Database Versioning / Migration

Flyway und Liquibase

Michael Inden

Freiberuflicher Consultant und Trainer

Speaker Intro

- **Michael Inden, Jahrgang 1971**
- **Diplom-Informatiker, C.v.O. Uni Oldenburg**

- **~8 ¼ Jahre SSE bei Heidelberger Druckmaschinen AG in Kiel**
- **~6 ¾ Jahre TPL, SA bei IVU Traffic Technologies AG in Aachen**
- **~4 ¼ Jahre LSA / Trainer bei Zühlke Engineering AG in Zürich**
- **~3 Jahre TL / CTO bei Direct Mail Informatics / ASMIQ in Zürich**
- **Freiberuflicher Consultant, Trainer und Konferenz-Speaker**
- **Autor und Gutachter beim dpunkt.verlag**

E-Mail: michael.inden@hotmail.ch
Blog: https://jaxenter.de/author/minden
https://www.wearedevelopers.com/magazine/java-records

Kurse: Bitte sprecht mich an!

Agenda

Workshop Contents

- **PART 1: Einführung**

- **PART 2: Flyway**

- **PART 3: Liquibase**

PART 1:
Einführung

Typische Abläufe während der Entwicklung

1. Ständiges **Refactoring** zur **Verbesserung** der **Codequalität** (in der Regel **täglich**: Rename, Extract Local Variable / Method, ...)

2. Wiederholtes **Redesign**: (Größerer) Umbau von Klassen und Packages

- **Refactorings = Semantik erhaltende Transformation und ohne Auswirkungen auf das beobachtbare Verhalten / Datenbank**

- **Redesigns = können auch Änderungen in der Datenbank nötig machen**

Einführung
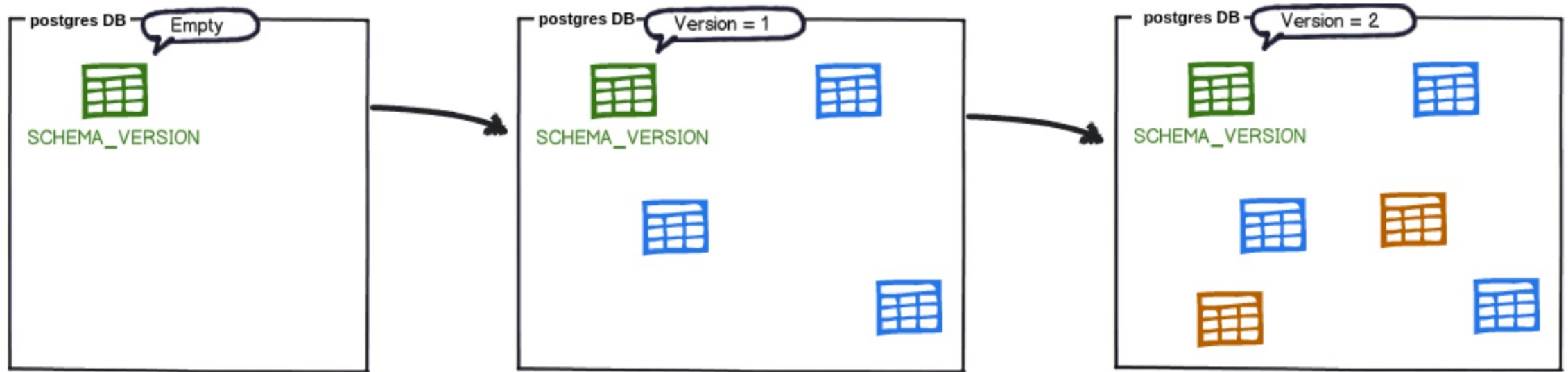
- **Initialer Stand des Modells**

```
class Article {
    int authorId;
    String text;
}
```

```
CREATE TABLE article(
    id INT,
    author_id INT,
    copy VARCHAR(2000)
)
```

- **Neue Anforderungen** oder **Änderungen** erfordern **Datenbankanpassungen**

```
class Article {
    int authorId;
    String copy;
    LocalDateTime
published;
}
```

```
ALTER TABLE article
ADD COLUMN published
TIMESTAMP;
```

- **Neue Anforderungen** oder **Änderungen** erfordern **Datenbankanpassungen**

```java
class Article {
    List<Integer>
authorIds;
    String copy;
    LocalDateTime
published;
}
```

```sql
CREATE TABLE
article_authors(
    article_id INT,
    author_ID INT
);
INSERT INTO
article_authors
SELECT id, author_id FROM
article;
ALTER TABLE article
DROP COLUMN author_id;
```

Einführung

- **Unterschiedliche Softwarestände** benötigen unterschiedliche **Varianten** des **Datenbankmodells**

- **Code und Datenbank müssen konsistent zueinander gehalten werden**

- **Schwierigkeiten** bei Auslieferungen diese **Stände korrekt bereitzustellen** ohne Neuinstallation und bestehende Daten zu verlieren

- **Daher benötigt es eine Möglichkeit zur Datenbankversionierung und Datenbankmigration**

Wie erreicht man das?

- **Lösungen:**

  - **Keine Versionierung** ... ☹ ... Ziemliche Probleme bei Installationen im Feld

  - **Versionierung von Hand**:
    - Inkremente durch einzelne Skripts mit SQL-Anweisungen modellieren
    - Datenbankupdate-Mechanismus selbst bauen:
      – Versionsindikatoren für Skripte
      – Versionstabelle

  - **Versionierung mithilfe von Tools**
    - Flyway
    - Liquibase

PART 2:
Flyway

Integration in Maven Build

```xml
<!-- FLYWAY -->
<plugin>
    <groupId>org.flywaydb</groupId>
    <artifactId>flyway-maven-plugin</artifactId>
    <version>7.9.1</version>
</plugin>
```

Integration in Maven Build

```xml
<properties>
    <!-- Properties are prefixed with flyway. -->
    <flyway.driver>org.hsqldb.jdbcDriver</flyway.driver>
    <flyway.url>jdbc:hsqldb:hsql://localhost:9001/java-profi</flyway.url>
    <flyway.user>sa</flyway.user>
    <flyway.password></flyway.password>

    <flyway.schemas>flywayexample</flyway.schemas>
    <flyway.createSchemas>true</flyway.createSchemas>
</properties>
```

Definition der Änderungen

```
▼ 📂 src/main/resources
   ▼ 🗂 db.migration
        📄 U3.0__create_person.sql
        📄 U3.5__populate_person.sql
        📄 V1.0__create_company.sql
        📄 V2.0__populate_company.sql
        📄 V3.0__create_person.sql
        📄 V3.5__populate_person.sql
        📄 V3.7__update_person_add_columns.sql
        📄 V3.8__update_person_add_birthday.sql
        📄 V4.0__create_address.sql
```

Versioned Migrations

```
Prefix    Separator
  ↓         ↓
 V 2 __ Add_new_table
     ↑         ↑
  Version   Description
```

Undo Migrations

```
Prefix    Separator
  ↓         ↓
 U 2 __ Add_new_table
     ↑         ↑
  Version   Description
```

Repeatable Migrations

```
Prefix Separator
  ↓   ↓
 R __ Add_new_table
         ↑
     Description
```
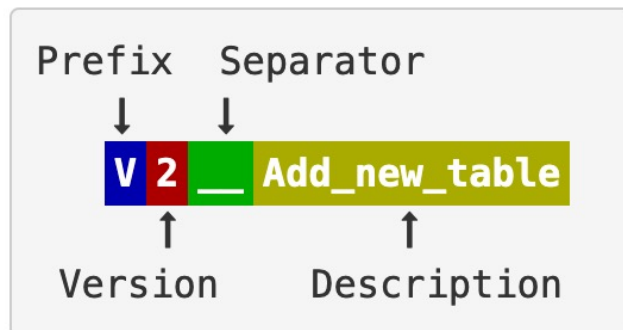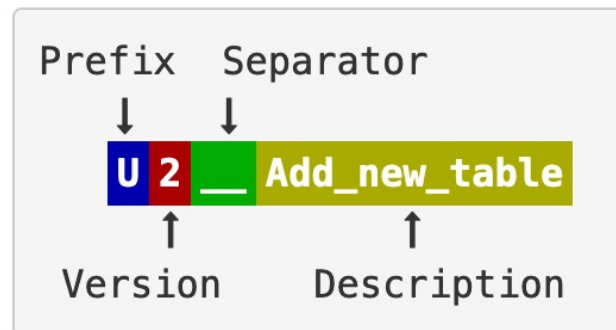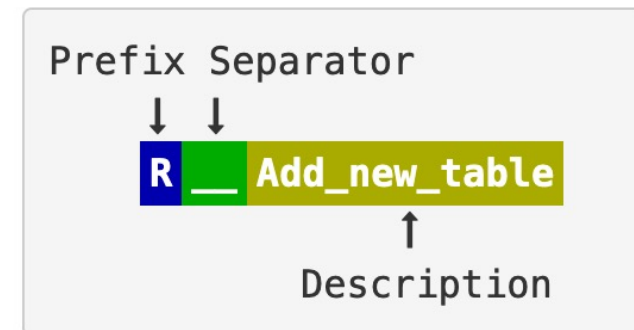
Definition der Änderungen

V1.0__create_company.sql

```sql
create table company
(
    id                            uuid         not null,
    name                          varchar(255) not null,
    web_page_address              varchar(255),
    billing_contact_email_address varchar(255),
    primary_contact_email_address varchar(255),
    constraint pk_company primary key (id)
);
```

Definition der Änderungen

V3.0__create_person.sql

```sql
CREATE TABLE PERSON (
    id int not null primary key,
    first_name varchar(255) not null,
    last_name varchar(255) not null
);
```

V3.5__populate_person.sql

```sql
insert into PERSON (ID, FIRST_NAME, LAST_NAME) values (1, 'Tim', 'Bötzmeyer');
insert into PERSON (ID, FIRST_NAME, LAST_NAME) values (2, 'Tom', 'Jerry');
insert into PERSON (ID, FIRST_NAME, LAST_NAME) values (3, 'Michael', 'Inden');
insert into PERSON (ID, FIRST_NAME, LAST_NAME) values (4, 'Sophie', 'Inden');
```

Definition der Änderungen

V3.7__update_person_add_columns.sql

```sql
ALTER TABLE PERSON
ADD Email varchar(255);


ALTER TABLE PERSON
ADD last_modified_date timestamp;


UPDATE PERSON
SET last_modified_date=current_timestamp;


UPDATE PERSON
SET Email=CONCAT(first_name, '_', last_name, '@specialcompany.ch');
```

Maven Kommandos

```
mvn flyway:migrate
mvn flyway:repair
mvn flyway:info
mvn flyway:clean
```

```
Commands
--------
migrate  : Migrates the database
clean    : Drops all objects in the configured schemas
info     : Prints the information about applied, current and pending migrations
validate : Validates the applied migrations against the ones on the classpath
undo     : [teams] Undoes the most recently applied versioned migration
baseline : Baselines an existing database at the baselineVersion
repair   : Repairs the schema history table
```

Maven Kommandos -- `mvn flyway:migrate`

```
[INFO] --- flyway-maven-plugin:7.9.1:migrate (default-cli) @ LiquiBaseDemo ---
[INFO] Flyway Community Edition 7.9.1 by Redgate
[INFO] Database: jdbc:hsqldb:hsql://localhost:9001/java-profi (HSQL Database Engine 2.6)
[WARNING] Flyway upgrade recommended: HSQLDB 2.6 is newer than this version of Flyway and
support has not been tested. The latest supported version of HSQLDB is 2.5.
[INFO] Creating schema "flywayexample" ...
[INFO] Creating Schema History table "flywayexample"."flyway_schema_history" ...
[INFO] Current version of schema "flywayexample": null
[INFO] Migrating schema "flywayexample" to version "1.0 - create company"
[INFO] Migrating schema "flywayexample" to version "2.0 - populate company"
[INFO] Migrating schema "flywayexample" to version "3.0 - create person"
[INFO] Migrating schema "flywayexample" to version "3.5 - populate person"
[INFO] Migrating schema "flywayexample" to version "3.7 - update person add columns"
[INFO] Migrating schema "flywayexample" to version "3.8 - update person add birthday"
[INFO] Migrating schema "flywayexample" to version "4.0 - create address"
[INFO] Successfully applied 7 migrations to schema "flywayexample", now at version v4.0
(execution time 00:00.106s)
[INFO] Flyway Community Edition 7.9.1 by Redgate
[WARNING] Flyway upgrade recommended: HSQLDB 2.6 is newer than this version of Flyway and
support has not been tested. The latest supported version of HSQLDB is 2.5.
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

Maven Kommandos -- `mvn flyway:info`

```
[INFO] --- flyway-maven-plugin:7.9.1:info (default-cli) @ LiquiBaseDemo ---
[INFO] Flyway Community Edition 7.9.1 by Redgate
[INFO] Database: jdbc:hsqldb:hsql://localhost:9001/java-profi (HSQL Database Engine 2.6)
[WARNING] Flyway upgrade recommended: HSQLDB 2.6 is newer than this version of Flyway and support
has not been tested. The latest supported version of HSQLDB is 2.5.
[INFO] Schema version: 4.0
[INFO]
[INFO]
+-----------+---------+---------------------------+--------+---------------------+---------+
| Category  | Version | Description               | Type   | Installed On        | State   |
+-----------+---------+---------------------------+--------+---------------------+---------+
|           |         | << Flyway Schema Creation >> | SCHEMA | 2021-05-27 13:51:26 | Success |
| Versioned | 1.0     | create company            | SQL    | 2021-05-27 13:51:26 | Success |
| Versioned | 2.0     | populate company          | SQL    | 2021-05-27 13:51:26 | Success |
| Versioned | 3.0     | create person             | SQL    | 2021-05-27 13:51:26 | Success |
| Versioned | 3.5     | populate person           | SQL    | 2021-05-27 13:51:26 | Success |
| Versioned | 3.7     | update person add columns | SQL    | 2021-05-27 13:51:26 | Success |
| Versioned | 3.8     | update person add birthday | SQL    | 2021-05-27 13:51:26 | Success |
| Versioned | 4.0     | create address            | SQL    | 2021-05-27 13:51:26 | Success |
+-----------+---------+---------------------------+--------+---------------------+---------+
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```
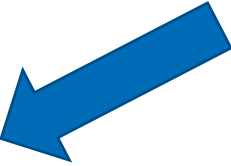
Integration in Maven Build mit Java-Migration

```xml
<!-- FLYWAY -->
<plugin>
    <groupId>org.flywaydb</groupId>
    <artifactId>flyway-maven-plugin</artifactId>
    <version>7.9.1</version>
    <configuration>
        <locations>
            <location>classpath:db/migration</location>
        </locations>
    </configuration>
</plugin>


<dependency>
    <groupId>org.flywaydb</groupId>
    <artifactId>flyway-core</artifactId>
    <version>7.9.1</version>
</dependency>
```

Integration in Maven Build mit Java-Migration

```java
public class V3_6__InsertRandomPersons extends BaseJavaMigration
{
    @Override
    public void migrate(Context context) throws Exception
    {
        Connection connection = context.getConnection();

        try (PreparedStatement statement = connection.prepareStatement(
            "INSERT INTO PERSON (ID, FIRST_NAME, LAST_NAME) VALUES (5, 'Anne', 'Will')"))
        {
            statement.execute();
        }

        for (int i = 1; i < 11; i++)
        {
            try (PreparedStatement statement = connection.prepareStatement(
                String.format("INSERT INTO PERSON (ID, FIRST_NAME, LAST_NAME)" +
                              "VALUES (%d, 'Peter_%d', 'Lustig_%d')", i + 1000, i, i)))
            {
                statement.execute();
            }
        }
    }
}
```

Integration in Maven Build mit Java-Migration

```
+-----------+---------+------------------------------+--------+---------------------+---------+
| Category  | Version | Description                  | Type   | Installed On        | State   |
+-----------+---------+------------------------------+--------+---------------------+---------+
|           |         | << Flyway Schema Creation >> | SCHEMA | 2021-05-27 14:46:24 | Success |
| Versioned | 1.0     | create company               | SQL    | 2021-05-27 14:46:24 | Success |
| Versioned | 2.0     | populate company             | SQL    | 2021-05-27 14:46:24 | Success |
| Versioned | 3.0     | create person                | SQL    | 2021-05-27 14:46:25 | Success |
| Versioned | 3.5     | populate person              | SQL    | 2021-05-27 14:46:25 | Success |
| Versioned | 3.6     | InsertRandomPersons          | JDBC   | 2021-05-27 14:46:25 | Success |
| Versioned | 3.7     | update person add columns    | SQL    | 2021-05-27 14:46:25 | Success |
| Versioned | 3.8     | update person add birthday   | SQL    | 2021-05-27 14:46:25 | Success |
| Versioned | 4.0     | create address               | SQL    | 2021-05-27 14:46:25 | Success |
+-----------+---------+------------------------------+--------+---------------------+---------+
```

DEMO

FlywayLiquiBaseDemo

```xml
<!--
<configuration>
    <locations>
        <location>classpath:db/migration</location>
    </locations>
</configuration>
-->
```

Externe Infos

- **https://flywaydb.org/**

- **https://flywaydb.org/documentation/concepts/migrations.html**

- **https://flywaydb.org/documentation/concepts/migrations#java-based-migrations**

- **https://reflectoring.io/database-migration-spring-boot-flyway/**

- **http://www.mastertheboss.com/other/flyway/getting-started-with-flyway**

Externe Infos

- **https://thorben-janssen.com/flyway-getting-started/**

- **https://thorben-janssen.com/java-based-database-migrations-callbacks-flyway/**

- **https://blog.codecentric.de/en/2017/01/flyway-tutorial-managing-database-migrations/**

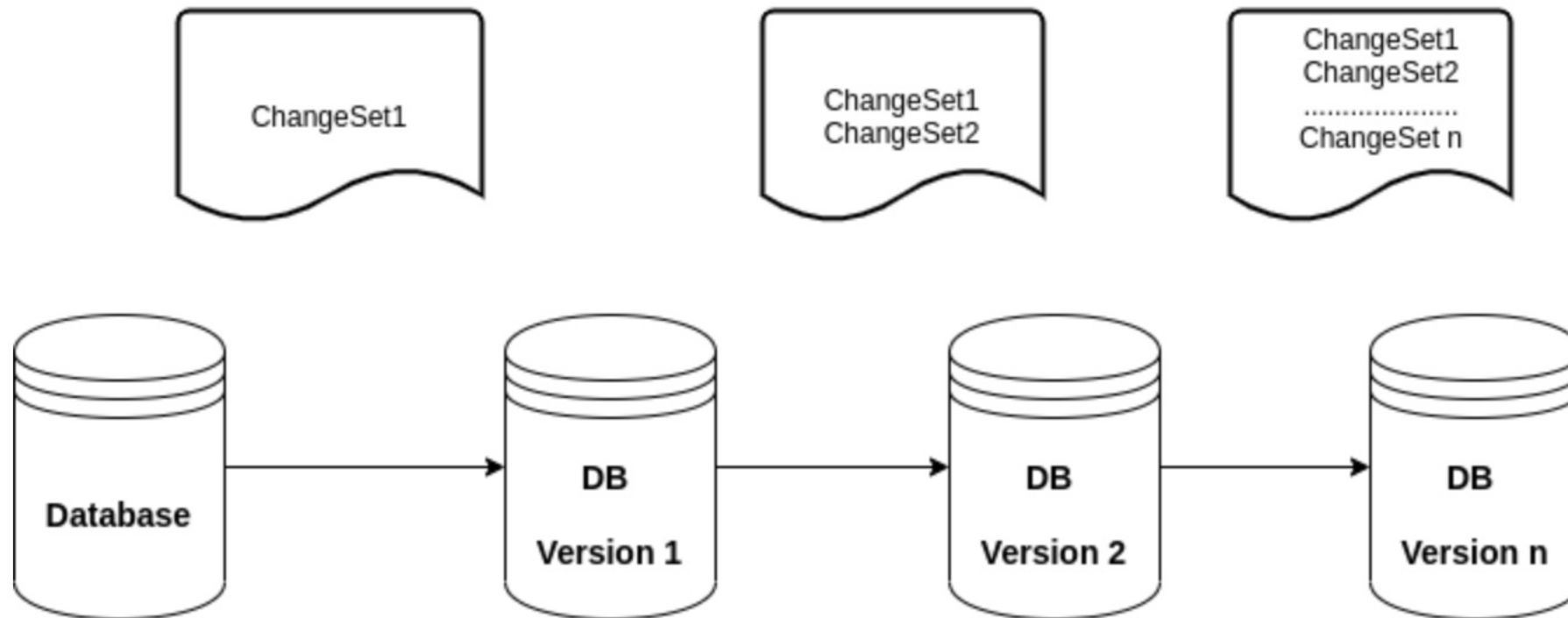- **https://hellokoding.com/database-migration-evolution-with-flyway-and-jpa-hibernate/**

- **https://www.baeldung.com/database-migrations-with-flyway**

PART 3: Liquibase

LiquiBase Basics

LiquiBase Basics

```xml
<changeSet id="1" author="michael">
    <addColumn tableName="person">
    <column name="last_modified_date"  type="timestamp"/>
    </addColumn>
</changeSet>


<changeSet id="multipleCreates" author="michael">
    <createTable tableName="person">
        <column name="id" type="int"/>
        <column name="username" type="varchar(30)"/>
    </createTable>

    <createTable tableName="roles">
        <column name="id" type="int"/>
        <column name="name" type="varchar(30)"/>
        <column name="description" type="varchar(2000)"/>
    </createTable>
</changeSet>
```

LiquiBase Basics

- Liquibase verwendet **changeSets**, um eine einzelne **Änderung** an der Datenbank zu beschreiben.

- Ein **changeSet** besteht aus einer Menge / Einheit von Änderungen, die Liquibase an einer Datenbank durchführt.

- Ein **changeLog** ist eine Liste von Änderungen, die durch mehrere **changeSets** erstellt wurden

- Ein **changeSet** wird durch 3 Elemente identifiziert: "ID" und "Autor" und das Enthaltensein im **changeLog**.

LiquiBase Basics

```xml
<?xml version="1.1" encoding="UTF-8" standalone="no"?>
<databaseChangeLog
  xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
                      http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.9.xsd">


    <changeSet author="tim (generated)" id="1571293572660-1">
        <createTable tableName="TAB1">
            <column name="ID" type="NUMBER">
                <constraints primaryKey="true" primaryKeyName="TAB1_PK"/>
            </column>
            <column name="DESCRIPTION" type="VARCHAR2(50 BYTE)"/>
        </createTable>
    </changeSet>
    <changeSet author="tim (generated)" id="1571293572660-2">
        <createSequence maxValue="9999999999999999999999999999" sequenceName="TAB1_SEQ"
                        startValue="21"/>
    </changeSet>
</databaseChangeLog>
```

Externe Infos

- **https://www.liquibase.org/**

- **https://www.liquibase.org/get-started**

- **https://www.liquibase.org/get-started/best-practices**

- **https://www.liquibase.org/get-started/how-liquibase-works**

- **https://blog.codecentric.de/en/2015/01/managing-database-migrations-using-liquibase/**

Externe Infos

- **https://www.baeldung.com/liquibase-refactor-schema-of-java-app**

- **https://medium.com/podiihq/getting-started-with-liquibase-8965897092aa**

- **https://oracle-base.com/articles/misc/liquibase-automating-your-database-deployments**

- **https://www.cloudbees.com/blog/liquibase-tutorial-manage-database-schema/**

- **https://community.bonitasoft.com/blog/liquibase-versioning-database-schema**

Questions?

Thank You