INSTALLATION PROCEDURE AND STARTING THE APLICATION

OVERVIEW

This project is developed with Eclipse IDE. A standard project can use a large Database like Oracle, PostgreSql, MySql and so. Here this is not possible due to the portability to the project. So an important part of the project uses HSQLDB.
HSQLDB is the **natural** evolution of H2 Database, but unlike H2 which obliges to recreate all the tables each time you use the project because H2 is only in RAM, HSQLB can use a file representing the BDD. Here you have to prepare the using **only the first time**. During subsequent connections the tables will exist.
You can find all the libraries of HSQLDB under a "`resources`" repository like an important part of the project in a" `hsqldb`" repository.
COMMENT : Don't pay attention to the red cross on " `hsqldb` " repository, it's due to an URI in the library, that not **cause** any problems to the project

When you clone the project from a gitHub to your PC, the project will be installed on a repository which has, of course a letter to define it. On my pc this letter is "`E`" but for convenience in the rest of the description I will replace this letter by "**`xxxx`**"

DESCRIPTION OF THE USAGE OF THE EMBEDDED DATABASE

Before launching the application **the very first time only**, you must launch a Windows "`cmd`" command prompt in order to create the database, in the hsqldb directory which is for example for me under:

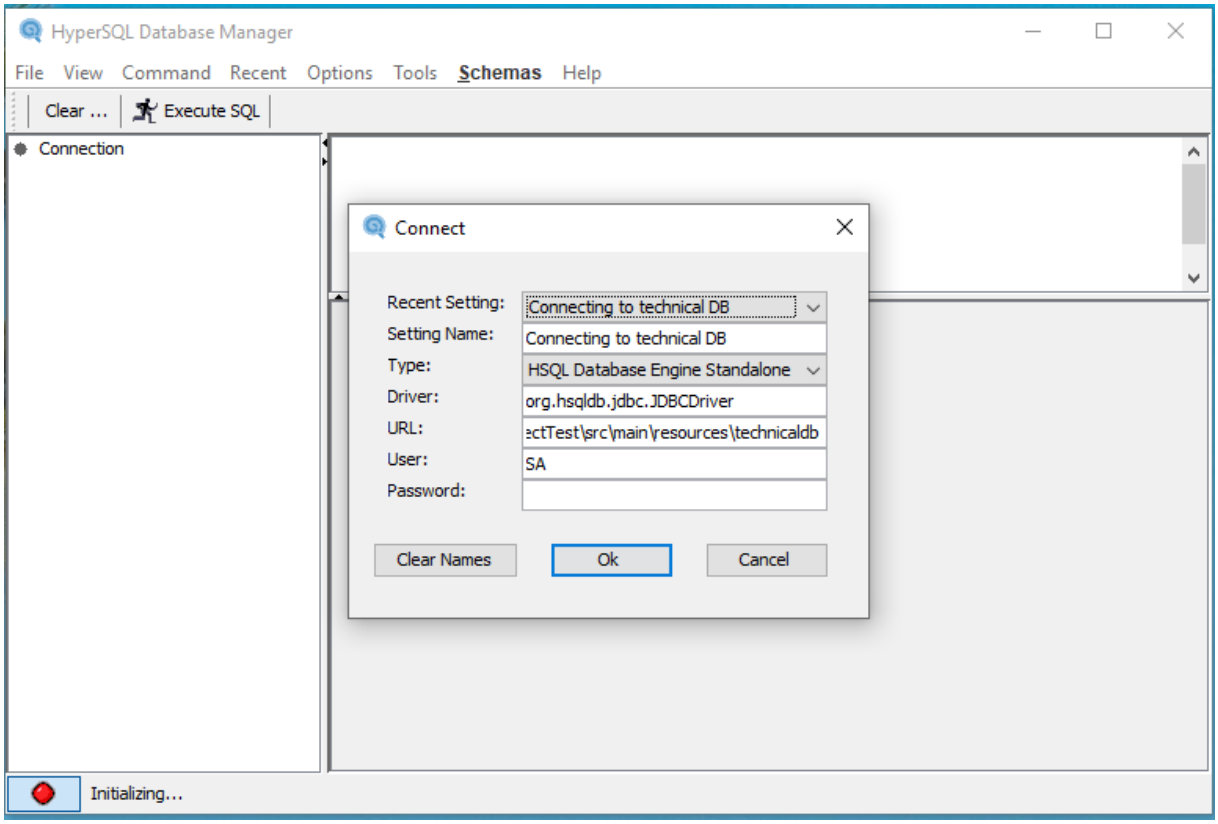
```
xxxx\projectTest\src\main\resources\hsqldb
```

COMMENT: The most important thing is to find the path of the hsqldb directory after getting the project from GITHUB on your PC.

And enter the following line:

```
java -cp ./lib/hsqldb.jar org.hsqldb.util.DatabaseManagerSwing
```

COMMENT : Le `hsqldb.jar` is in the current folder.
This action will launch the "`database Manager`"



COMMENT : Keep opened you "`cmd`" command to have your "`database Manager`" window.

Information contained in the popup "`Connect`"

Recent Setting	Leave it as it is (will be changed automaticly)
Setting Name	Type the text you want (here : `Connecting to technical DB`)
Type	`Select HSQL Database Engine Standalone`
Driver	Leave it as it is
URL	`jdbc:hsqldb:file:E:\projectTest\src\main\resources\technicaldb` **
User	`SA`
Password	Leave it blank

** The path depends on where "`projectTest\src\main\resources`" is located to create the technicaldb script DB name

Click on "**Ok**" button

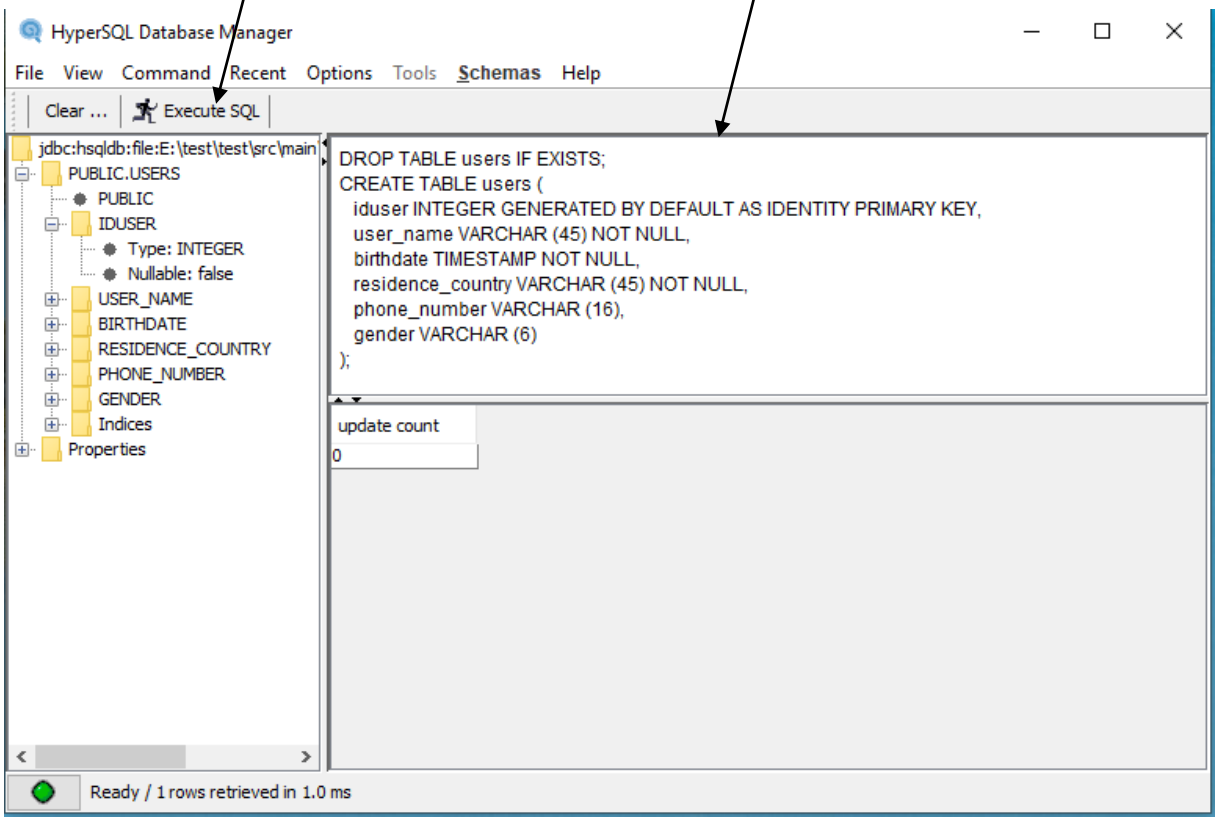
Location of the database is now at my place:

`xxxx\projectTest\src\main\resources\technicaldb`

COMMENT : this may be different depending on where the project is located

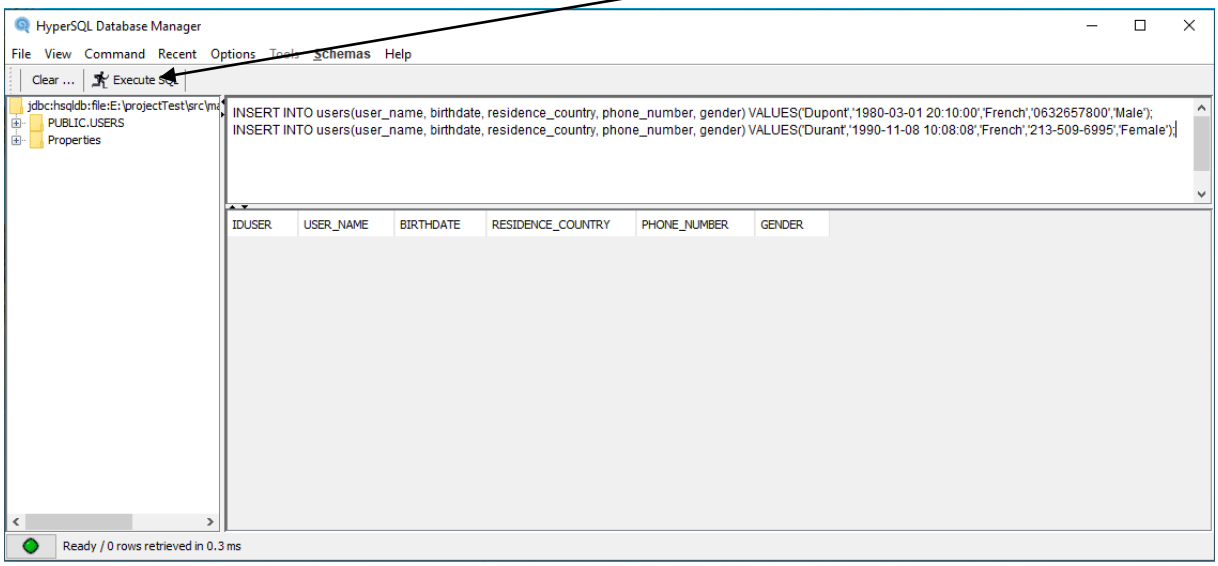
A new window opens in which the table that will be contained in the "`technicaldb`" database is created.

The SQL code for creating the table is in the "`src/main/resources/sql_tables`" folder, copy/paste the code to create the table in the location of the image,
and click on "**Execute SQL**".



From there on the left panel we see the table and its columns.

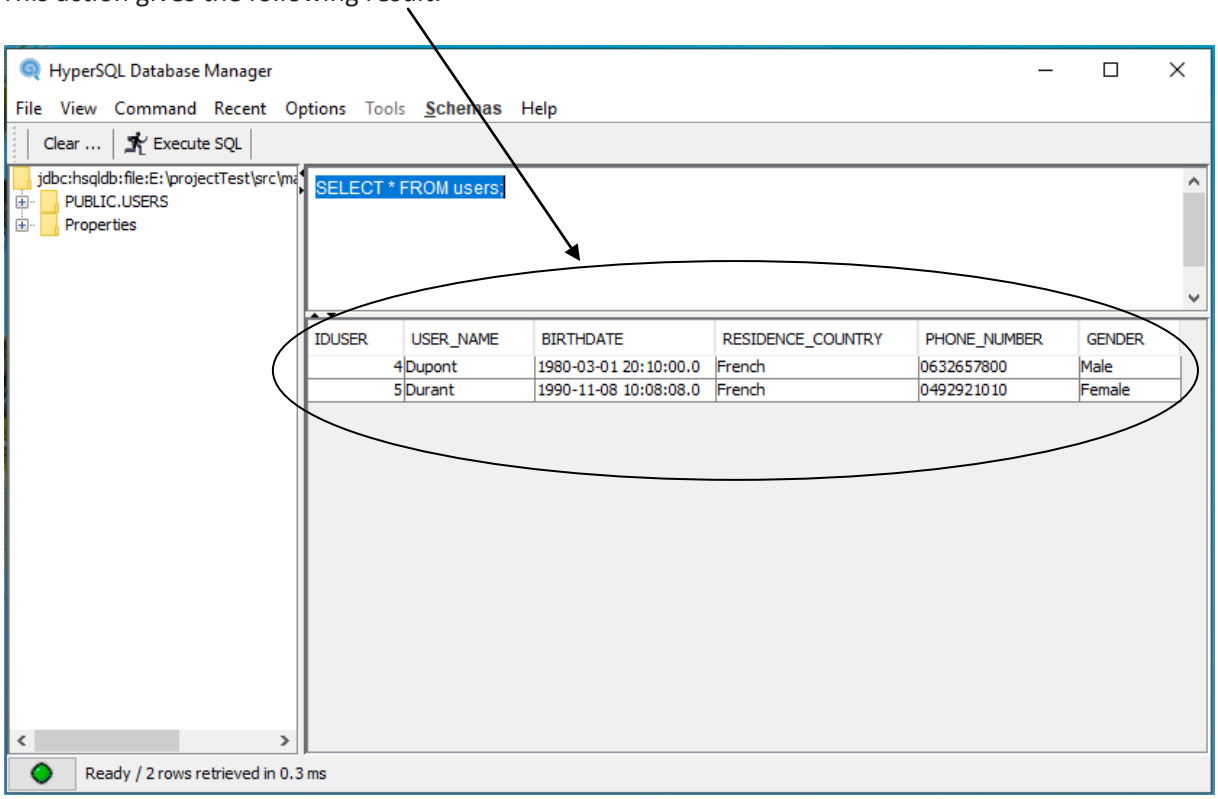
Still from the SQL code, we will populate the table created.
Copy/paste the code found in the "`src/main/resources/sql_tables` " folder
(Like for example: `INSERT INTO users`....), and click on "**Execute SQL**".



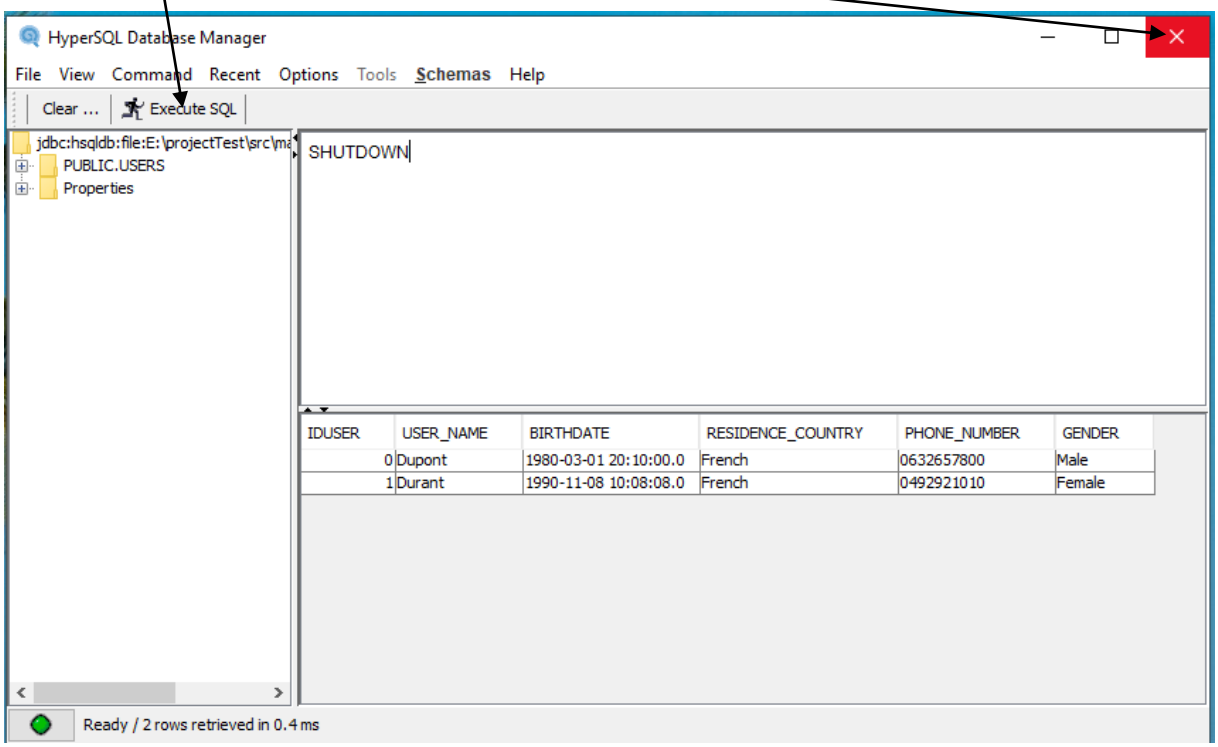
In order to check that the 2 lines are well recorded, we enter the following instruction as shown in the image:

```
SELECT * FROM users;
```

This action gives the following result:



At the end to close the database manager type : `shutdown` in the image,
click on "**Execute SQL**", and click on the "**red cross**" at the high right of the application.



IMPORTANT COMMENT : if the "database manager" is opened, when you run the project the Spring boot application won't start, so have to close the "database manager" before. The opposite thing is also true

For information only:

If the database was created in server mode:

Using a Windows "cmd" command prompt, in my case go to:

```
xxxx\projectTest\src\main\resources\hsqldb
```

And enter the following line:

```
java -cp ./lib/hsqldb.jar org.hsqldb.server.Server --database.0
file.technicaldb --dbname0.testdb
```

To launch the database, from a "cmd" command prompt, for me go to:

```
xxxx\projectTest\src\main\resources\hsqldb \data
```

And enter the following line:

```
java -cp ../lib/hsqldb.jar org.hsqldb.server.Server --database.0
file:technicaldb --dbname.0 testdb
```

This action launches the server and connects to the previously created database that we leave running.

By opening another Windows command prompt "cmd" and to go to the directory:

```
xxxx\projectTest\src\main\resources\hsqldb
```

And enter the following line:

```
java -cp ./lib/hsqldb.jar org.hsqldb.util.DatabaseManagerSwing
```

This action will launch the "database Manager"
COMMENT : Le hsqldb.jar is in the current folder.

If you want to launch the server alone go to:

```
xxxx\projectTest\src\main\resources\hsqldb
```

And enter the following line:

```
java -cp ./lib/hsqldb.jar org.hsqldb.server.Server
```

Notice: the HSQLDB plug in has two errors presents in its folders, these errors do not interfere with the proper functioning of the application. These errors own of the HSQLDB package.

INSTALL AND START THE PROJECT ON ECLIPSE

To install the project the first thing is to recover it in the GitHub repository.
Through Git, clone the project to your Eclipse IDE.

`git clone https://github.com/alexiszakhar/technical.git`

with Maven :

`mvn clean install` (check Skip Tests)

After under Eclipse and on the title:

`Right click -> Maven -> Update Project...` and check the project name checkbox

Start to run the "`DaoApplicatinTests.java`" class that will add a new user and test the connection to the BDD by :

`Right click -> Run As -> JUnit Test`

After that run the "`ServicesApplicatinTests.java`" class that will also add a new user, who will serve in Postman rest tests.

To use the Postman calls, you have to import in your Postman application the "TechnicalFront.postman_collection" JSON file found in " `src/main/resources/postman_tests`" folder in the project application.

Before running the project and in a goal to get a log file operational, you have to change a path in "`logback.xml`" file. For me the path is:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <appender name="fileAppender" class="ch.qos.logback.core.FileAppender">
        <file>xxxx\projectTest\target\technical.log</file>
        <append>true</append>
        <encoder>
            <pattern>%d [%thread] %-5level  %logger{35} - %msg%n</pattern>
        </encoder>
    </appender>

    <root level="TRACE">
        <appender-ref ref="fileAppender" />
    </root>
</configuration>
```

So replace this by your path.

Then run the application by launching the server with a Maven action:

`mvn spring-boot:run` (don't forget to check Skip Tests case).

TESTING CONTROLLER

To test the controller class, you can find all the tests about this on:

`xxxx\projectTest\src\test\resources\postman_tests`

INFORMATION ABOUT THE PROJECT CODE

It asks on the project to only save a French adult person. To check this it exist some libraries to treat dates, but a timestamp that represents to the nearest second a delta of 18 years is easier to calculate The majority of a person and embed less code in the project.