INSTALLATION PROCEDURE AND STARTING THE APLICATION (with Eclipse)

DESCRIPTION OF THE USAGE OF THE EMBEDDED DATABASE

Before launching the application the very first time, 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:

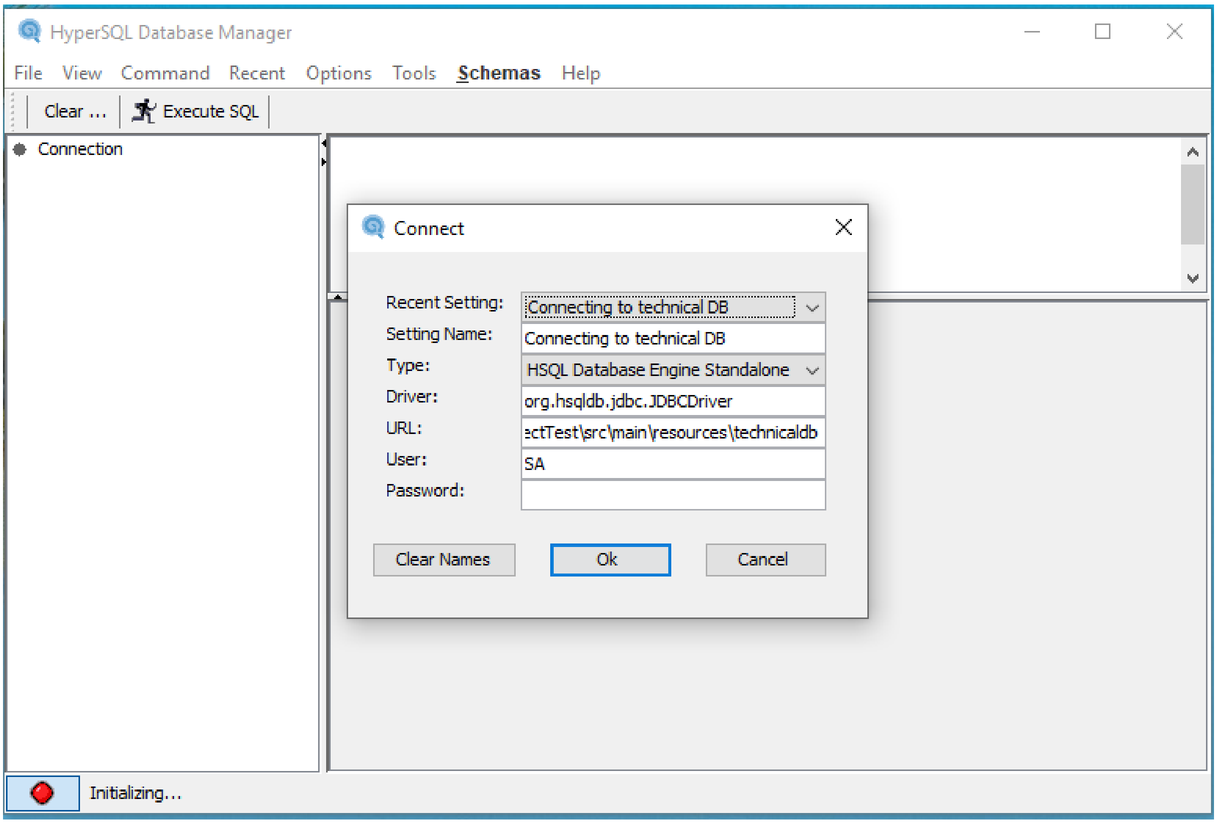
```
E:\projectTest\src\main\resources\hsqldb
```

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`"


HyperSQL Database Manager
File View Command Recent Options Tools Schemas Help
Clear ... Execute SQL
Connection
Connect
Clear Names Ok Cancel
Initializing...

Recent Setting: Connecting to technical DB
Setting Name: Connecting to technical DB
Type: HSQL Database Engine Standalone
Driver: org.hsqldb.jdbc.JDBCDriver
URL: ɛctTest\src\main\resources\technicaldb
User: SA
Password:

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

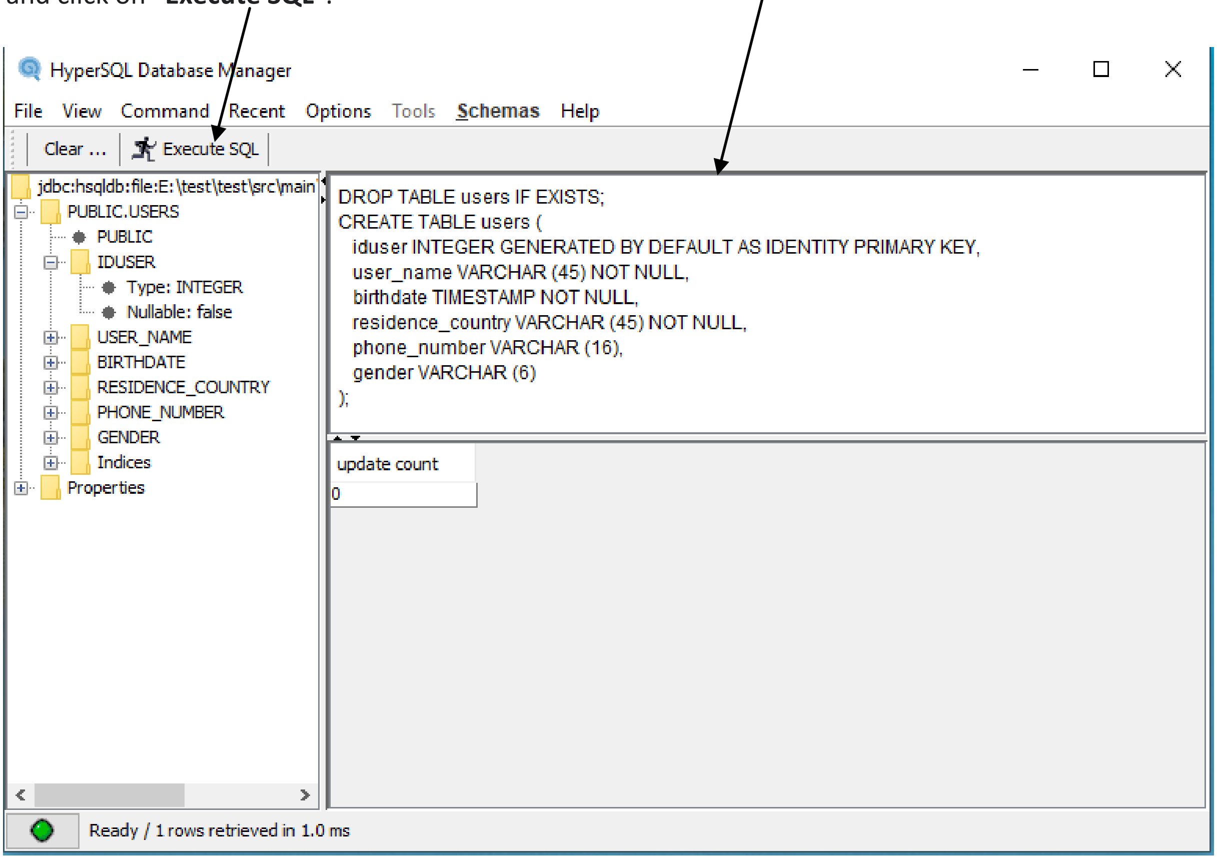
Location of the database is now at my place:

`E:\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**".



```
DROP TABLE users IF EXISTS;
CREATE TABLE users (
  iduser INTEGER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
  user_name VARCHAR (45) NOT NULL,
  birthdate TIMESTAMP NOT NULL,
  residence_country VARCHAR (45) NOT NULL,
  phone_number VARCHAR (16),
  gender VARCHAR (6)
);
```

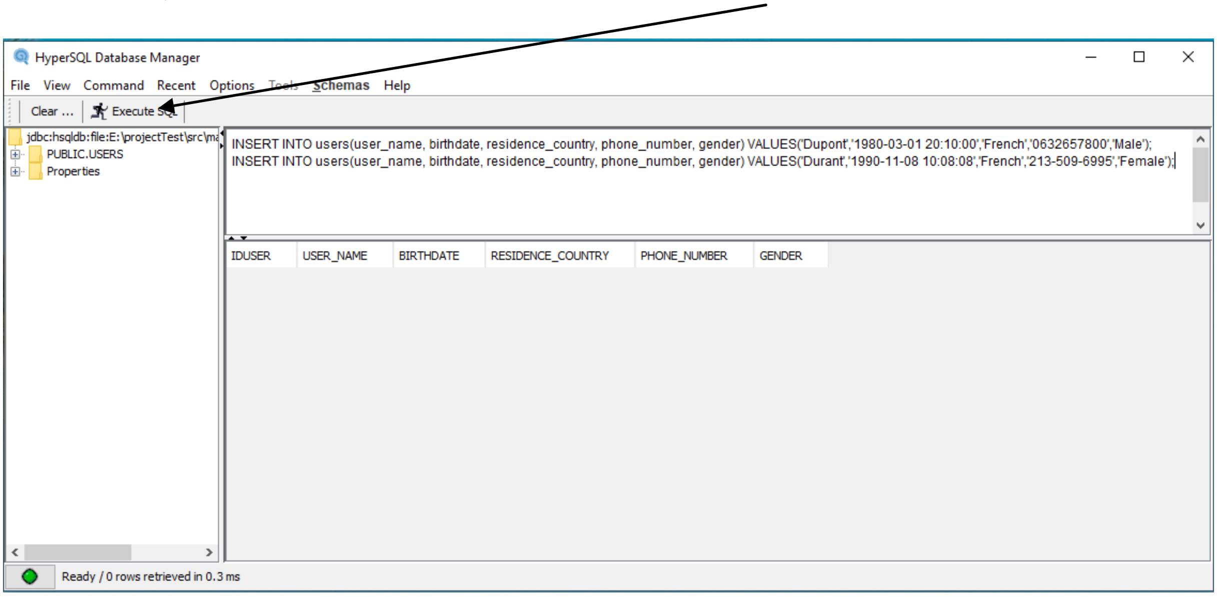
update count
0

Ready / 1 rows retrieved in 1.0 ms

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**".


HyperSQL Database Manager
File View Command Recent Options Tools Schemas Help
Clear ... Execute SQL
jdbc:hsqldb:file:E:\projectTest\src\ma
PUBLIC.USERS
Properties
Ready / 0 rows retrieved in 0.3 ms

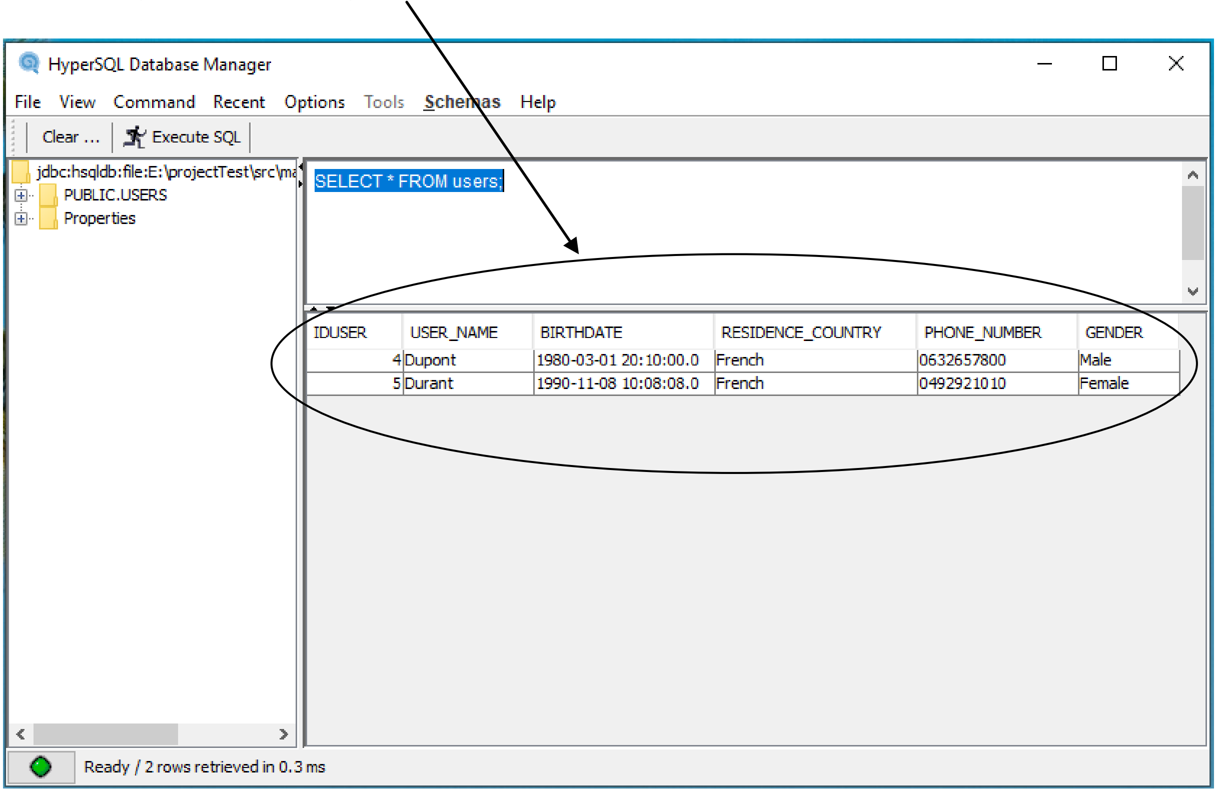
```
INSERT INTO users(user_name, birthdate, residence_country, phone_number, gender) VALUES('Dupont','1980-03-01 20:10:00','French','0632657800','Male');
INSERT INTO users(user_name, birthdate, residence_country, phone_number, gender) VALUES('Durant','1990-11-08 10:08:08','French','213-509-6995','Female');
```

IDUSER	USER_NAME	BIRTHDATE	RESIDENCE_COUNTRY	PHONE_NUMBER	GENDER

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

```
SELECT * FROM users;
```

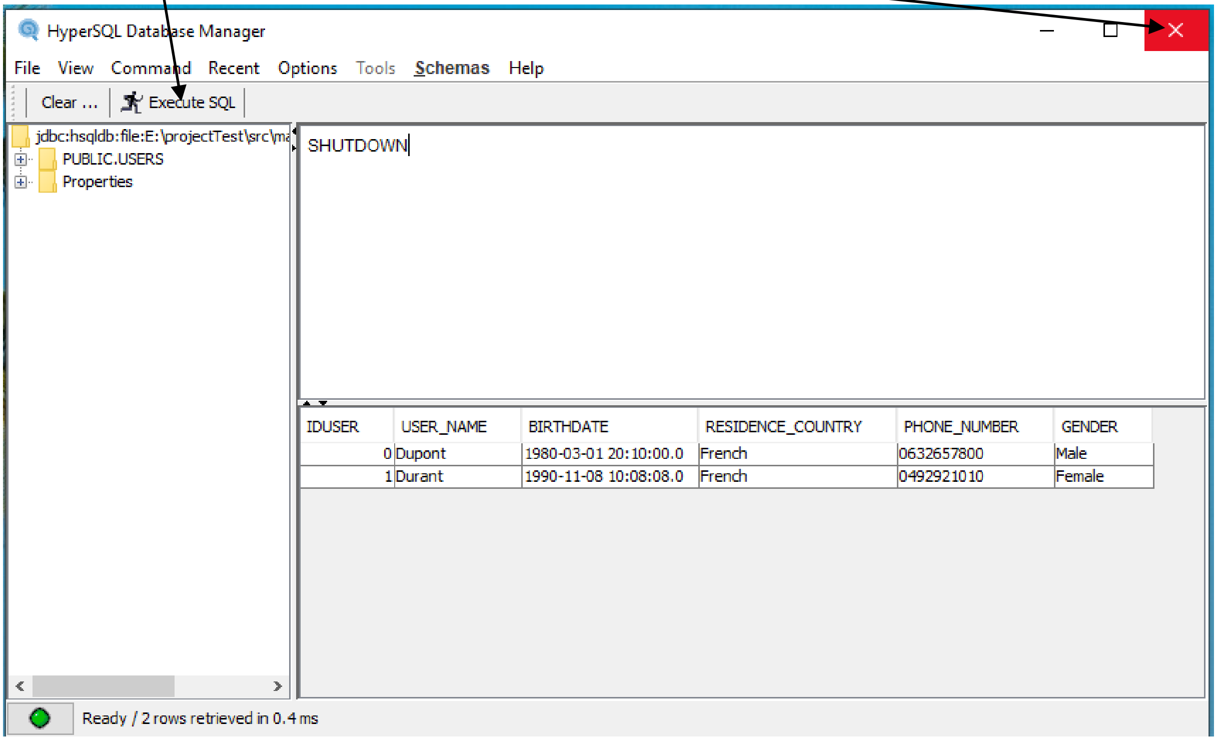
This action gives the following result:


HyperSQL Database Manager
File View Command Recent Options Tools Schemas Help
Clear ... Execute SQL
jdbc:hsqldb:file:E:\projectTest\src\ma
PUBLIC.USERS
Properties
Ready / 2 rows retrieved in 0.3 ms

```
SELECT * FROM users;
```

IDUSER	USER_NAME	BIRTHDATE	RESIDENCE_COUNTRY	PHONE_NUMBER	GENDER
4	Dupont	1980-03-01 20:10:00.0	French	0632657800	Male
5	Durant	1990-11-08 10:08:08.0	French	0492921010	Female

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.



```
SHUTDOWN
```

IDUSER	USER_NAME	BIRTHDATE	RESIDENCE_COUNTRY	PHONE_NUMBER	GENDER
0	Dupont	1980-03-01 20:10:00.0	French	0632657800	Male
1	Durant	1990-11-08 10:08:08.0	French	0492921010	Female

Ready / 2 rows retrieved in 0.4 ms

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:

```
E:\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:

```
E:\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:

```
E:\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:

```
E:\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>E:\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).