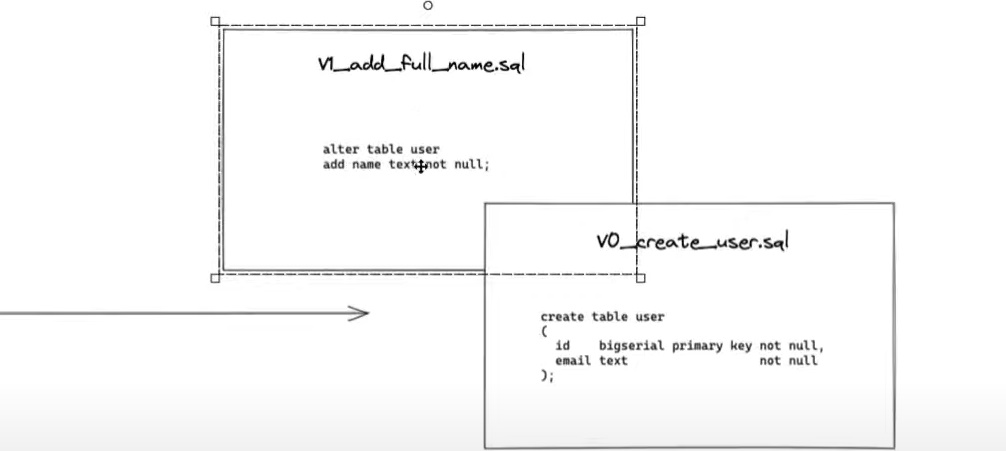
* Migration - doar un sql script care ia baza de date de la un state la altul, care modifica schema doar
* Migration ne permite sa impartim baza de date pe versiuni.



* E ceva asemanator cu commiturile. V0 avea un tabel user, apoi V1 deja a adaugat in tabel o noua coloana.
* Flyway stie mereu la ce versiune suntem la momentul actual
* FlyWay poate crea si schema, ceea ce alte frameworks pentru migration nu pot

**Pasi**

1. Plugin:

<plugin>  
 <groupId>org.flywaydb</groupId>  
 <artifactId>flyway-maven-plugin</artifactId>  
 <version>9.16.0</version>  
</plugin>

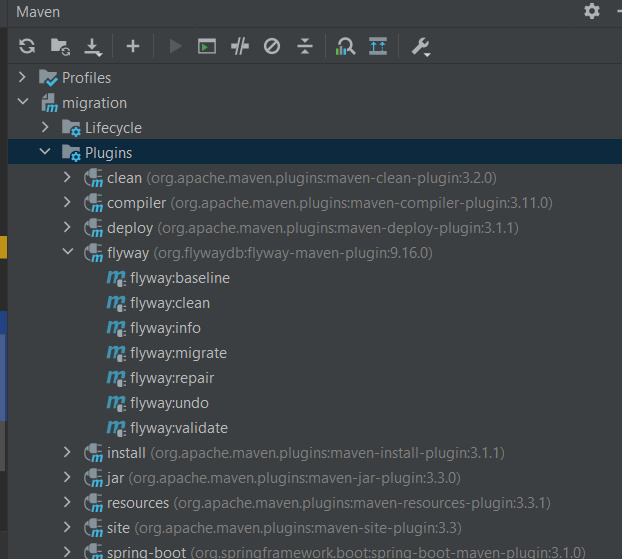
si dependenta, pentru mysql:

<dependency>  
 <groupId>org.flywaydb</groupId>  
 <artifactId>flyway-mysql</artifactId>  
 <version>9.16.0</version>  
</dependency>

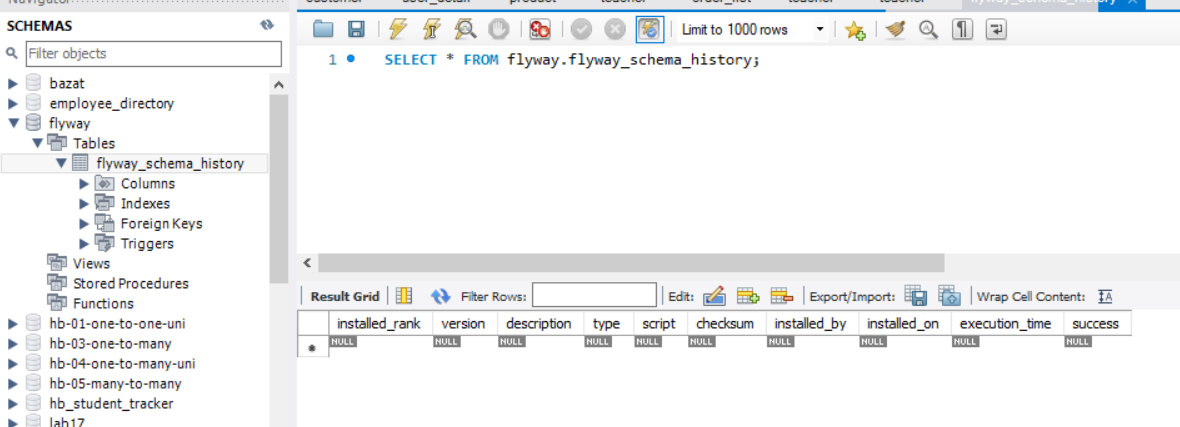
Apoi va trebui sa configuram flyway:

<plugin>  
 <groupId>org.flywaydb</groupId>  
 <artifactId>flyway-maven-plugin</artifactId>  
 <version>9.16.0</version>  
 <configuration>  
 <user>testuser</user>  
 <password>Frb2eshox!</password>  
 <url>jdbc:mysql://localhost:3306/flyway</url>  
 <schemas>  
 <schema>flyway</schema>  
 </schemas>  
 <driver>com.mysql.jdbc.Driver</driver>  
 </configuration>  
</plugin>

1. Proiectul se cheama migration, si aici gasim flyway:



dam migrate. Ca baza de date exista sau nu, insa daca nu, va crea schema, flyway va crea un tabel in schema:



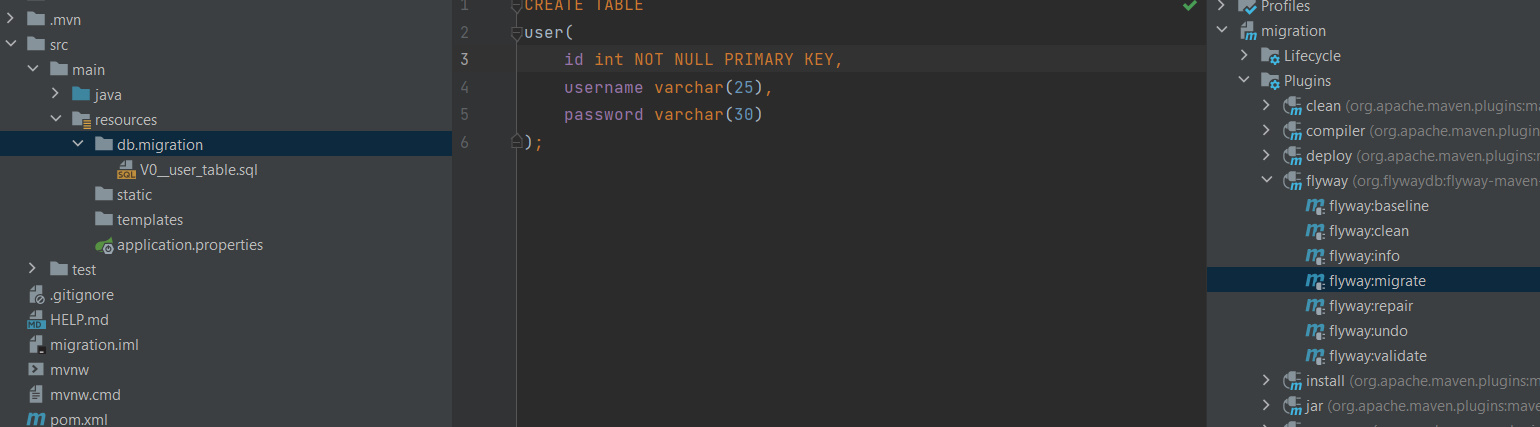
acesta e schema version table

**Add sql scripts**

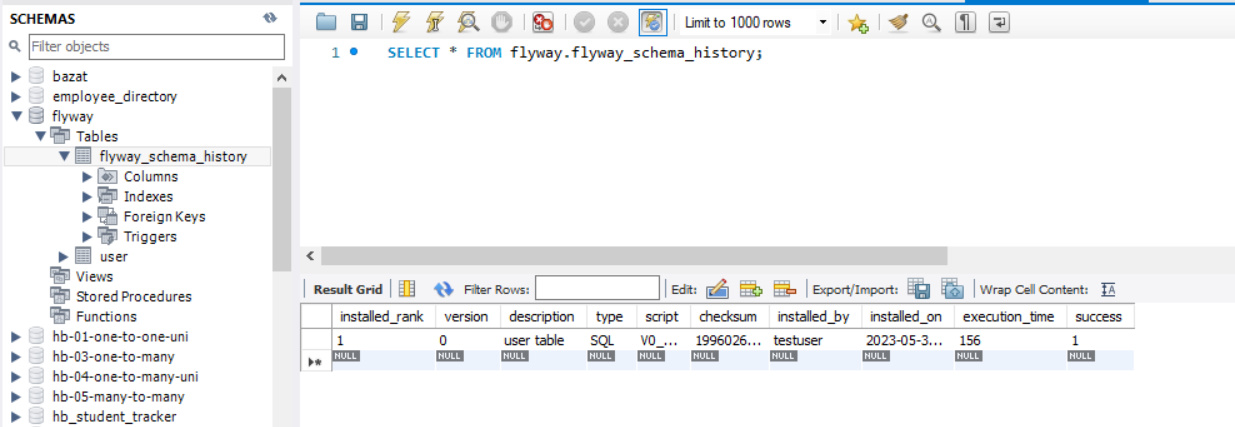
* Flyway cauta scripturile noastre in resources/db/migration
* Numele la fisiere urmeaza o conventie:

Vversiune\_\_descriere\_cu\_underscore.sql

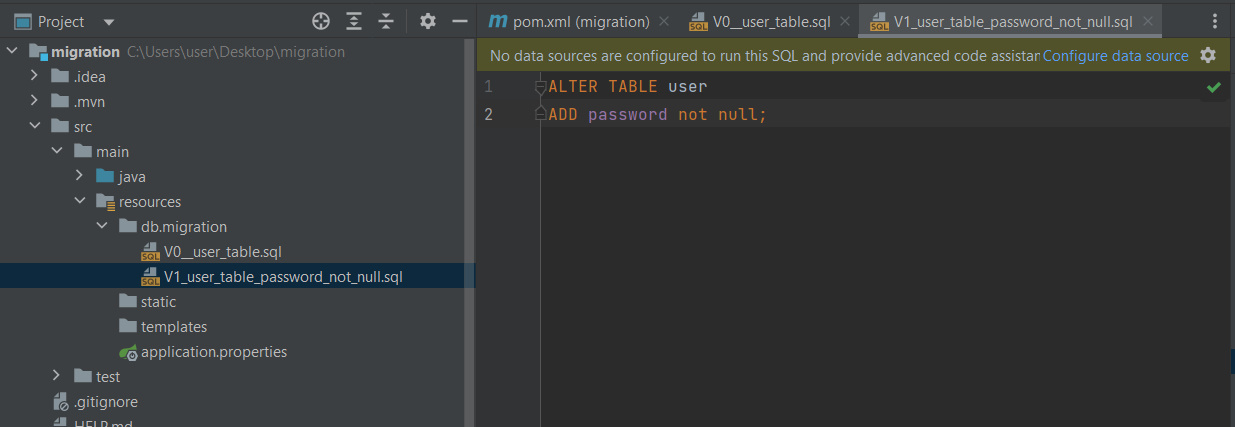
Apoi migrate:



si tabelul a fost creat iar flyway a inregistrat aceasta informatie in tabelul lui



Acum, daca vrem peste un timp sa cream o constrangere in table, ca not null pentru password, mai cream un fisier si gata:



**Commands**

Flyway supports the following basic commands to manage database migrations:

* ***Info*:** Prints current status/version of a database schema. It prints which migrations are pending, which migrations have been applied, the status of applied migrations, and when they were applied.
* ***Migrate*:** Migrates a database schema to the current version. It scans the classpath for available migrations and applies pending migrations.
* ***Baseline*:** Baselines an existing database, excluding all migrations, including *baselineVersion*. Baseline helps to start with Flyway in an existing database. Newer migrations can then be applied normally.
* ***Validate*:** Validates current database schema against available migrations.
* ***Repair*:** Repairs metadata table.
* ***Clean*:** Drops all objects in a configured schema. Of course, we should never use *clean* on any production database.