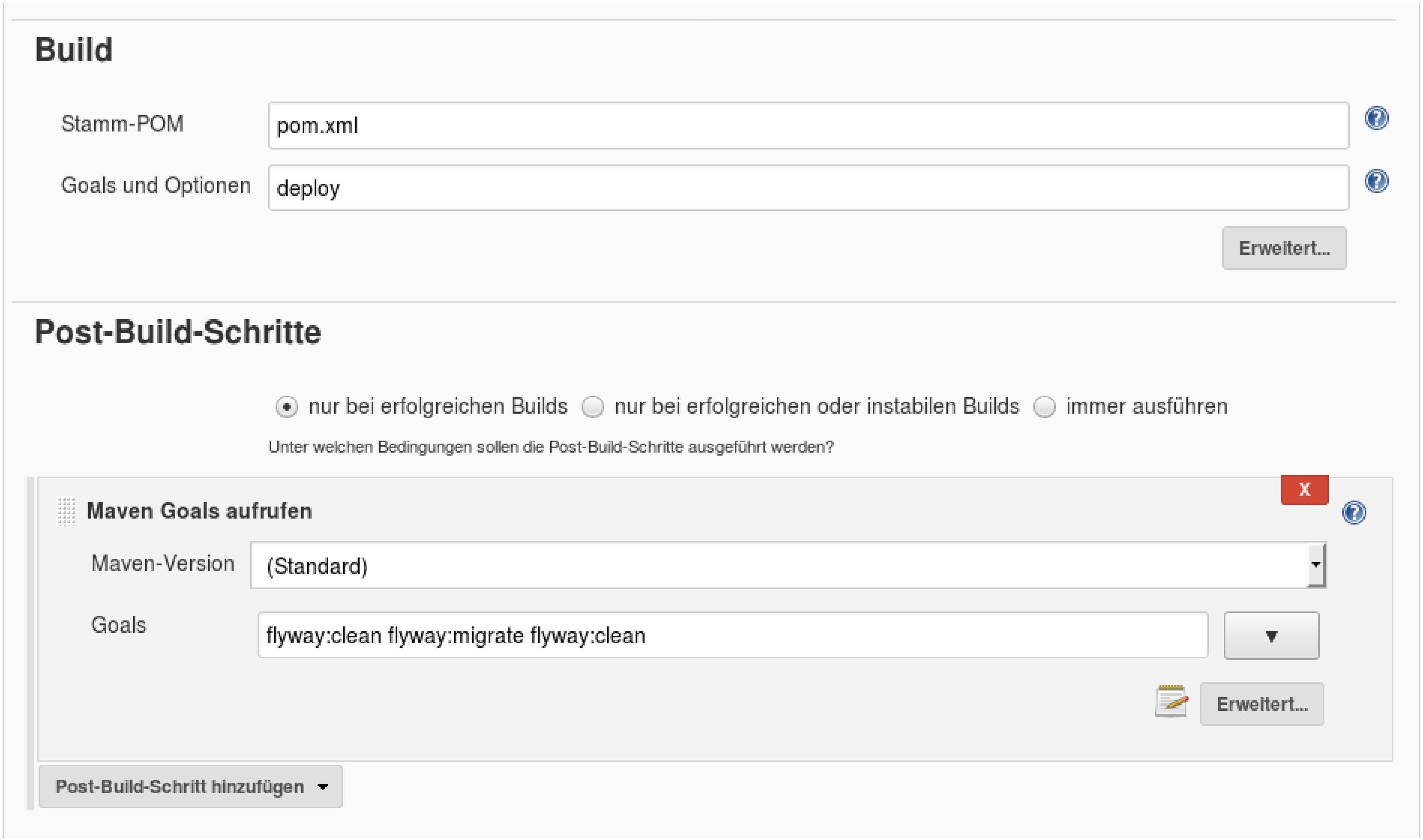


```xml
<build>
    <plugins>
        <plugin>
            <groupId>org.flywaydb</groupId>
            <artifactId>flyway-maven-plugin</artifactId>
            <version>${flyway.version}</version>
            <configuration>
                <schemas>
                    <schema>flyway_demo</schema>
                </schemas>
                <user>flyway</user>
                <password>flyway</password>
                <url>jdbc:mysql://192.168.33.10:3306</url>
            </configuration>
        </plugin>

    </plugins>
</build>
```

Maven Plugin



Maven Plugin



Build

Stamm-POM: pom.xml

Goals und Optionen: deploy

Post-Build-Schritte

[✓ nur bei erfolgreichen Builds] nur bei erfolgreichen oder instabilen Builds immer ausführen

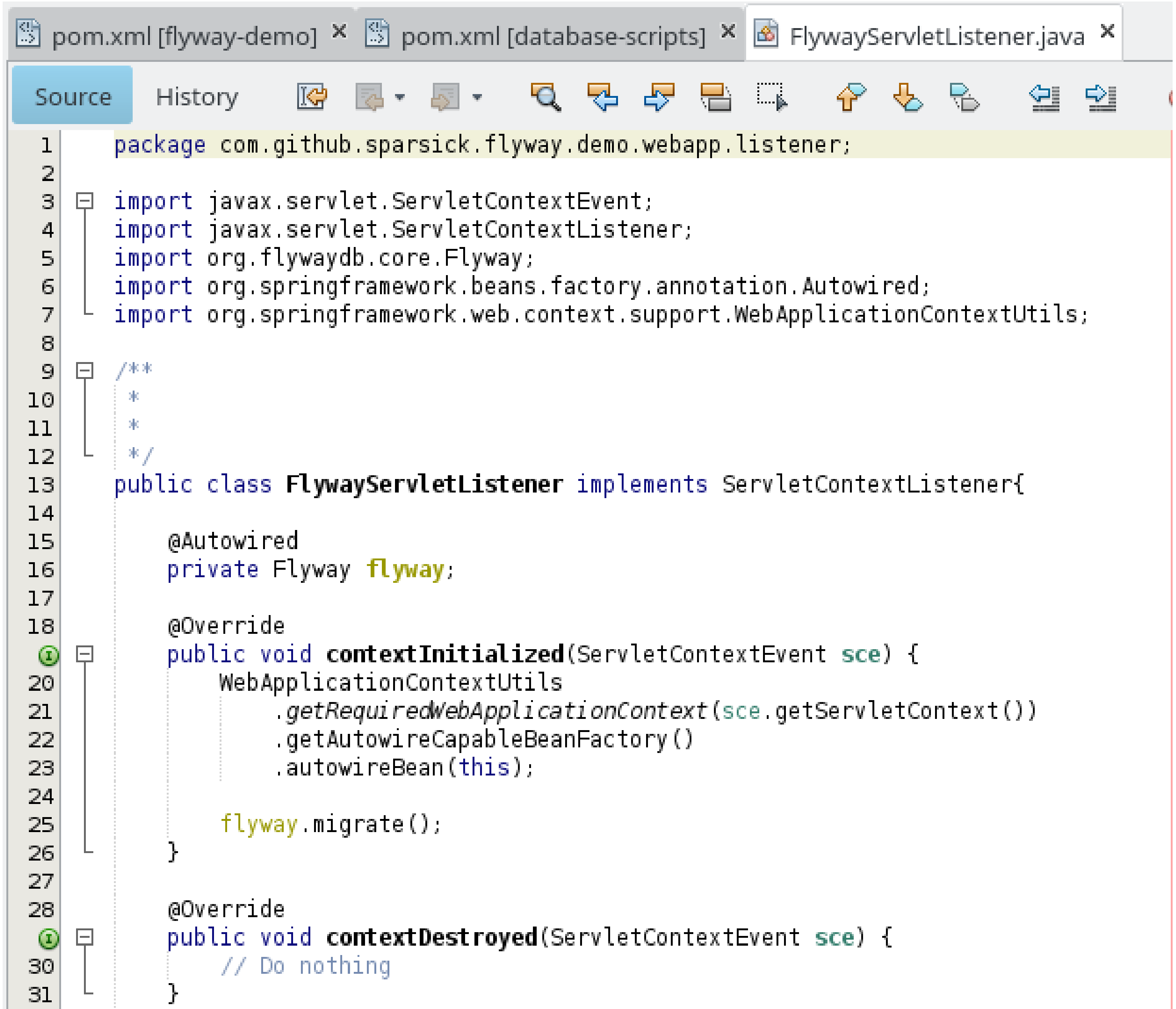
Unter welchen Bedingungen sollen die Post-Build-Schritte ausgeführt werden?

Maven Goals aufrufen

Maven-Version: (Standard)

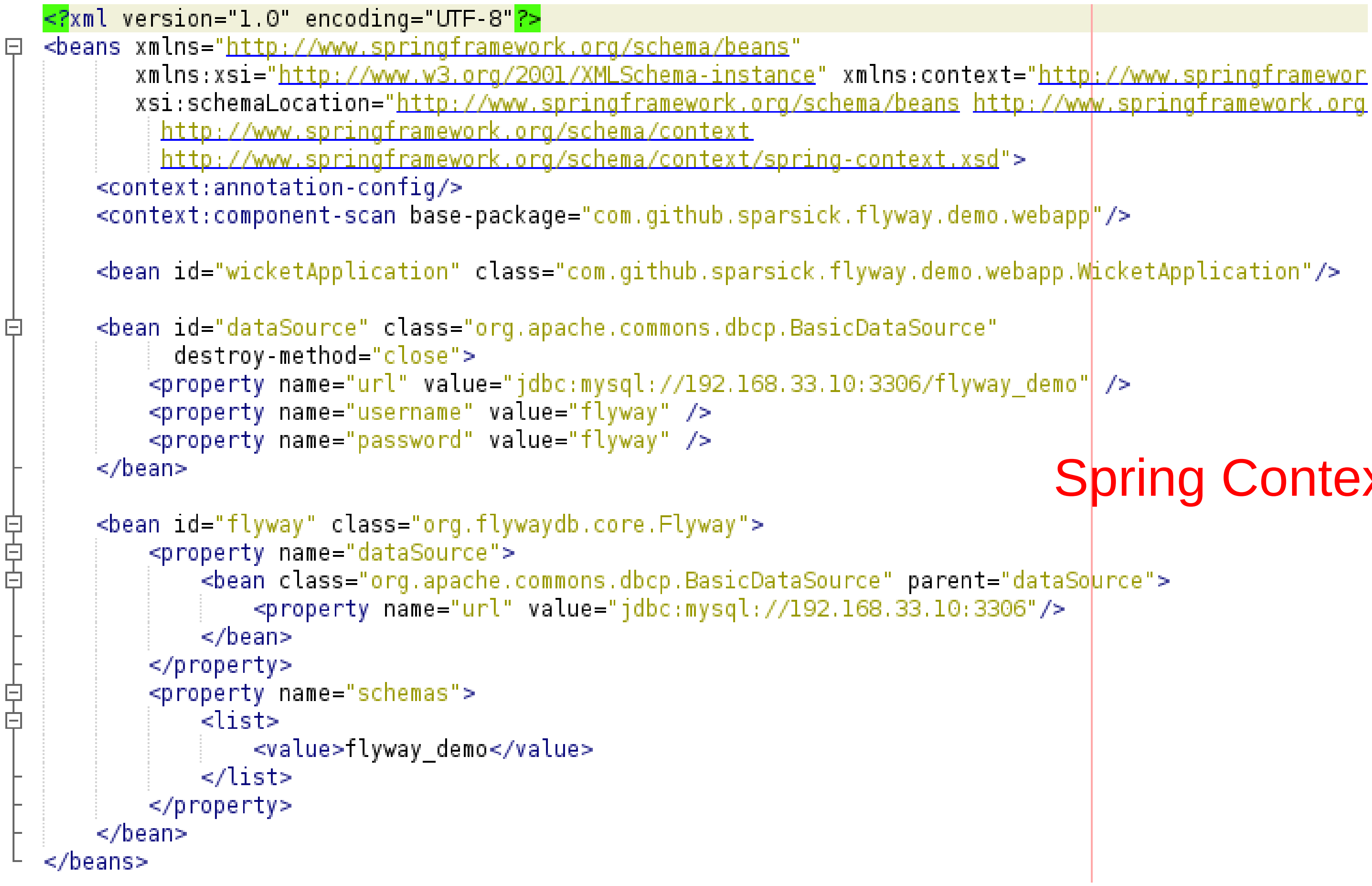
Goals: flyway:clean flyway:migrate flyway:clean

Java API



```java
package com.github.sparsick.flyway.demo.webapp.listener;

import javax.servlet.ServletContextEvent;
import javax.servlet.ServletContextListener;
import org.flywaydb.core.Flyway;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.context.support.WebApplicationContextUtils;

/**
 *
 *
 */
public class FlywayServletListener implements ServletContextListener{

    @Autowired
    private Flyway flyway;

    @Override
    public void contextInitialized(ServletContextEvent sce) {
        WebApplicationContextUtils
            .getRequiredWebApplicationContext(sce.getServletContext())
            .getAutowireCapableBeanFactory()
            .autowireBean(this);

        flyway.migrate();
    }

    @Override
    public void contextDestroyed(ServletContextEvent sce) {
        // Do nothing
    }
```

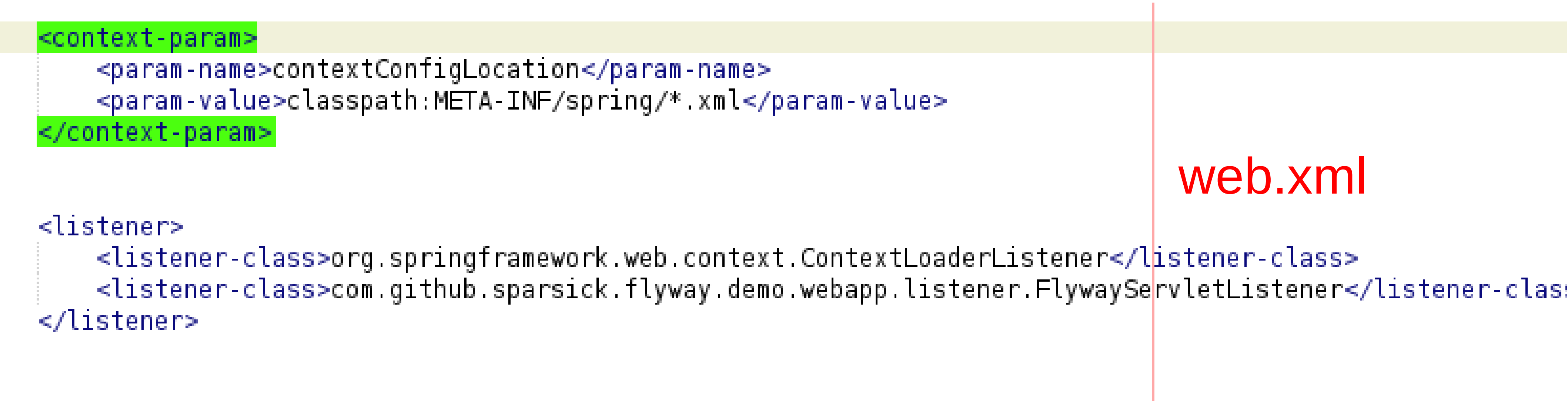
Java API



```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:context="http://www.springframewor
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org
         http://www.springframework.org/schema/context
         http://www.springframework.org/schema/context/spring-context.xsd">
    <context:annotation-config/>
    <context:component-scan base-package="com.github.sparsick.flyway.demo.webapp"/>

    <bean id="wicketApplication" class="com.github.sparsick.flyway.demo.webapp.WicketApplication"/>

    <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
          destroy-method="close">
        <property name="url" value="jdbc:mysql://192.168.33.10:3306/flyway_demo" />
        <property name="username" value="flyway" />
        <property name="password" value="flyway" />
    </bean>

    <bean id="flyway" class="org.flywaydb.core.Flyway">
        <property name="dataSource">
            <bean class="org.apache.commons.dbcp.BasicDataSource" parent="dataSource">
                <property name="url" value="jdbc:mysql://192.168.33.10:3306"/>
            </bean>
        </property>
        <property name="schemas">
            <list>
                <value>flyway_demo</value>
            </list>
        </property>
    </bean>
</beans>
```

Spring Context

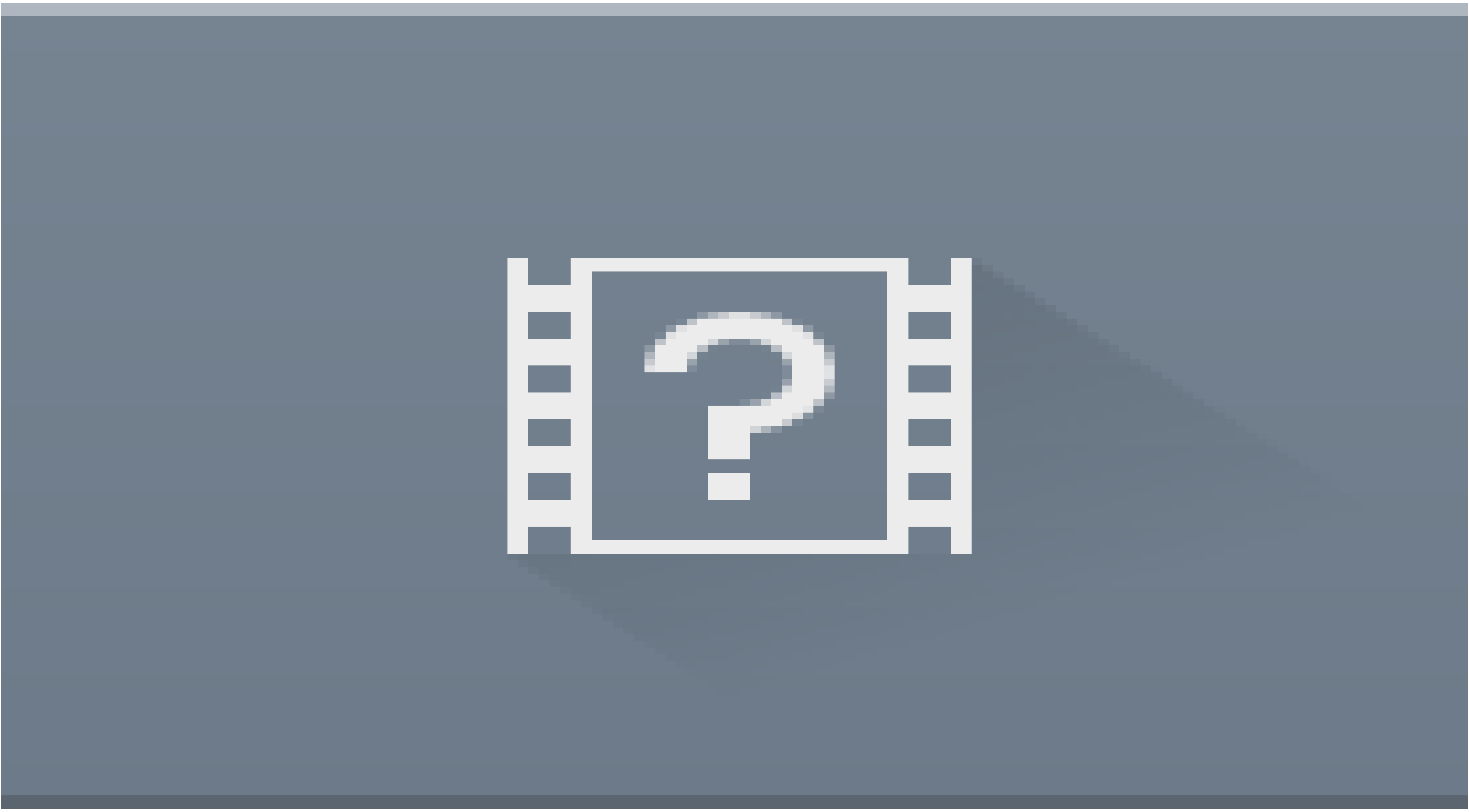
Java API



```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:META-INF/spring/*.xml</param-value>
</context-param>


<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
    <listener-class>com.github.sparsick.flyway.demo.webapp.listener.FlywayServletListener</listener-clas
</listener>
```

web.xml

Java API



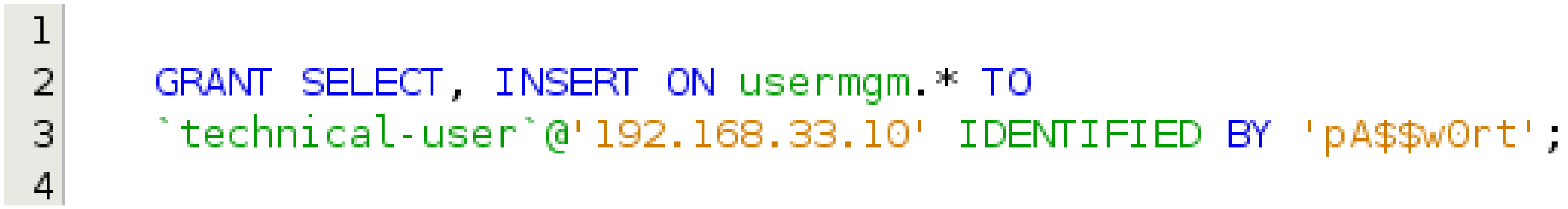
How To Set Up

- Treat database code like a normal source code
  - Put all database assets (DDL, DML, configurations, test data, stored procedures, functions etc) in your version control system. ✓
  - Test your database code after every change. ✓
- Give every developer his own database / Make test database being similar to the productiv database.
  - Set up the database by build scripts. ✓
- All database changes are transparent
  - Change history ✓

Pitfalls

No Instance-Specific Data

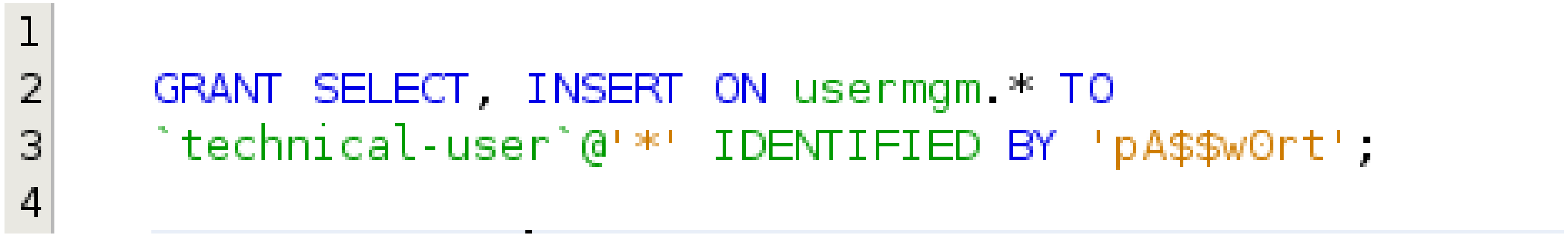
Example



```sql

GRANT SELECT, INSERT ON usermgm.* TO
`technical-user`@'192.168.33.10' IDENTIFIED BY 'pA$$w0rt';

```

No Instance-Specific Data

Possible Solution:

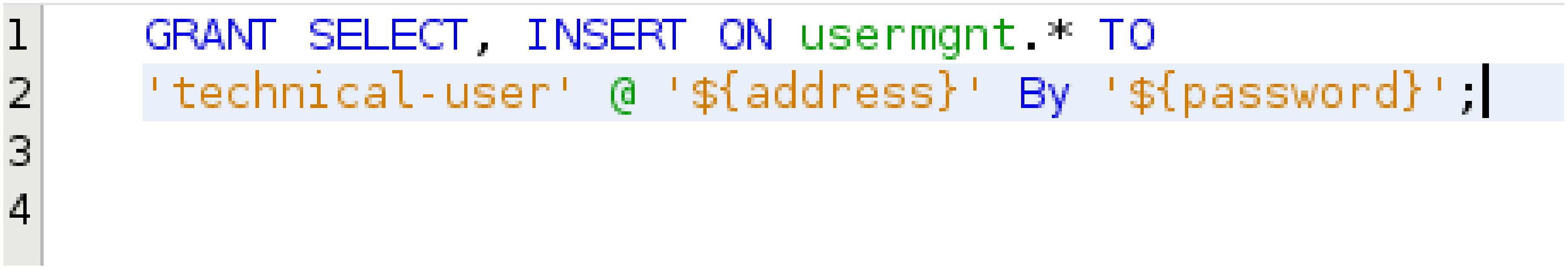


```
GRANT SELECT, INSERT ON usermgm.* TO
`technical-user`@'*' IDENTIFIED BY 'pA$$w0rt';
```

- Access control via firewall (iptables)

No Instance-Specific Data

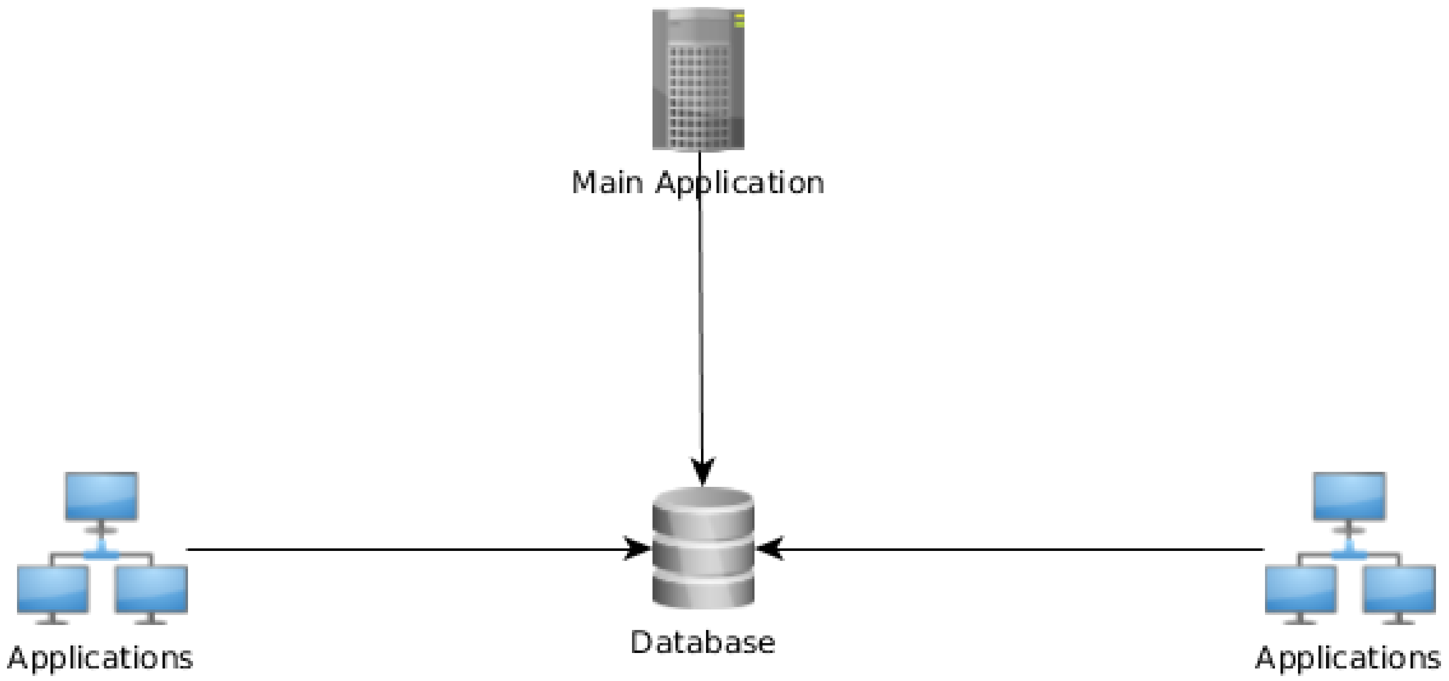
Possible Solution



```
      GRANT SELECT, INSERT ON usermgnt.* TO
      'technical-user' @ '${address}' By '${password}';
```

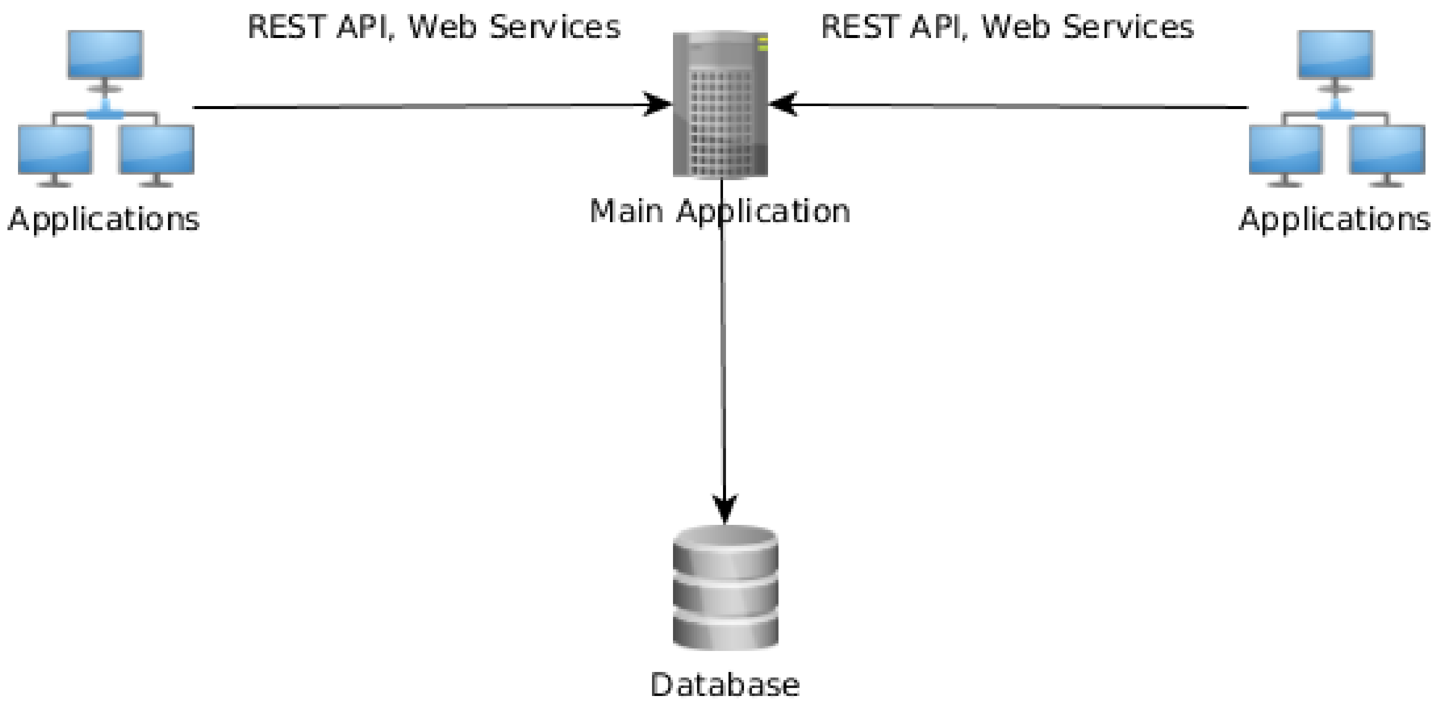
Database is used by several applications

Starting position:



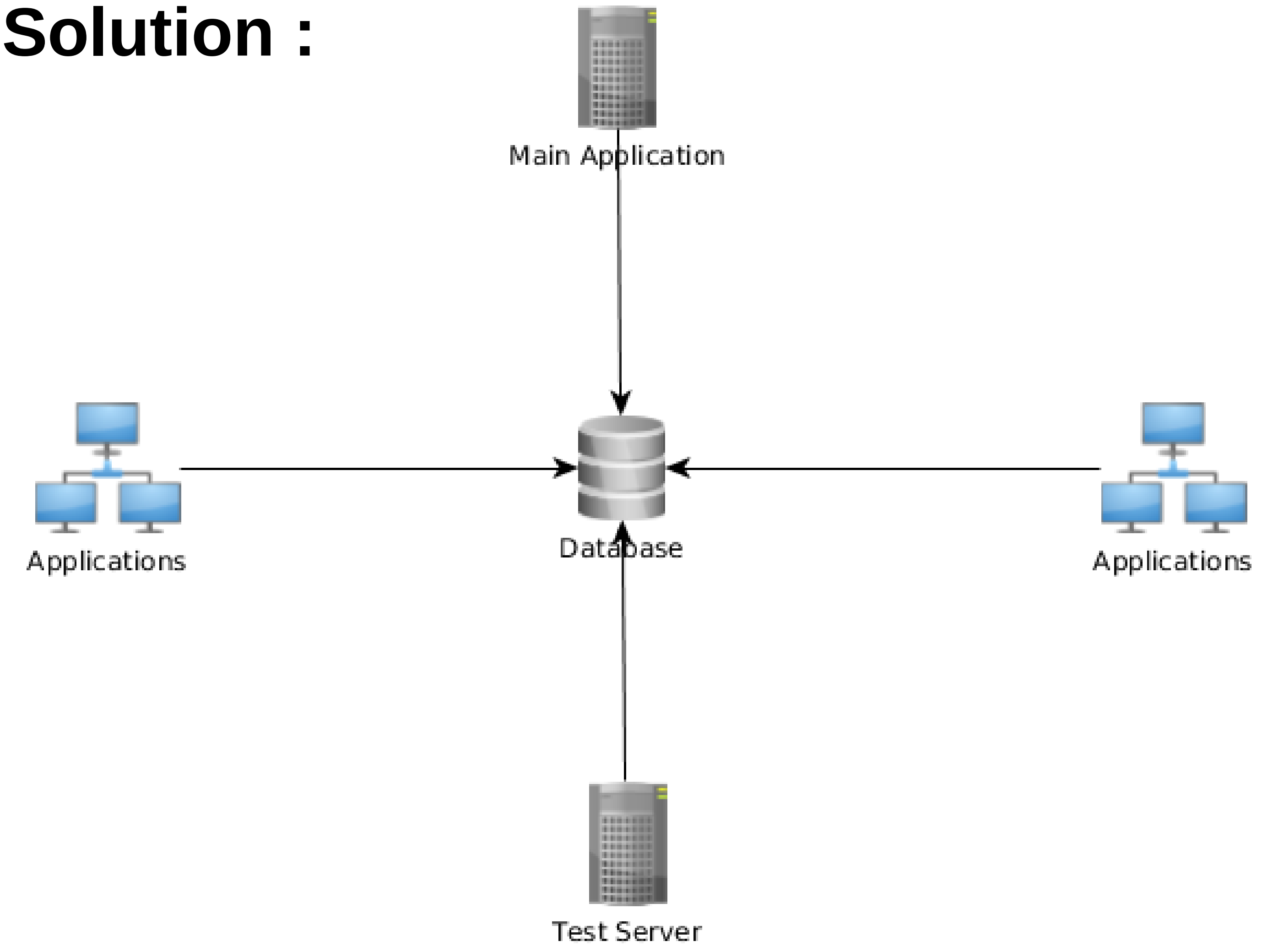
Database is used by several applications

Possible Solution

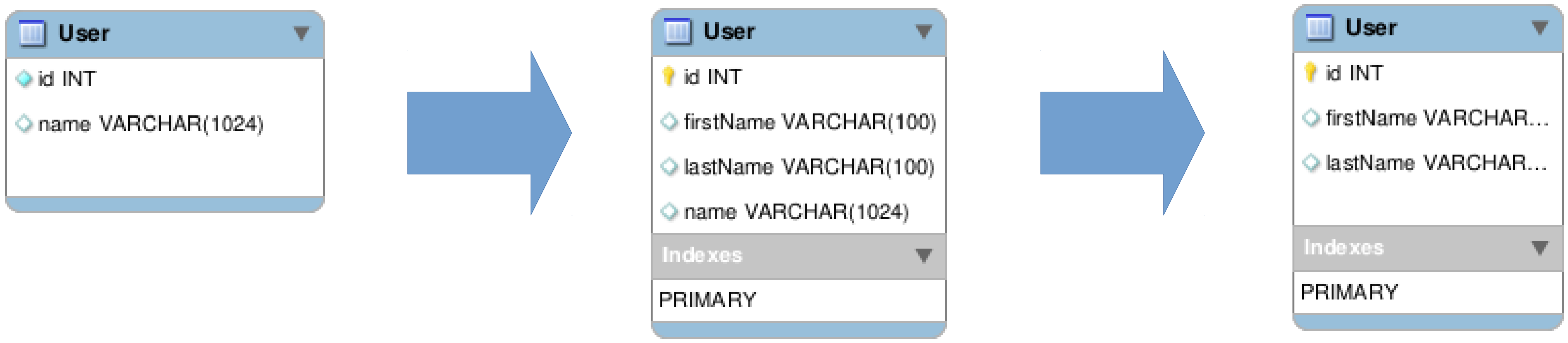


Database is used by several applications

Possible Solution :



Database is used by several applications



More Pitfalls (Short Extract)

- Data changes take too long time
- Data deletion
- Human factor
- ...

Further Information

- Continuous Integration von Paul M. Duvall, Steve Matyas und Andrew Glover
- Refactoring Databases: Evolutionary Database Design von Scott J. Ambler und Pramodkumar J. Sadalage
- Flyway Documentation
  http://flywaydb.org/documentation/migration/
  http://flywaydb.org/getstarted/
- Source code:
  https://github.com/sparsick/flyway-talk

Question?

mail@sandra-parsick.de
@SandraParsick