

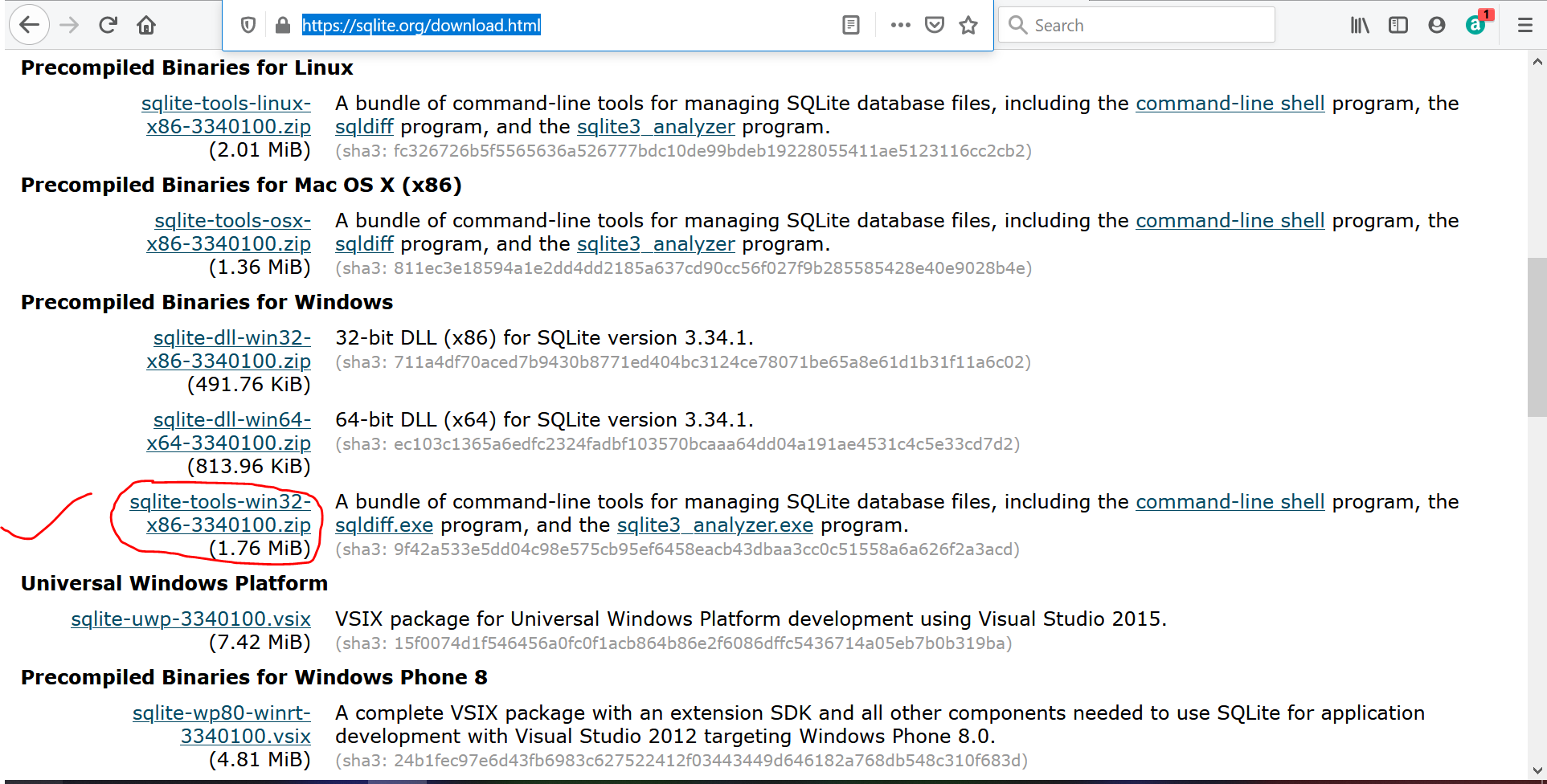
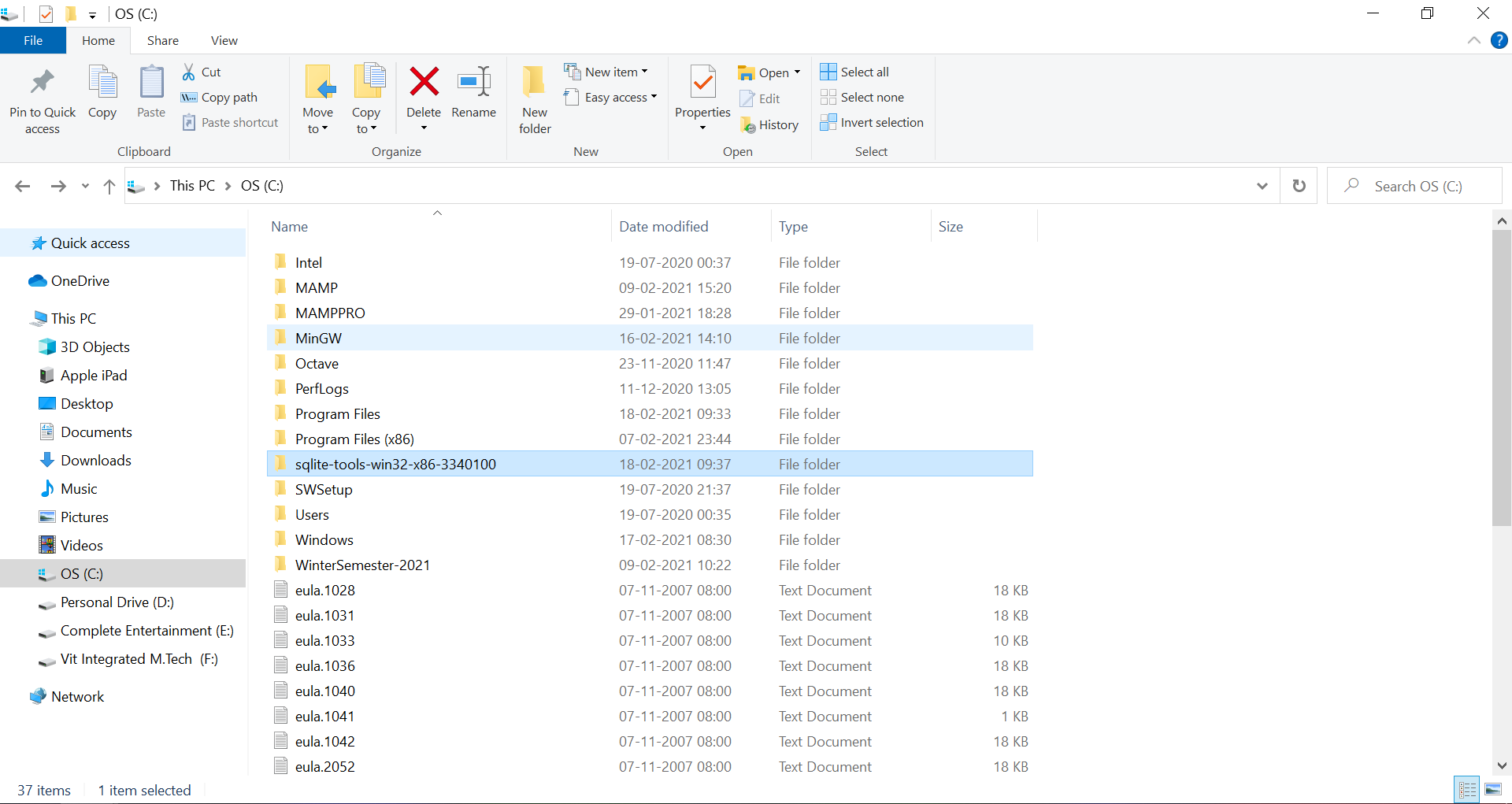
1st 🡪 driver  
2nd 🡪 connection  
3rd 🡪 statement  
4th 🡪 result

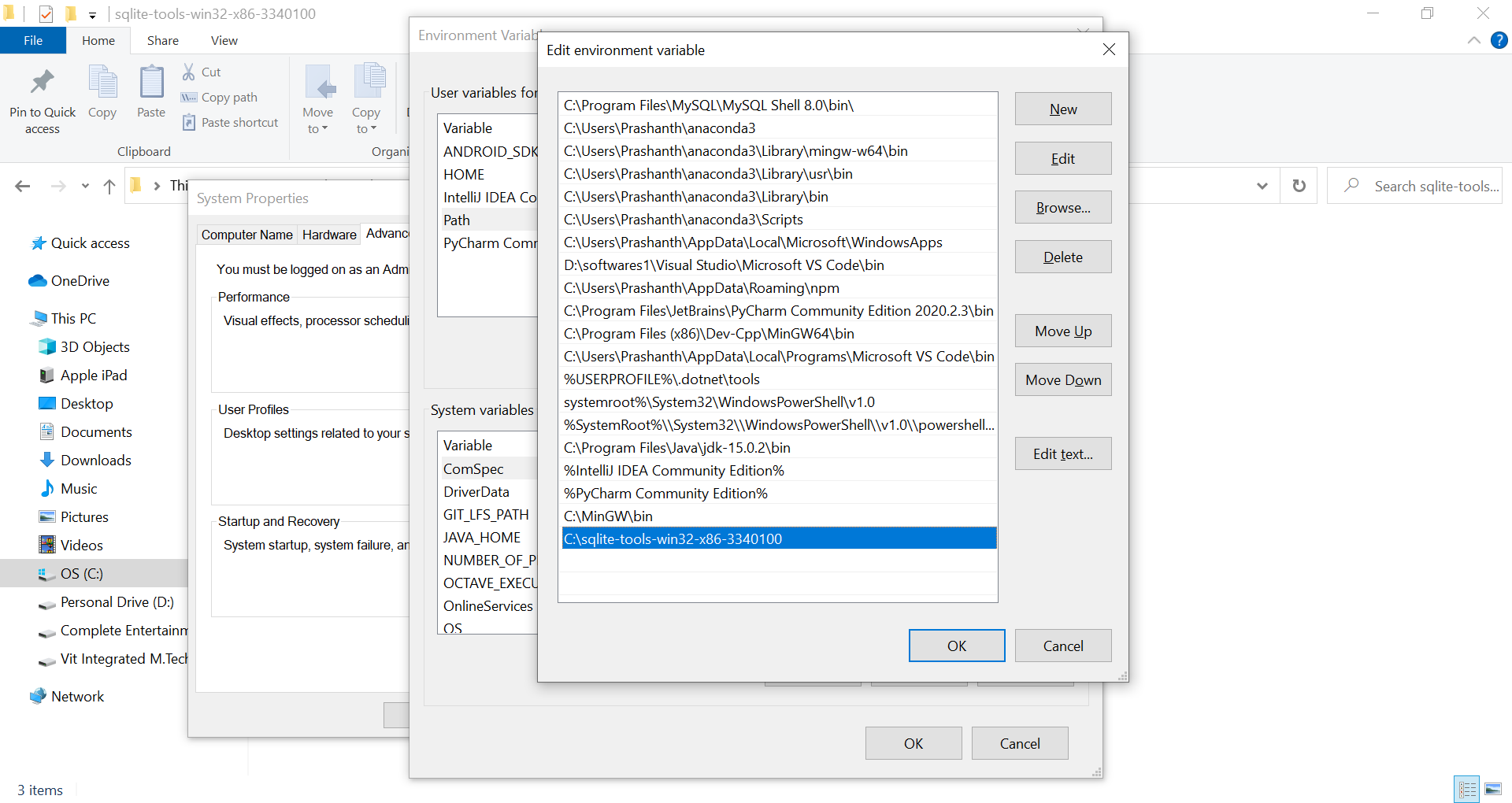
Driver 🡪 \* Helps in establishing a connection between the JAVA program and data-base. This can be done using API’s and function call.  
\* Conversion of data-types from the data-base to JAVA and vice-versa for mutual understanding.

Include sqlite-jdbc 3.20.1(driver) in IntelliJ 🡪  
<https://hacksmile.com/images-how-to-add-jar-files-in-intellij-idea/>

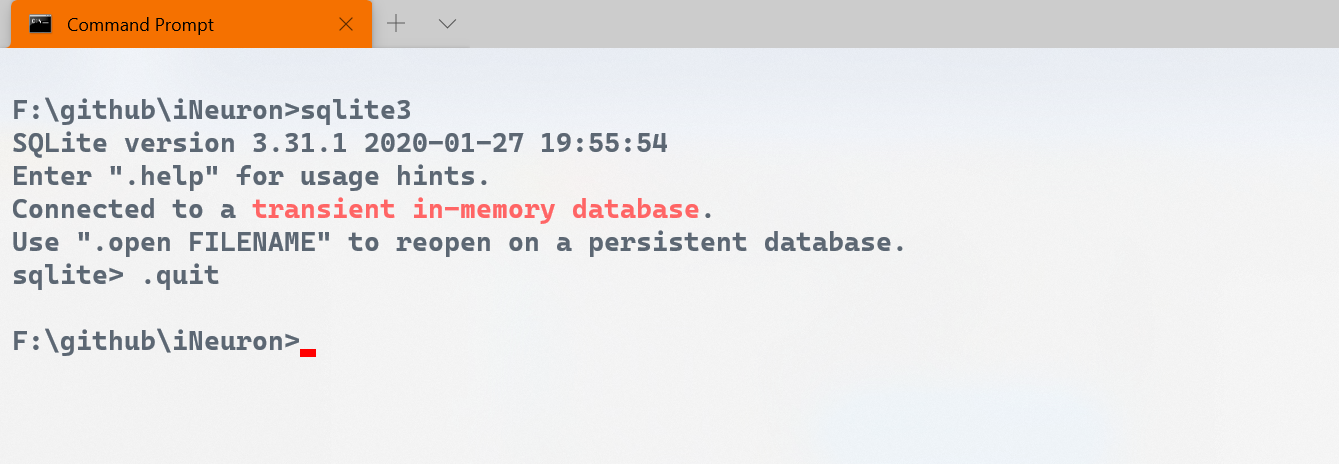
Installing JDBC

Installing sqlite  
<https://sqlite.org/download.html>

  
Unzip the downloaded file and Paste in C:drive  


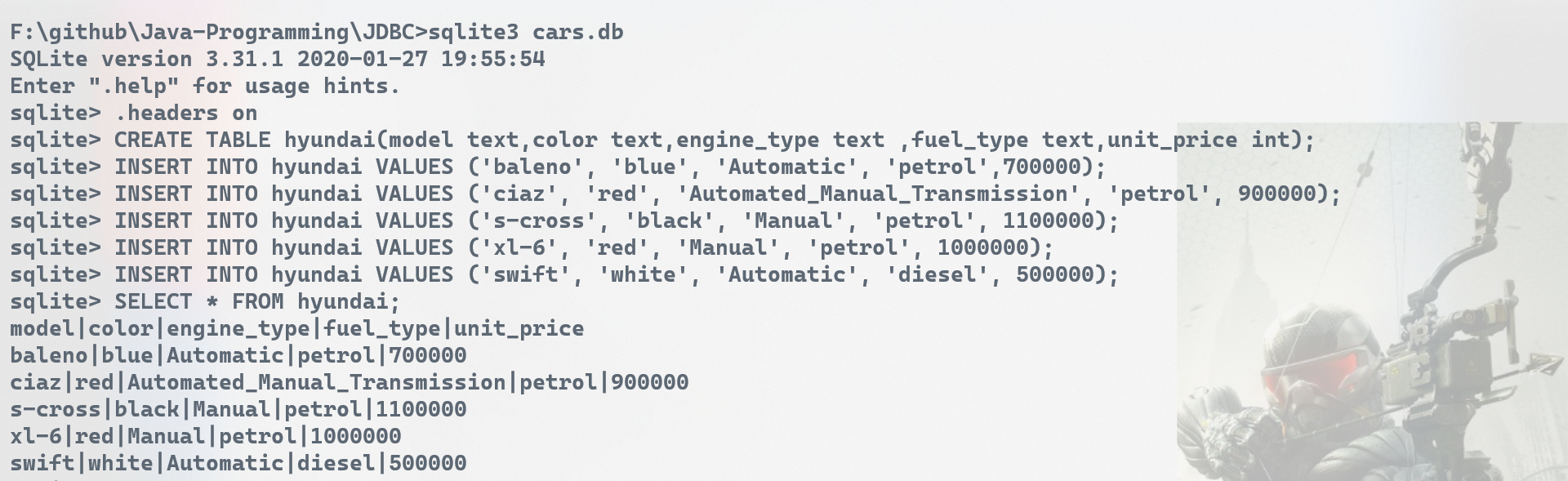
Add to the path variable  


See the Working in

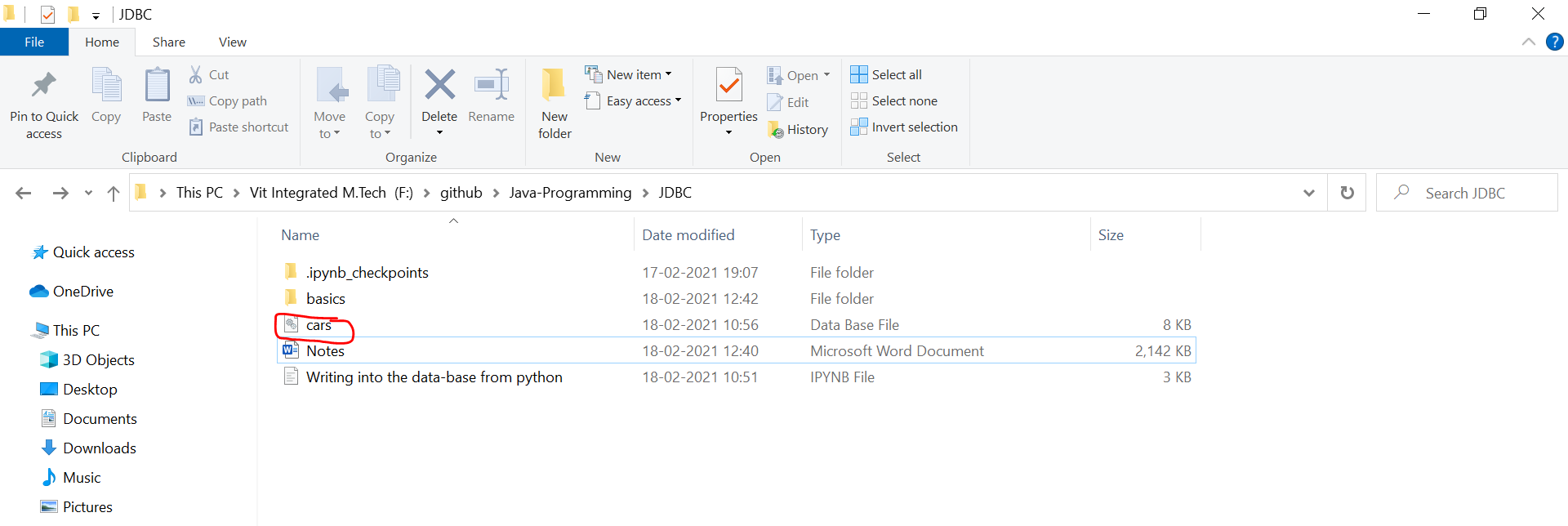


**Creating data-bases from sqlite3 command line**

Table-name Data-base name

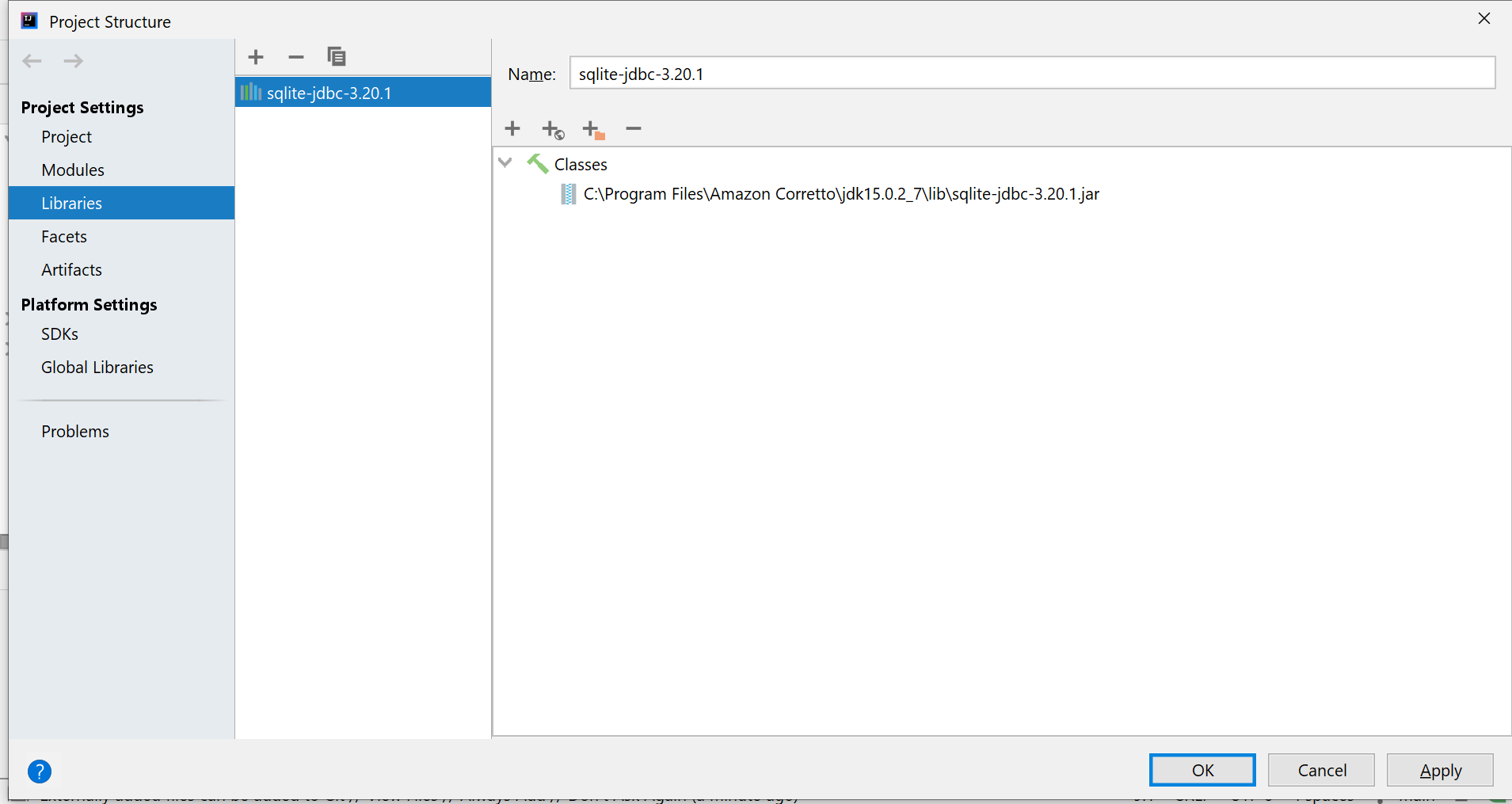






**Creating data-bases from Java**

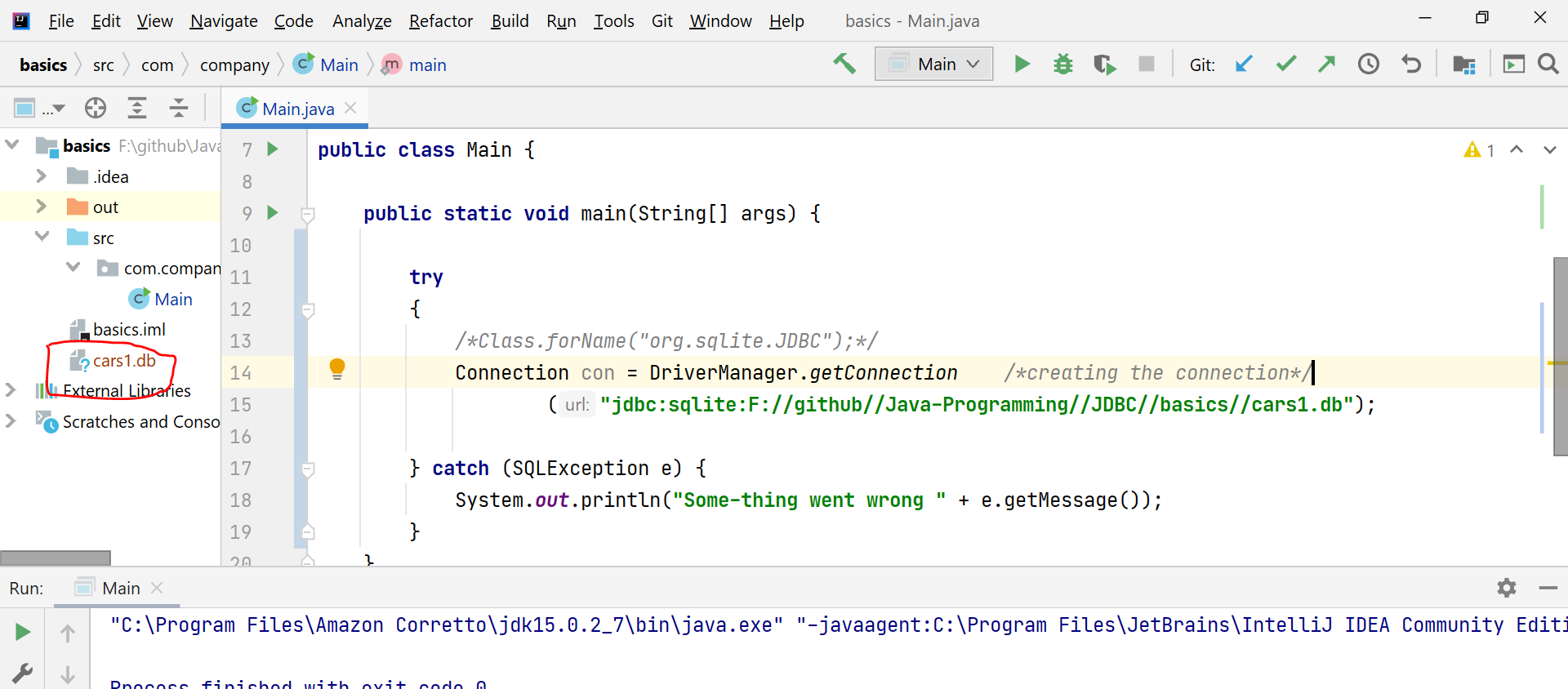
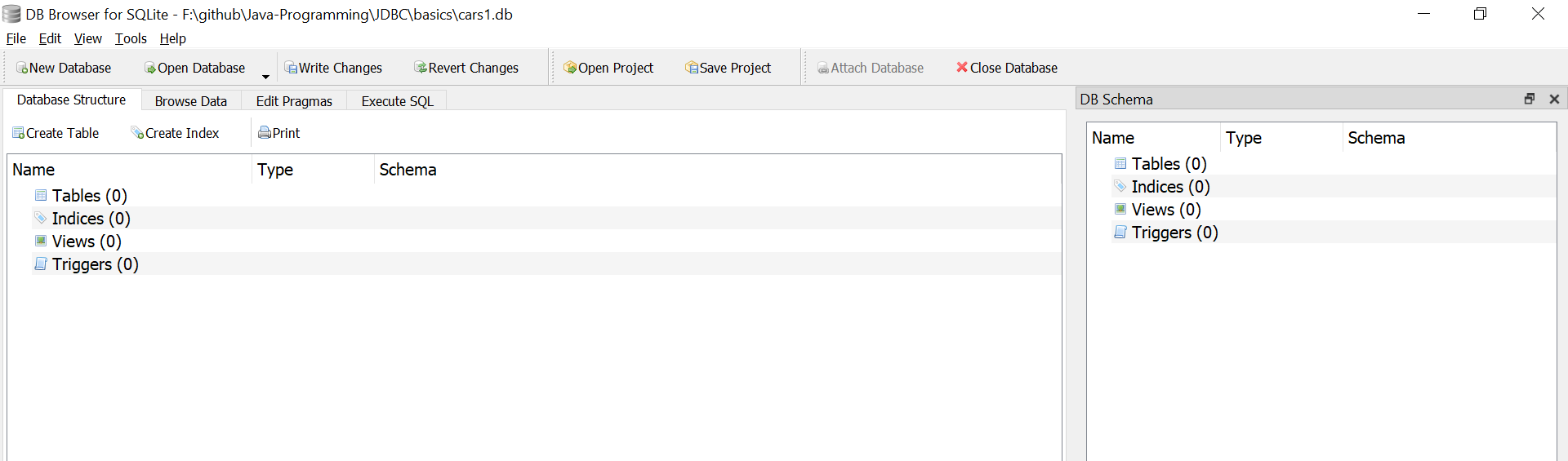
1) Add java-driver into our project.  
Inside the project, give file 🡪 Project structure, Click Libraries under Project Settings



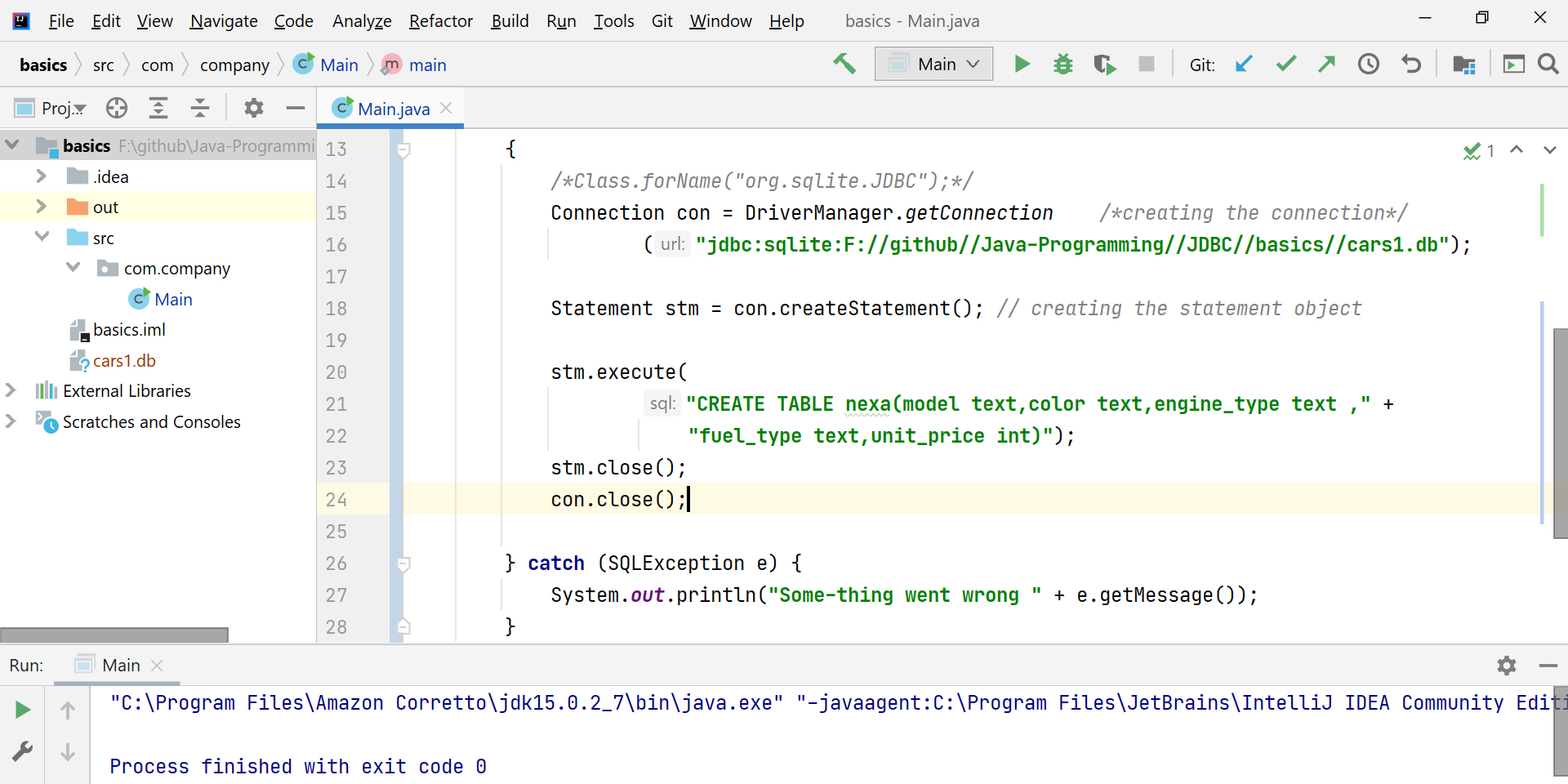
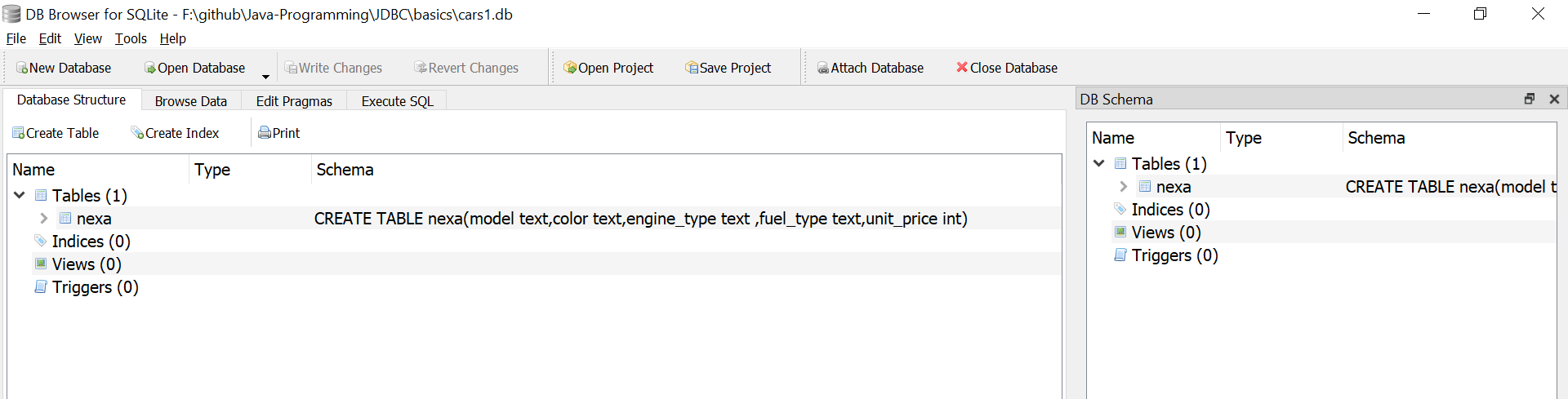
Click Apply and then OK

Now, we are going to apply the CRUD [Create Read Update Delete] operations.

Data-base creation

  
After running the main, cars1.db file is created   


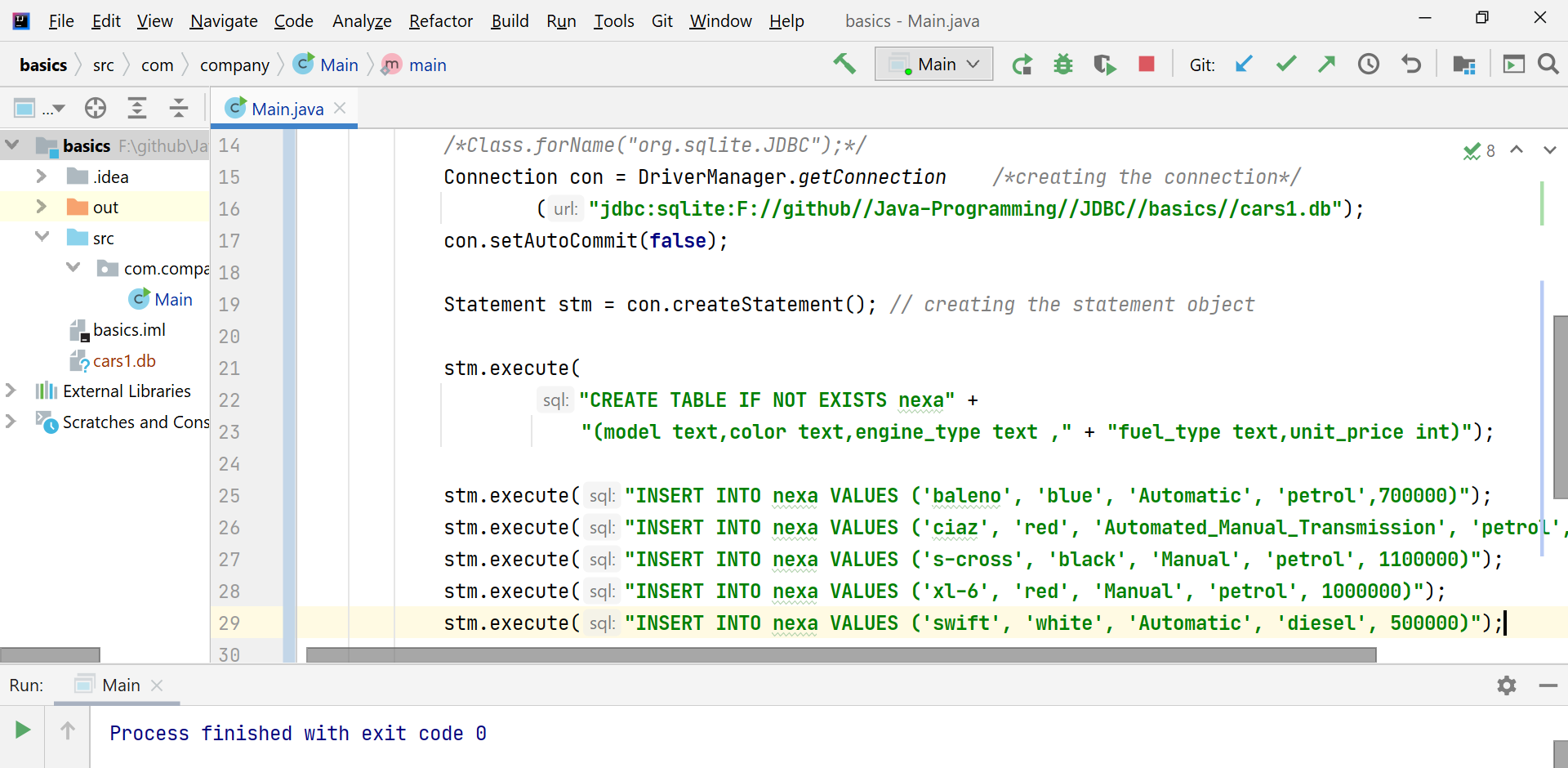
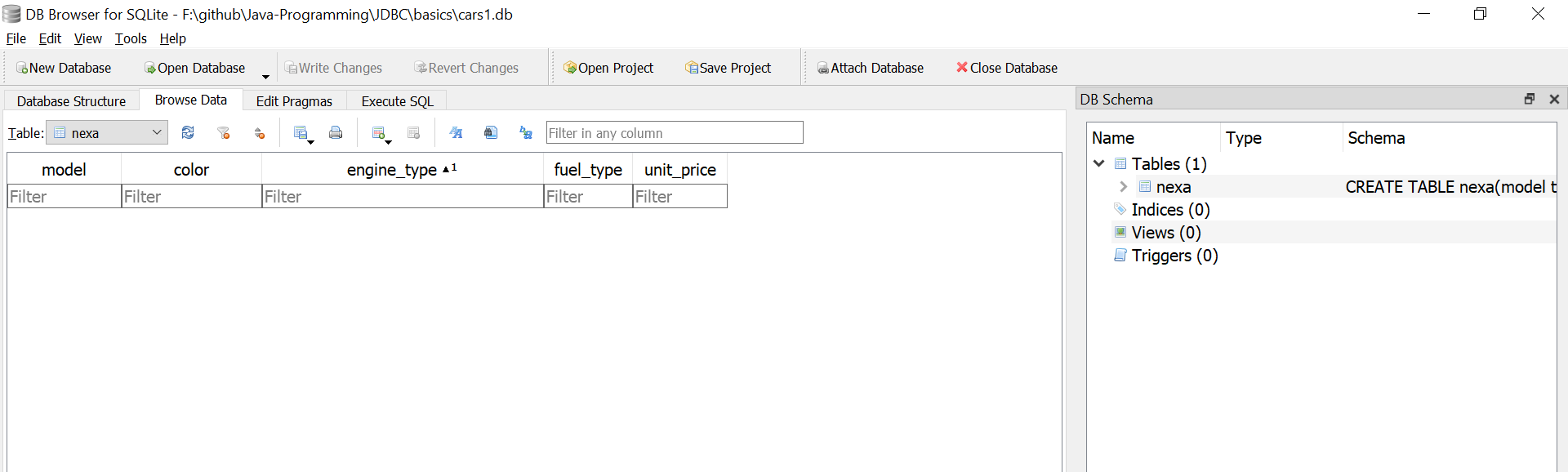
cars1 table is viewed in db browser sqlite.

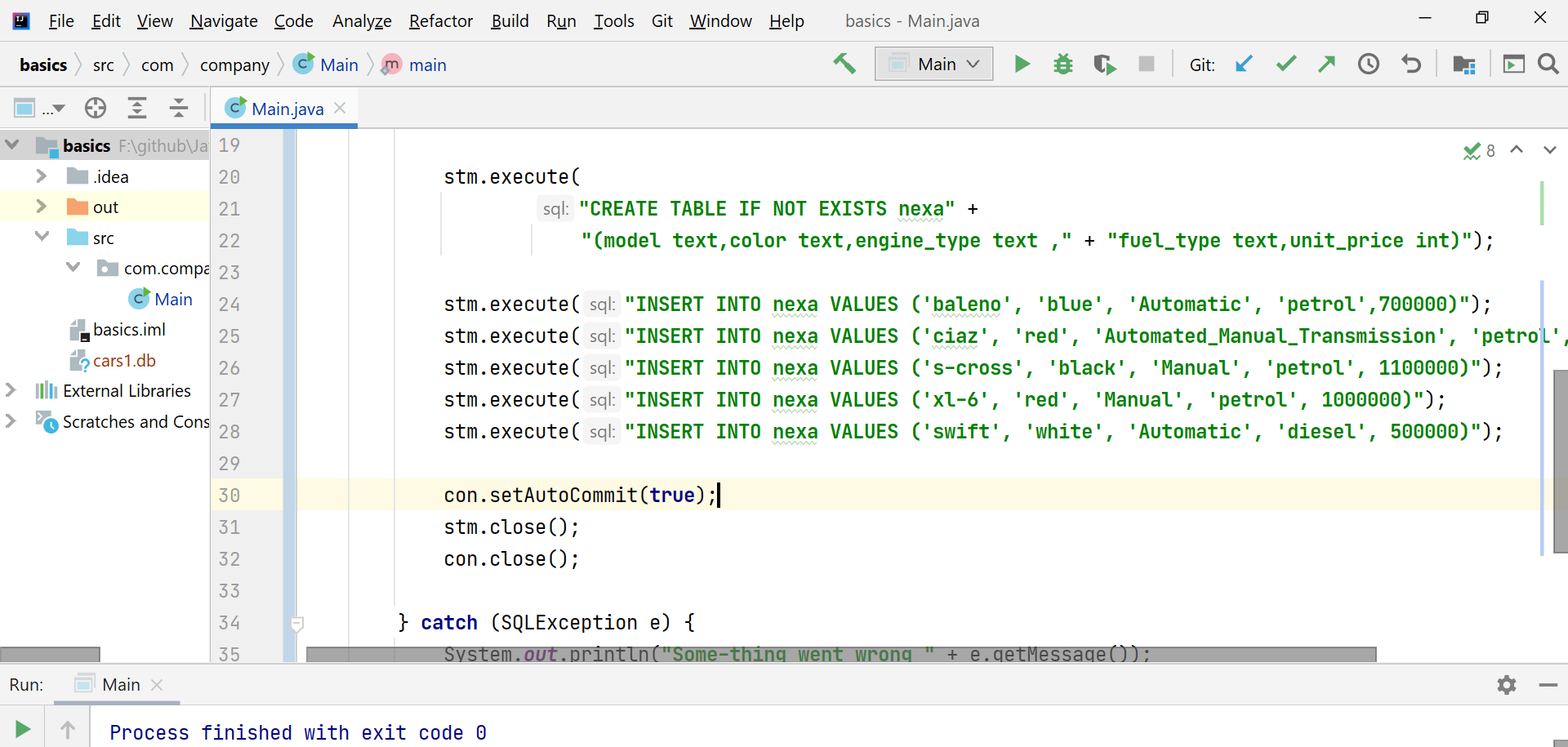
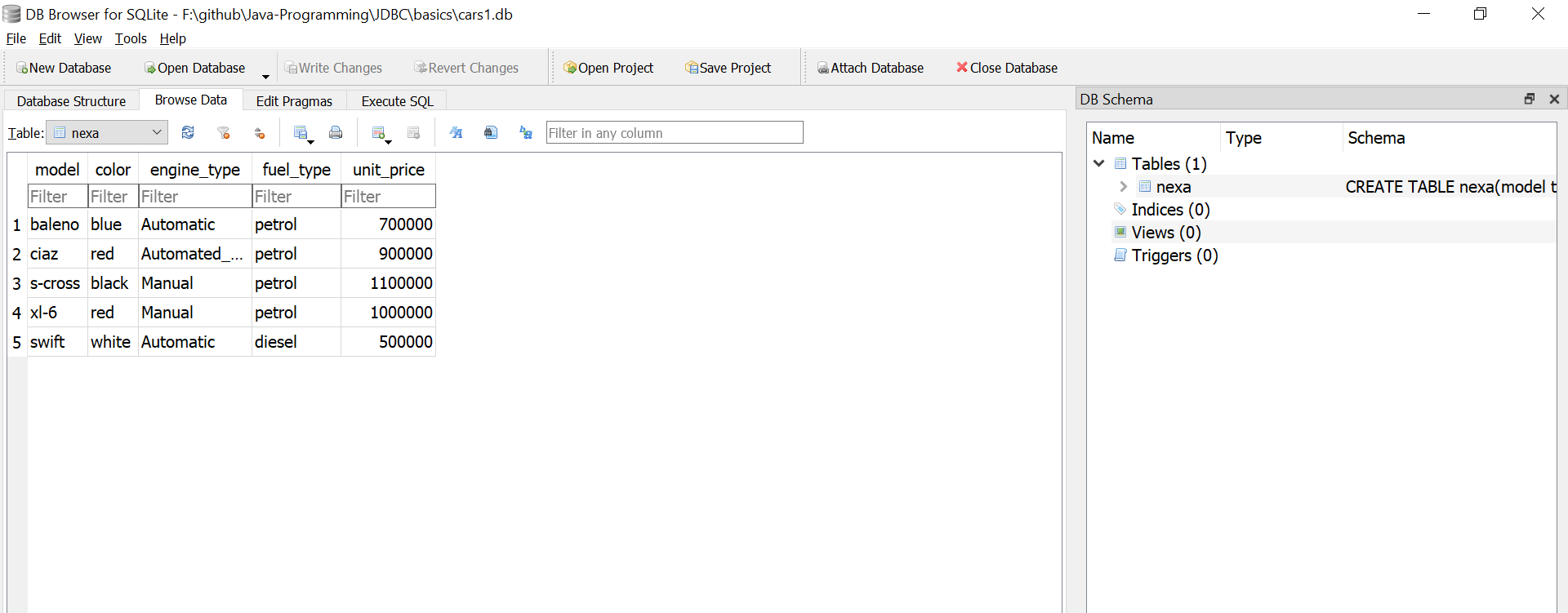
**C** 🡪 Table creation  
  


If we run this program, once again will get an error “table already exists”.

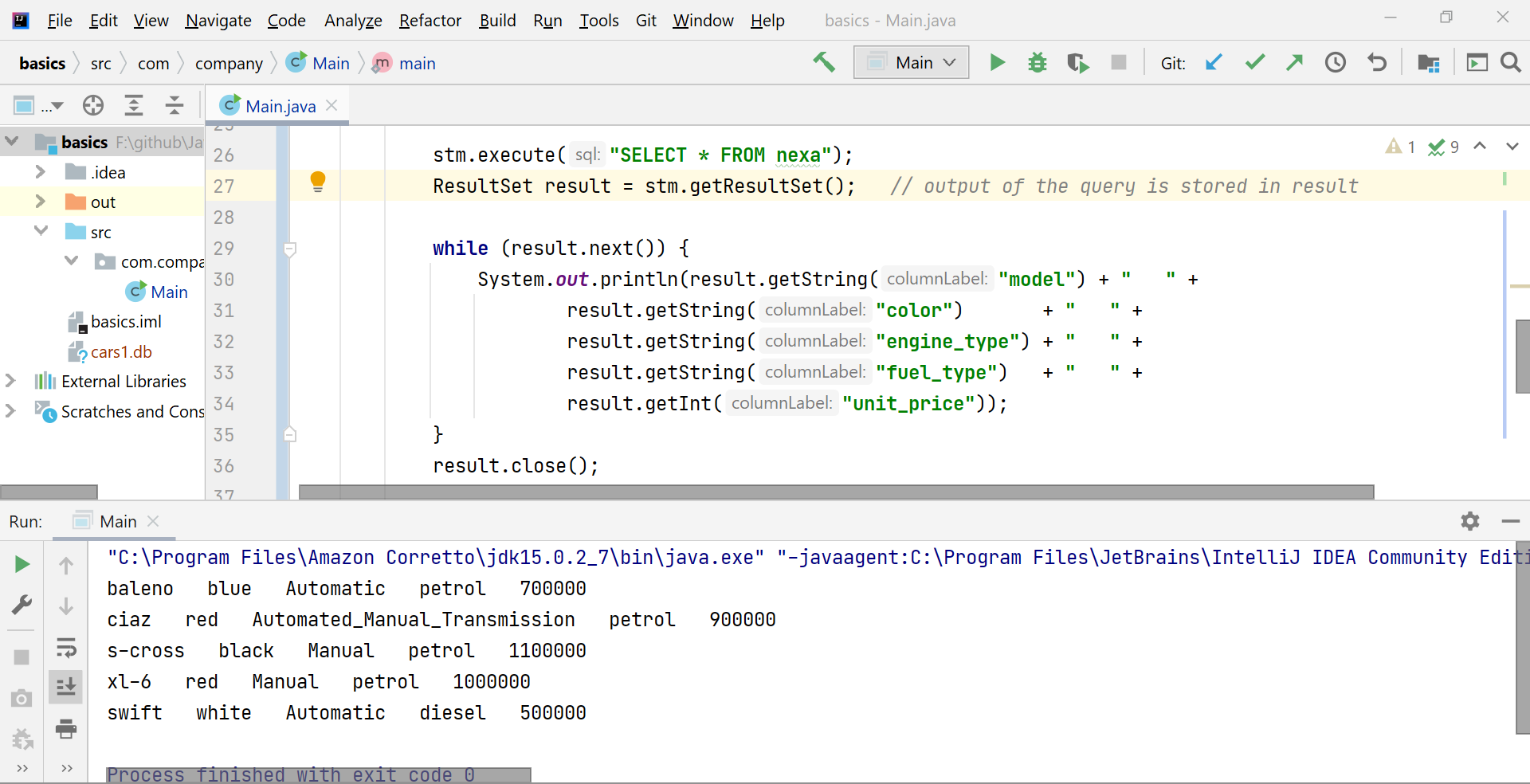
In-order to avoid that,

stm.execute(‘CREATE TABLE IF NOT EXISTS nexa”)

Inserting the values into the data-base by auto-committing(false)  


Inserting the values into the data-base by auto-committing(true)  
  


**D** 🡪 Displaying the results



Using “statement.executeQuery()”

