Flyway

Flyway是什么

官网解释地非常全面，可先大致阅读一下。

简单地说，`flyway`是一个能对数据库变更做版本控制的工具。

为什么要用Flyway

在多人开发的项目中，我们都习惯了使用`SVN`或者`Git`来对代码做版本控制，主要的目的就是为了解决多人开发代码冲突和版本回退的问题。

其实，数据库的变更也需要版本控制，在日常开发中，我们经常会遇到下面的问题：

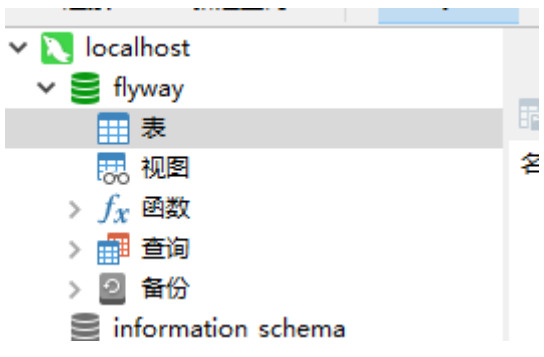
1. 自己写的`SQL`忘了在所有环境执行；
2. 别人写的`SQL`我们不能确定是否都在所有环境执行过了；
3. 有人修改了已经执行过的`SQL`，期望再次执行；
4. 需要新增环境做数据迁移；
5. 每次发版需要手动控制先发DB版本，再发布应用版本；
6. 其它场景...

有了`flyway`，这些问题都能得到很好的解决。

如何使用Flyway

3.1 准备数据库

在`mysql`中新建一个名为`flyway`的数据库



3.2 准备SpringBoot工程

创建项目（创建时可以选择`flyway`依赖），并配置数据源：

```yaml
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/flyway?serverTimezone=Asia/Shanghai
    username: root
    password: 123456
```

依赖如下：

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
```

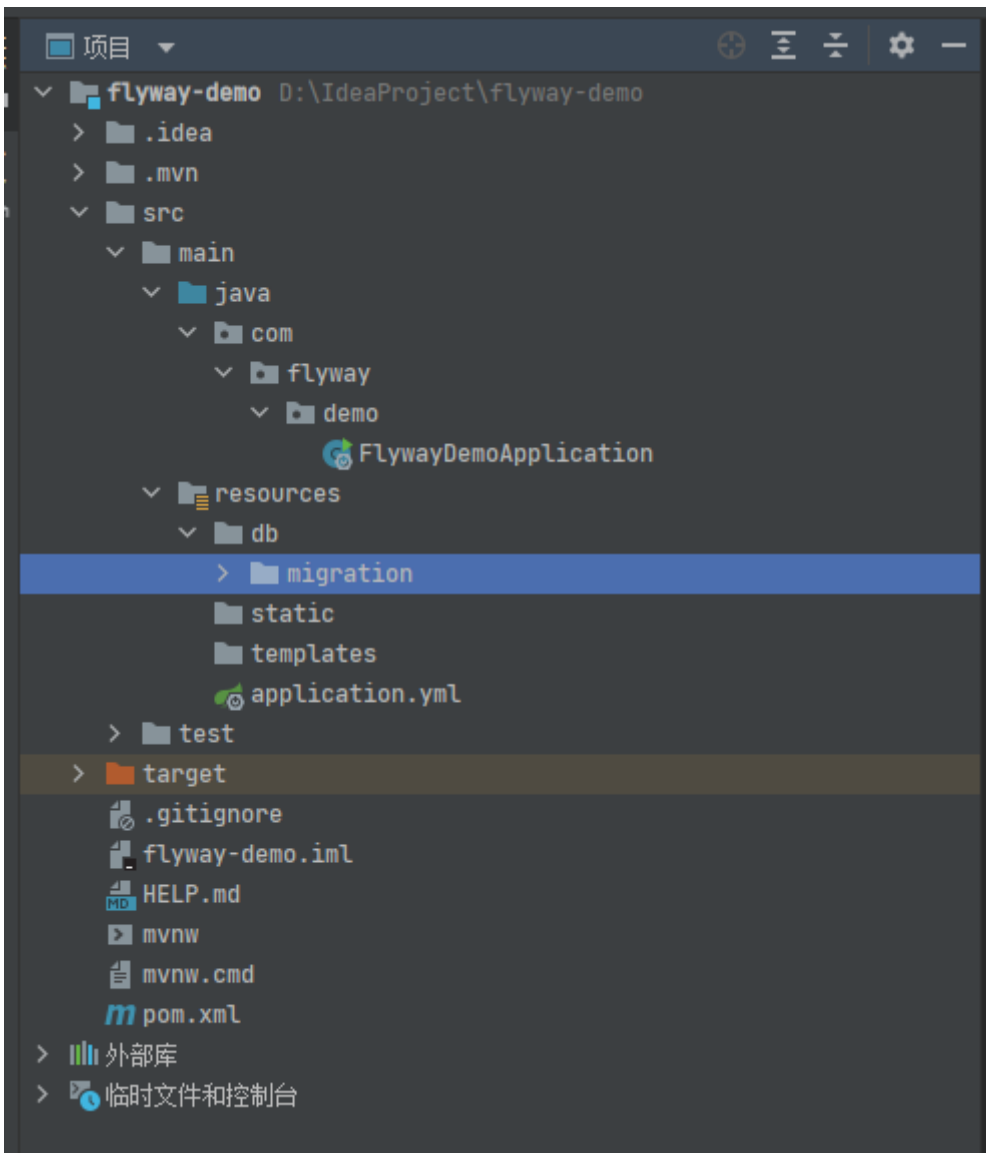
```
        <version>2.6.1</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.flyway</groupId>
    <artifactId>flyway-demo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>flyway-demo</name>
    <description>flyway-demo</description>
    <properties>
        <java.version>1.8</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.flywaydb</groupId>
            <artifactId>flyway-core</artifactId>
        </dependency>

        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <scope>runtime</scope>
        </dependency>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>


    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.flywaydb</groupId>
                <artifactId>flyway-maven-plugin</artifactId>
                <version>5.2.4</version>
```

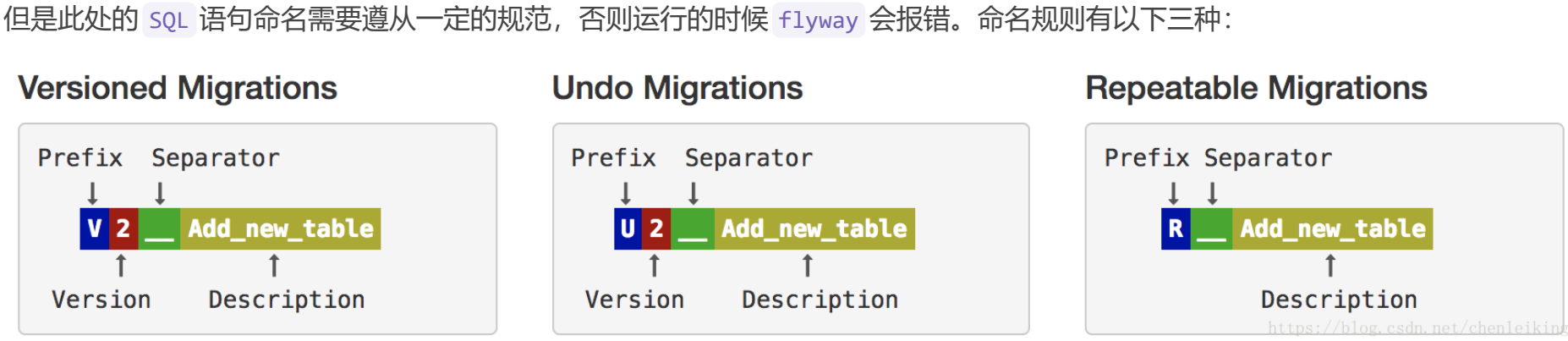
```
                <configuration>
                    <url>jdbc:mysql://localhost:3306/flyway?
useUnicode=true&amp;characterEncoding=utf8&amp;serverTimezone=GMT</url>
                    <user>root</user>
                    <password>root</password>
                    <driver>com.mysql.cj.jdbc.Driver</driver>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

目录结构如下：

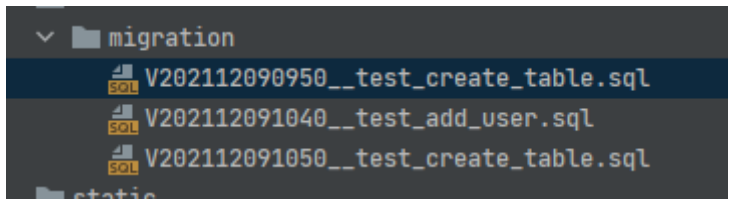


可以看到，自动帮我们创建了`db.migration`文件夹，默认情况下，该目录下的`.sql`文件就算是需要被flyway做版本控制的数据库`SQL`语句。

但是此处的`SQL`语句命名需要遵从一定的规范，否则运行的时候`flyway`会报错。命名规则有以下三种：



> **注意：`separator`是双下划线**
>
> 确保版本号唯一，`flyway`按照版本号顺序执行。`repeatable`没有版本号，因为`repeatable migrations`会在内容改变时重复执行
>
> 其中`V`，`U`，`R`分别对应`版本化迁移`、`撤消迁移`、`可重复迁移`
>
> 版本号推荐用时间(`yyyyMMddHHmm`)，举个栗子：`202112091110`



到这一步，flyway的默认配置已经足够我们开始运行了。此时，我们启动`SpringBoot`的主程序，如果以上步骤没有配置错误的话，运行截图如下：

```
2021-12-09 11:33:02.420  INFO 12692 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2021-12-09 11:33:02.447  INFO 12692 --- [           main] o.f.c.internal.license.VersionPrinter    : Flyway Community Edition 8.0.4 by Redgate
2021-12-09 11:33:02.447  INFO 12692 --- [           main] o.f.c.i.database.base.BaseDatabaseType   : Database: jdbc:mysql://localhost:3306/flyway (MySQL 8.0)
2021-12-09 11:33:02.491  INFO 12692 --- [           main] o.f.core.internal.command.DbValidate     : Successfully validated 3 migrations (execution time
 00:00.020s)
2021-12-09 11:33:02.528  INFO 12692 --- [           main] o.f.c.i.s.JdbcTableSchemaHistory         : Creating Schema History table `flyway`
 .`flyway_schema_history` ...
2021-12-09 11:33:03.132  INFO 12692 --- [           main] o.f.core.internal.command.DbMigrate      : Current version of schema `flyway`: << Empty Schema >>
2021-12-09 11:33:03.147  INFO 12692 --- [           main] o.f.core.internal.command.DbMigrate      : Migrating schema `flyway` to version "202112090950 - test
create table"
2021-12-09 11:33:03.420  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8mb3' is deprecated and will be removed in a future
 release. Please use utf8mb4 instead (SQL State: HY000 - Error Code: 1287)
2021-12-09 11:33:04.559  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8mb3' is deprecated and will be removed in a future
 release. Please use utf8mb4 instead (SQL State: HY000 - Error Code: 1287)
2021-12-09 11:33:05.095  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8' is currently an alias for the character set
 UTF8MB3, but will be an alias for UTF8MB4 in a future release. Please consider using UTF8MB4 in order to be unambiguous. (SQL State: HY000 - Error Code: 3719)
2021-12-09 11:33:05.095  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8_general_ci' is a collation of the deprecated
 character set UTF8MB3. Please consider using UTF8MB4 with an appropriate collation instead. (SQL State: HY000 - Error Code: 3778)
2021-12-09 11:33:05.095  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8' is currently an alias for the character set
 UTF8MB3, but will be an alias for UTF8MB4 in a future release. Please consider using UTF8MB4 in order to be unambiguous. (SQL State: HY000 - Error Code: 3719)
2021-12-09 11:33:05.095  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8_general_ci' is a collation of the deprecated
 character set UTF8MB3. Please consider using UTF8MB4 with an appropriate collation instead. (SQL State: HY000 - Error Code: 3778)
2021-12-09 11:33:05.095  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8mb3' is deprecated and will be removed in a future
 release. Please use utf8mb4 instead (SQL State: HY000 - Error Code: 1287)
2021-12-09 11:33:05.353  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8mb3' is deprecated and will be removed in a future
 release. Please use utf8mb4 instead (SQL State: HY000 - Error Code: 1287)
2021-12-09 11:33:05.827  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8mb3' is deprecated and will be removed in a future
 release. Please use utf8mb4 instead (SQL State: HY000 - Error Code: 1287)
2021-12-09 11:33:05.911  INFO 12692 --- [           main] o.f.core.internal.command.DbMigrate      : Migrating schema `flyway` to version "202112091040 - test
 add user"
2021-12-09 11:33:06.009  INFO 12692 --- [           main] o.f.core.internal.command.DbMigrate      : Migrating schema `flyway` to version "202112091050 - test
 create table"
2021-12-09 11:33:06.211  WARN 12692 --- [           main] o.f.c.i.s.DefaultSqlScriptExecutor       : DB: 'utf8mb3' is deprecated and will be removed in a future
 release. Please use utf8mb4 instead (SQL State: HY000 - Error Code: 1287)
```

名	自动递增值	修改日期	数据长度	引擎	行	注释
flyway_schema_history	0	2021-12-09 11:3...	16 KB	InnoDB	3	
menu	0		16 KB	InnoDB	0	菜单
menu_role	0		16 KB	InnoDB	0	菜单角色关联表
role	0		16 KB	InnoDB	0	角色
tesrt	0		16 KB	InnoDB	0	用户角色关联表
user	2	2021-12-09 11:3...	16 KB	InnoDB	1	用户
user_role	0		16 KB	InnoDB	0	用户角色关联表

此时，我们刷新数据库，可以看到`flyway`的历史记录表已经生成并插入了三个版本的记录：

installed_rank	version	description	type	script	checksum	installed_by	installed_on	execution_time	success
1	202112090950	test create table	SQL	V202112090950__test_create_table.sql	1886281998	root	2021-12-09 11:33:05	2683	1
2	202112091040	test add user	SQL	V202112091040__test_add_user.sql	123844679	root	2021-12-09 11:33:05	19	1
3	202112091050	test create table	SQL	V202112091050__test_create_table.sql	-588673816	root	2021-12-09 11:33:06	206	1

而且，`user`表也已经创建好了并插入了一条数据：

对象 | user @flyway (localhost) - 表

开始事务 | 文本 | 筛选 | 排序 | 导入 | 导出

id	name	password	email	create_time	update_time	version	deleted
1	1	1	1	2021-12-09 11:33:05	(Null)	1	0

我们不改变任何东西，再次执行主程序，日志如下：

```
2021-12-09 11:36:42.388  INFO 10728 --- [           main] o.f.c.internal.license.VersionPrinter    : Flyway Community Edition 8.0.4 by Redgate
2021-12-09 11:36:42.388  INFO 10728 --- [           main] o.f.c.i.database.base.BaseDatabaseType   : Database: jdbc:mysql://localhost:3306/flyway (MySQL 8.0)
2021-12-09 11:36:42.420  INFO 10728 --- [           main] o.f.core.internal.command.DbValidate     : Successfully validated 3 migrations (execution time
 00:00.015s)
2021-12-09 11:36:42.429  INFO 10728 --- [           main] o.f.core.internal.command.DbMigrate      : Current version of schema `flyway`: 202112091050
2021-12-09 11:36:42.429  INFO 10728 --- [           main] o.f.core.internal.command.DbMigrate      : Schema `flyway` is up to date. No migration necessary.
```

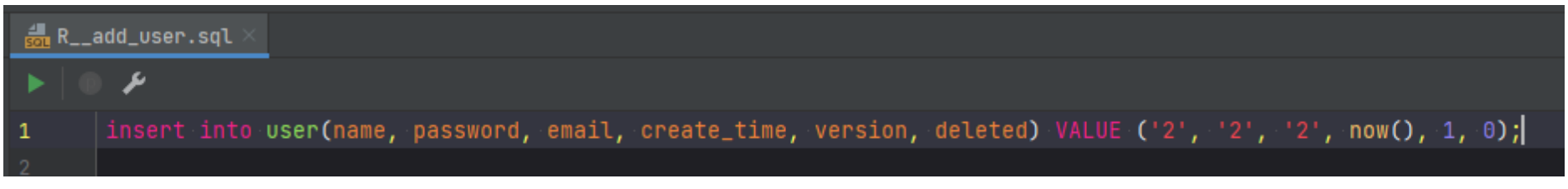
数据库的表中的内容也毫无任何变化。

我们来修改其中一张表，再次启动试试：

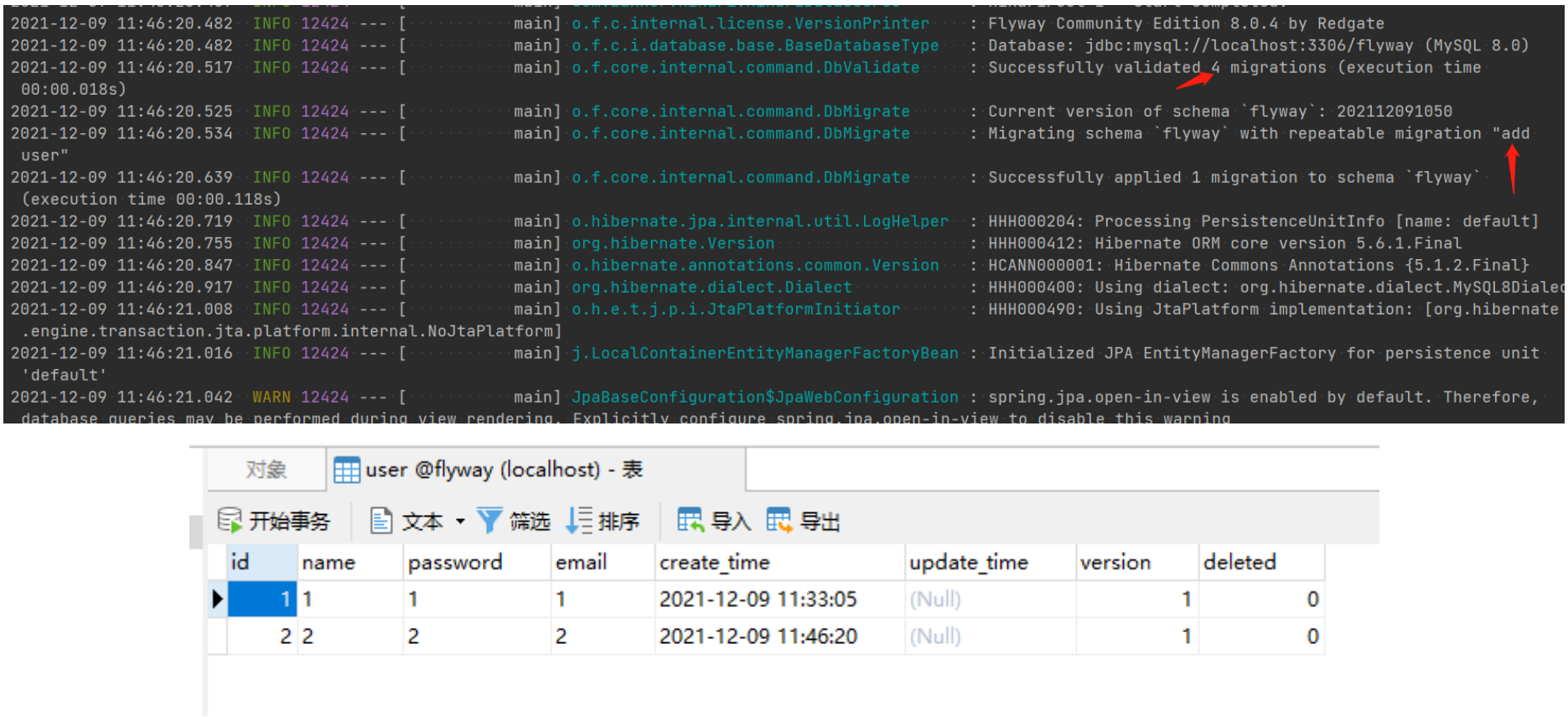
```
[org/springframework/boot/autoconfigure/flyway/FlywayAutoConfiguration$FlywayConfiguration.class]: Invocation of init method failed; nested exception i
 org.flywaydb.core.api.exception.FlywayValidateException: Validate failed: Migrations have failed validation
Migration checksum mismatch for migration version 202112091050
-> Applied to database : -588673816
-> Resolved locally    : -123644608. Either revert the changes to the migration, or run repair to update the schema history.
Need more flexibility with validation rules? Learn more: https://rd.gt/3AbJUZE
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.initializeBean(AbstractAutowireCapableBeanFactory.java:1804) ~[sprin
 .3.13.jar:5.3.13]
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.doCreateBean(AbstractAutowireCapableBeanFactory.java:620) ~[spring-b
 .13.jar:5.3.13]
    at org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.createBean(AbstractAutowireCapableBeanFactory.java:542) ~[spring-bea
 .jar:5.3.13]
    at org.springframework.beans.factory.support.AbstractBeanFactory.lambda$doGetBean$0(AbstractBeanFactory.java:335) ~[spring-beans-5.3.13.jar:5.3.13]
    at org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.getSingleton(DefaultSingletonBeanRegistry.java:234) ~[spring-beans-5.3.13.
 .13]
    at org.springframework.beans.factory.support.AbstractBeanFactory.doGetBean(AbstractBeanFactory.java:333) ~[spring-beans-5.3.13.jar:5.3.13]
```

> 第一行是报错信息，第三行是解决方案（之一）：将更改还原到迁移，或者运行修复以更新架构历史记录。

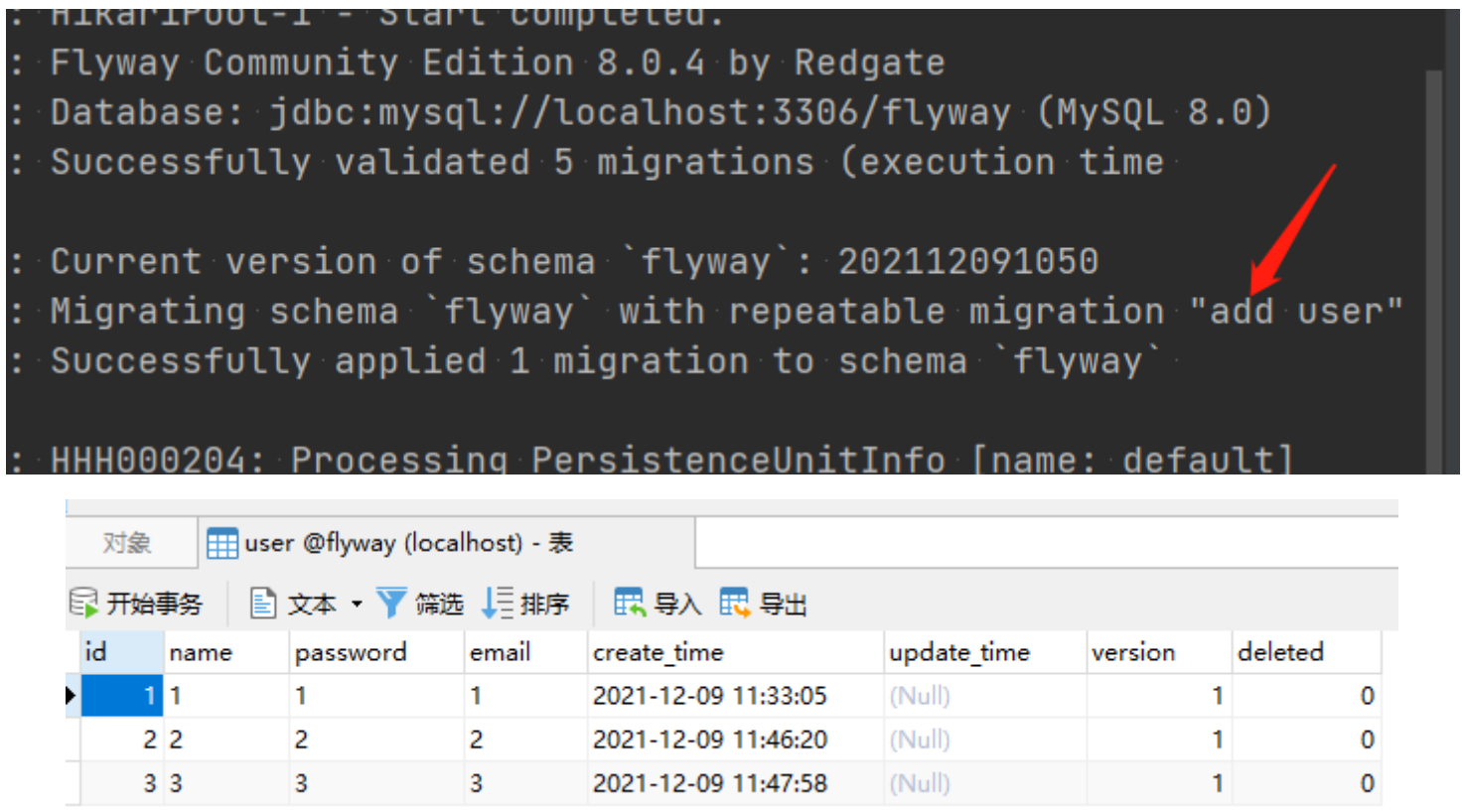
我们来创建一个 `R` （可重复迁移）开的 `sql` 文件，多次运行：



> 第一次



> 第二次，修改后运行



可以看到， `R` 开头的 `sql` 是可以循环运行的。

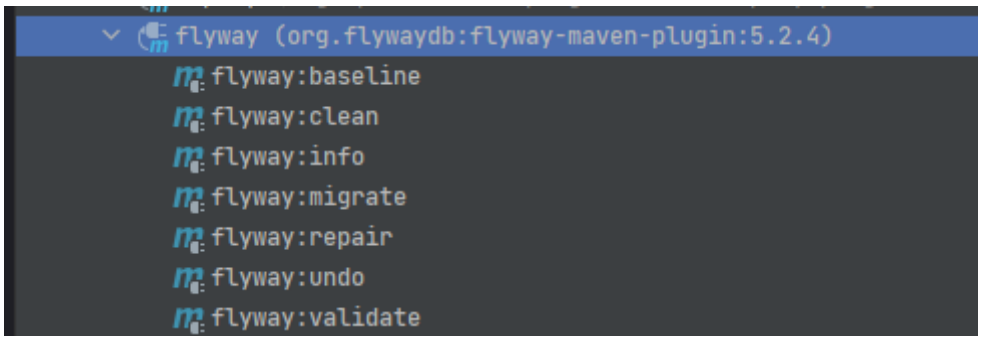
3.4 maven插件的使用

以上步骤中，每次想要 `migration` 都需要运行整个 `springboot` 项目，并且只能执行 `migrate` 一种命令，其实 `flyway` 还是有很多其它命令的。 `maven` 插件给了我们不需要启动项目就能执行 `flyway` 各种命令的机会。

在 `pom` 中引入 `flyway` 的插件，同时配置好对应的数据库连接。

```
<plugin>
    <groupId>org.flywaydb</groupId>
    <artifactId>flyway-maven-plugin</artifactId>
    <version>5.2.4</version>
    <configuration>
        <url>jdbc:mysql://localhost:3306/flyway?
useUnicode=true&amp;characterEncoding=utf8&amp;serverTimezone=GMT</url>
        <user>root</user>
        <password>root</password>
        <driver>com.mysql.cj.jdbc.Driver</driver>
    </configuration>
</plugin>
```

然后更新 `maven` 插件列表，就可以看到 `flyway` 的全部命令了。



此时，我们双击执行上图中的 `flyway:migrate` 的效果和启动整个工程执行 `migrate` 的效果是一样的。

其它命令的作用如下列出，各位可自行实验体会：

1. baseline

   对已经存在数据库 `Schema` 结构的数据库一种解决方案。实现在非空数据库新建 `MetaData` 表，并把 `Migrations` 应用到该数据库；也可以在已有表结构的数据库中实现添加 `Metadata` 表。

2. clean

   清除掉对应数据库 `Schema` 中所有的对象，包括表结构，视图，存储过程等， `clean` 操作在 `dev` 和 `test` 阶段很好用，但在生产环境务必禁用。

3. info

   用于打印所有的 `Migrations` 的详细和状态信息，也是通过 `MetaData` 和 `Migrations` 完成的，可以快速定位当前的数据库版本。

4. repair

   repair操作能够修复 `metaData` 表，该操作在 `metadata` 出现错误时很有用。

5. undo

   撤销操作，社区版不支持。

6. validate

   验证已经 `apply` 的 `Migrations` 是否有变更，默认开启的，原理是对比 `MetaData` 表与本地 `Migrations` 的 `checkNum` 值，如果值相同则验证通过，否则失败。

3.5 flyway补充知识

1. `flyway` 执行 `migrate` 必须在空白的数据库上进行，否则报错；
2. 对于已经有数据的数据库，必须先 `baseline` ，然后才能 `migrate` ；
3. `clean` 操作是删除数据库的所有内容，包括 `baseline` 之前的内容；
4. 尽量不要修改已经执行过的 `SQL` ，即便是R开头的可反复执行的 `SQL` ，它们会不利于数据迁移；

附录

`flyway` 的配置清单：

```
flyway.baseline-description对执行迁移时基准版本的描述.
flyway.baseline-on-migrate当迁移时发现目标schema非空，而且带有没有元数据的表时，是否自动执行基准迁移，默认false.
```

flyway.baseline-version开始执行基准迁移时对现有的schema的版本打标签，默认值为1.

flyway.check-location检查迁移脚本的位置是否存在，默认false.

flyway.clean-on-validation-error当发现校验错误时是否自动调用clean，默认false.

flyway.enabled是否开启flywary，默认true.

flyway.encoding设置迁移时的编码，默认UTF-8.

flyway.ignore-failed-future-migration当读取元数据表时是否忽略错误的迁移，默认false.

flyway.init-sqls当初始化好连接时要执行的SQL.

flyway.locations迁移脚本的位置，默认db/migration.

flyway.out-of-order是否允许无序的迁移，默认false.

flyway.password目标数据库的密码.

flyway.placeholder-prefix设置每个placeholder的前缀，默认${.

flyway.placeholder-replacementplaceholders是否要被替换，默认true.

flyway.placeholder-suffix设置每个placeholder的后缀，默认}.

flyway.placeholders.[placeholder name]设置placeholder的value

flyway.schemas设定需要flywary迁移的schema，大小写敏感，默认为连接默认的schema.

flyway.sql-migration-prefix迁移文件的前缀，默认为V.

flyway.sql-migration-separator迁移脚本的文件名分隔符，默认__

flyway.sql-migration-suffix迁移脚本的后缀，默认为.sql

flyway.tableflyway使用的元数据表名，默认为schema_version

flyway.target迁移时使用的目标版本，默认为latest version

flyway.url迁移时使用的JDBC URL，如果没有指定的话，将使用配置的主数据源

flyway.user迁移数据库的用户名

flyway.validate-on-migrate迁移时是否校验，默认为true