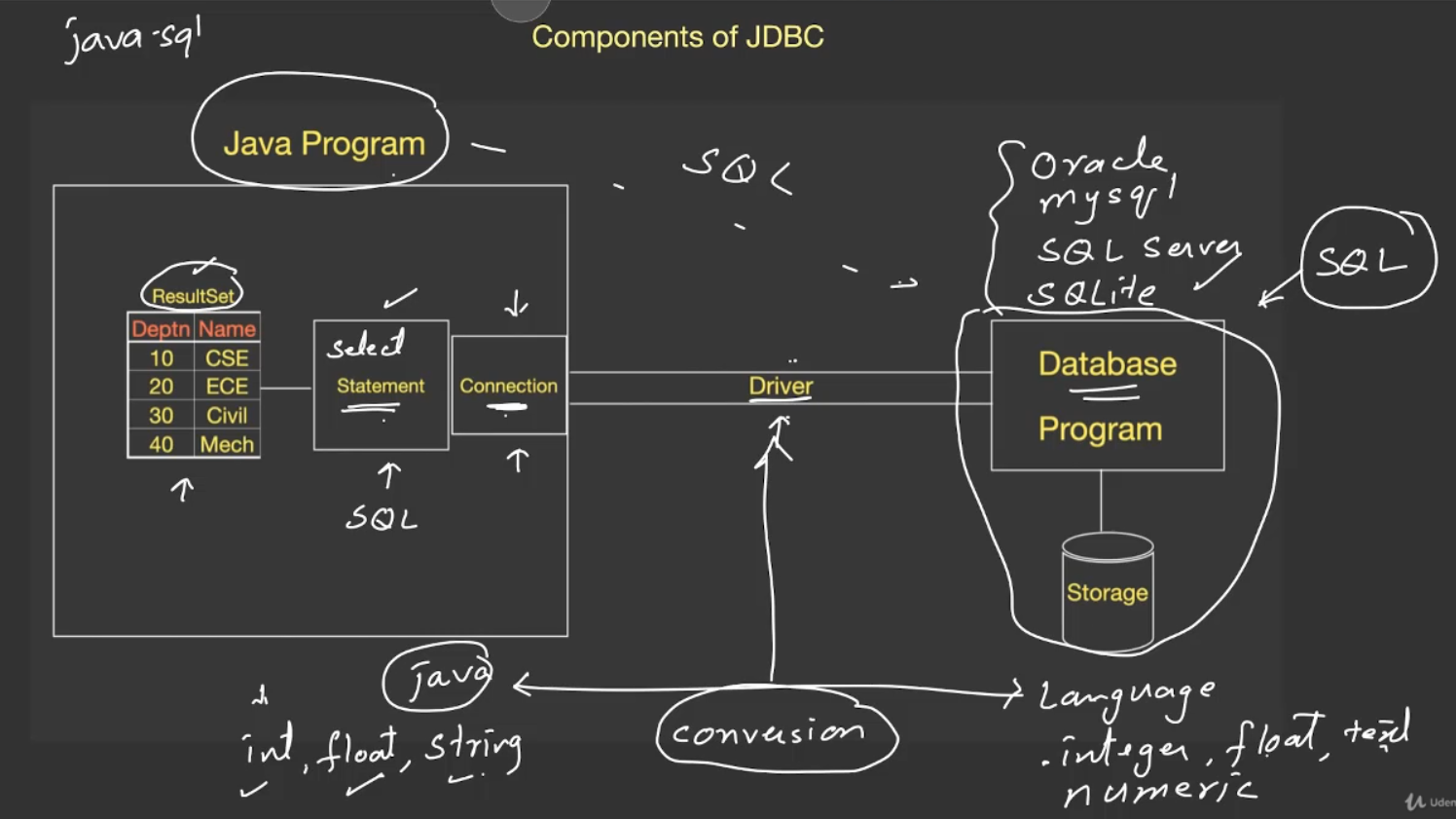
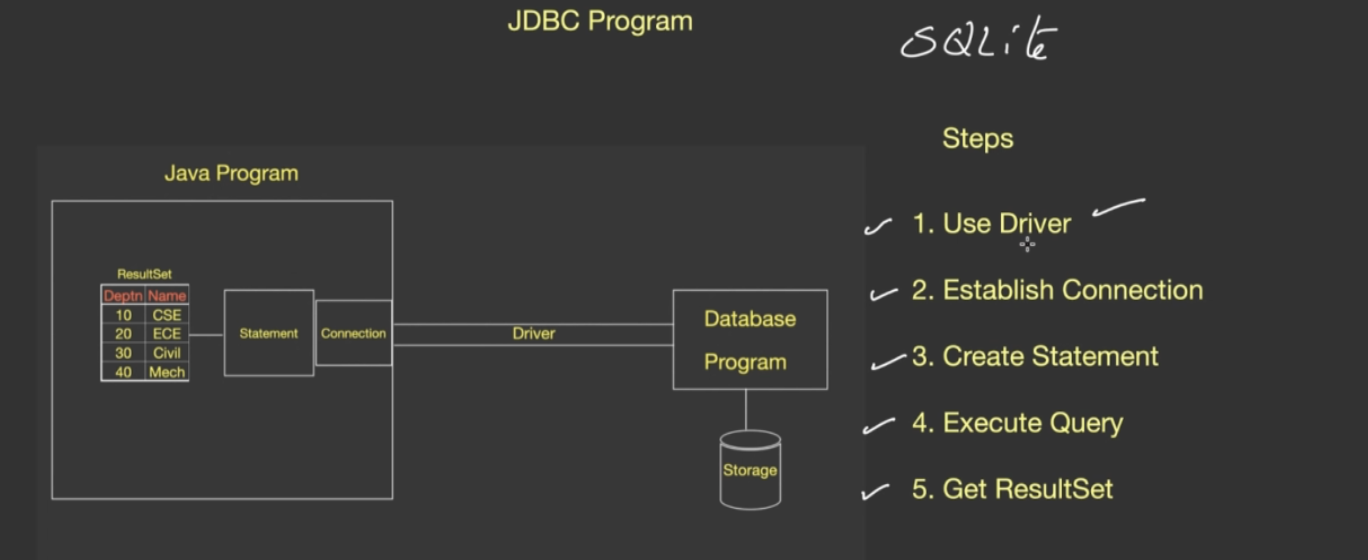
# **Components of JDBC**



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.

# **Initial Steps for JDBC program**

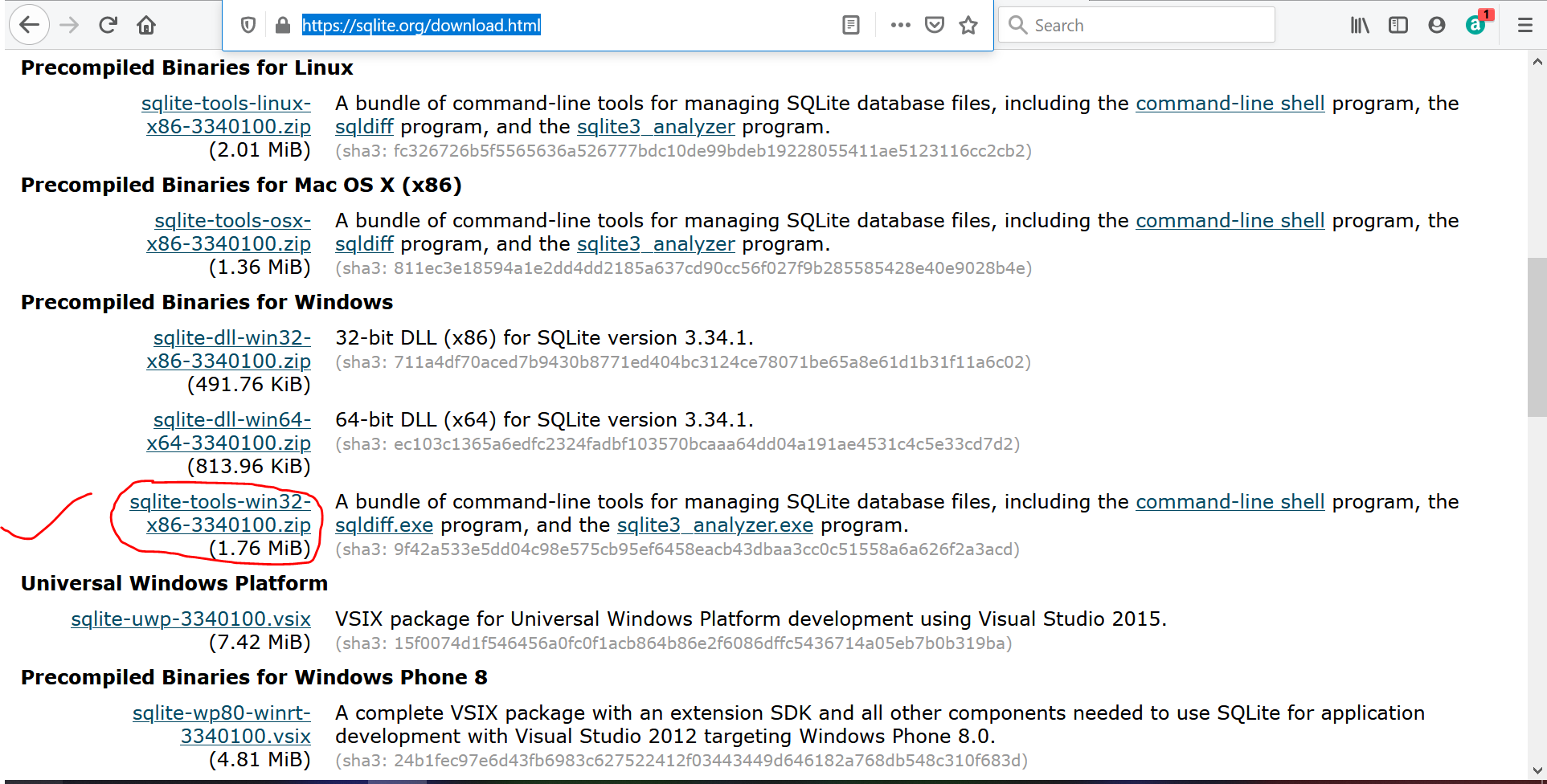
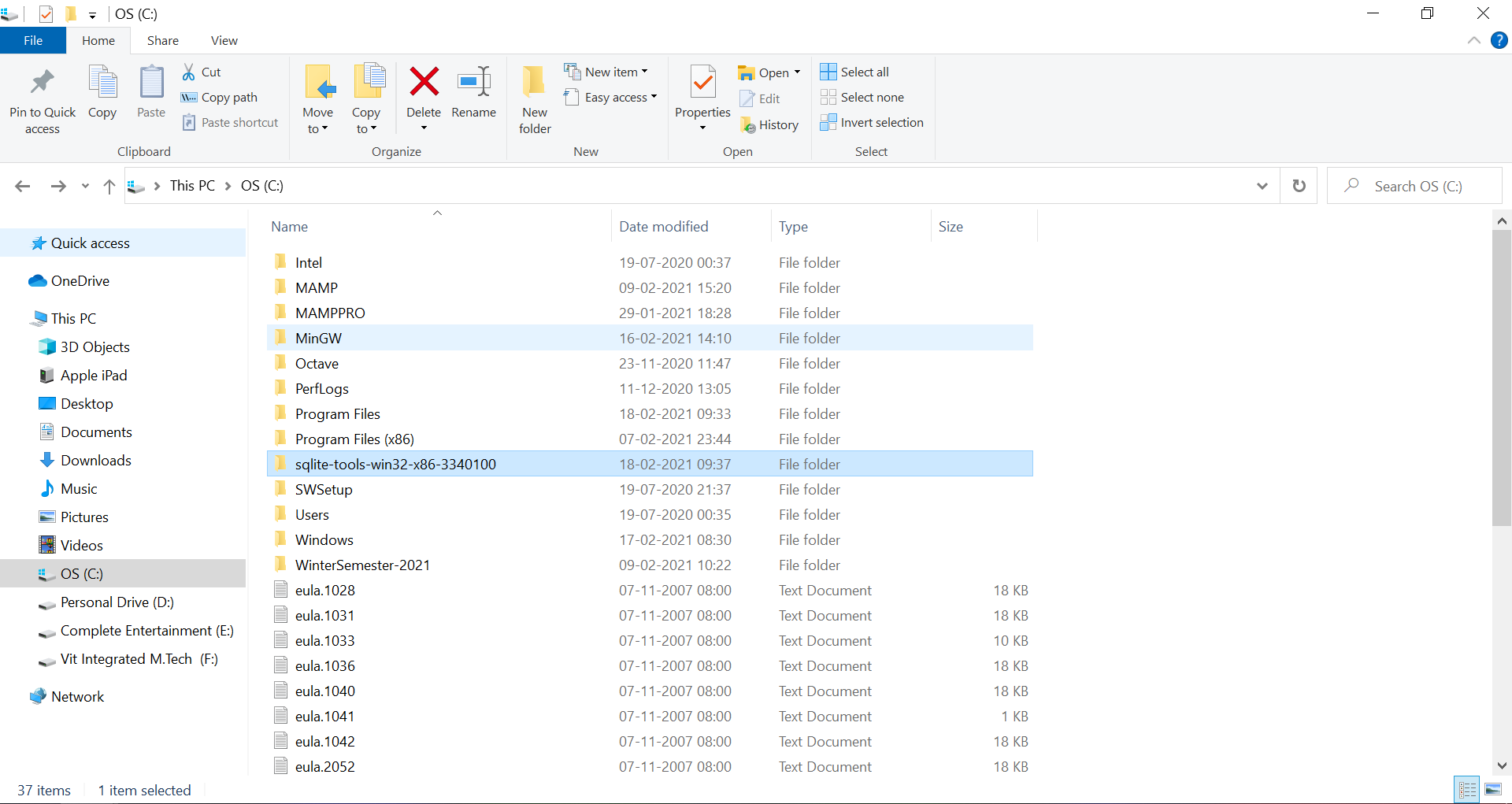


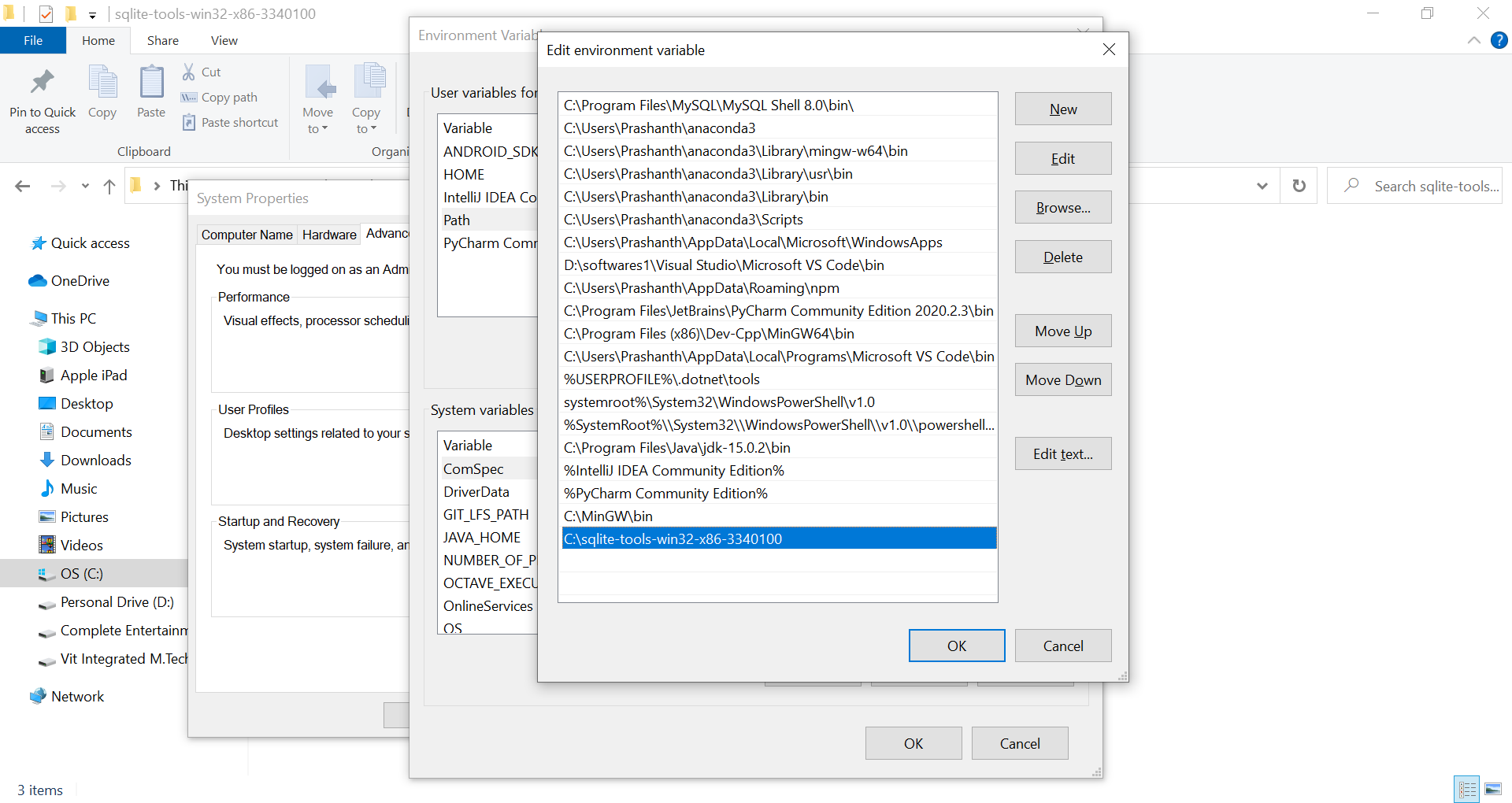
sqlite-jdbc driver 🡪 http://www.java2s.com/Code/JarDownload/sqlite/sqlite-jdbc-3.7.2.jar.zip

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

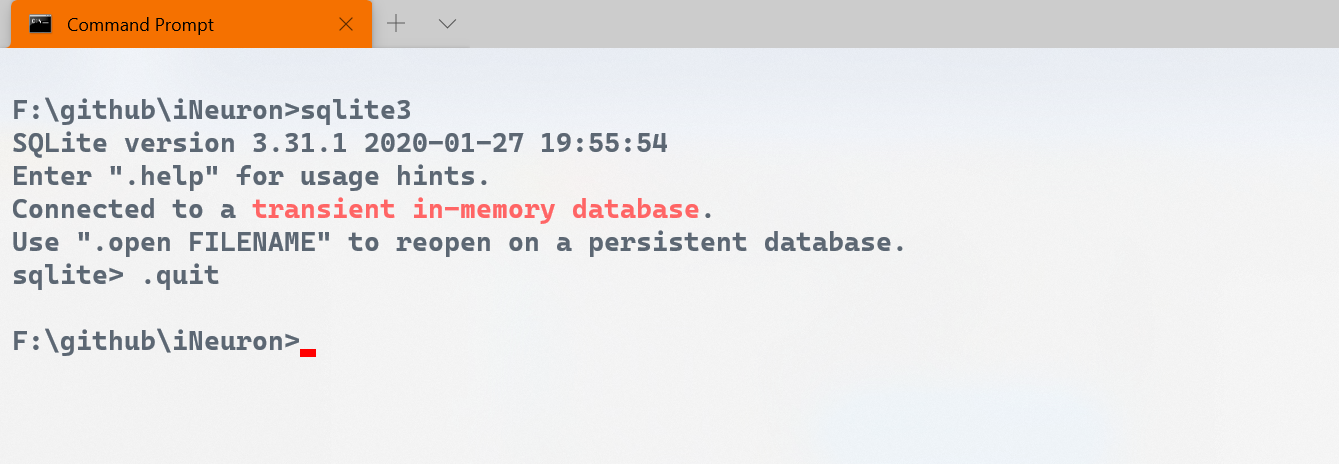
Installing JDBC

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

  
Unzip the downloaded file and Paste in C:drive  


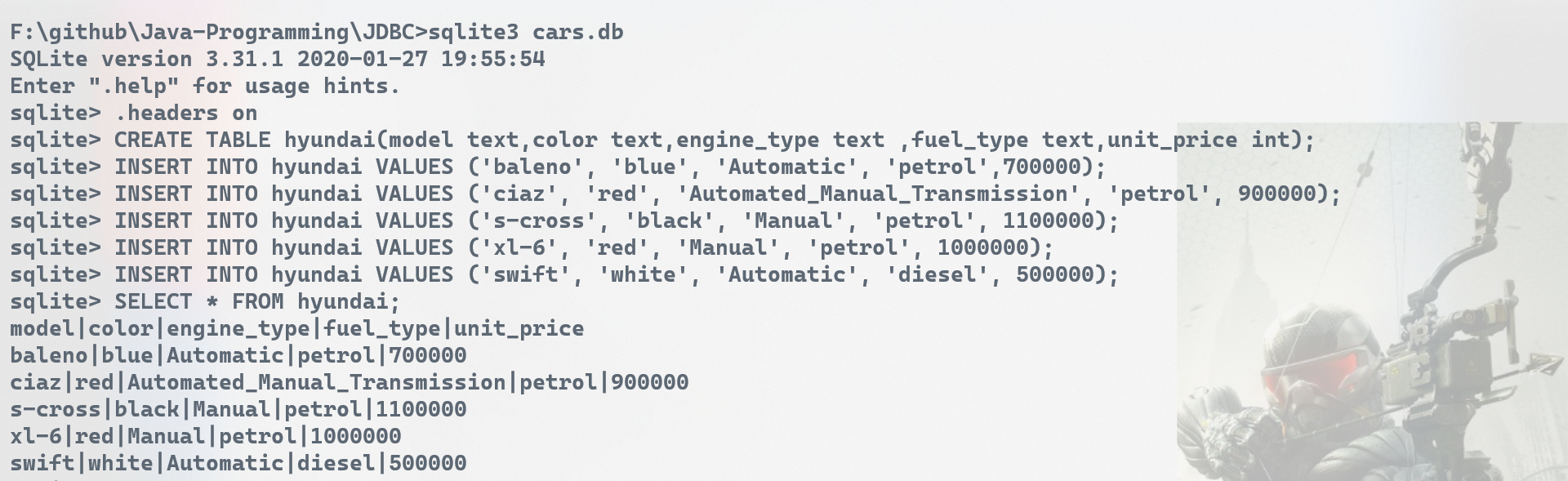
Add to the path variable  


See the Working in

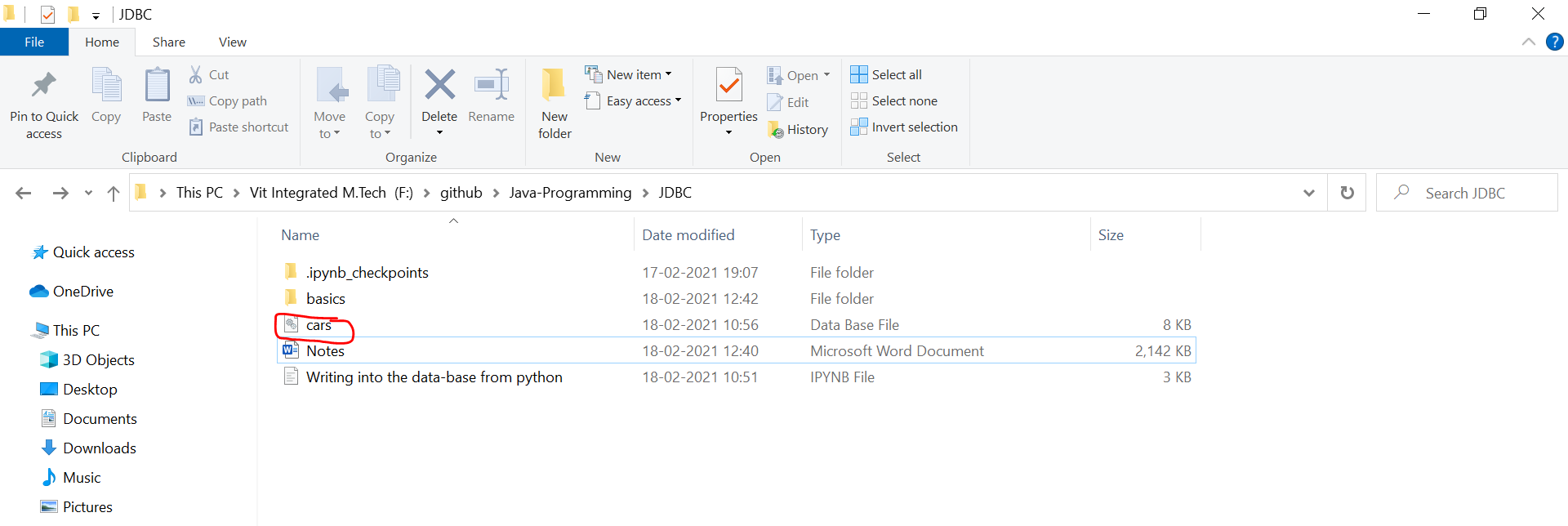


**Creating data-bases from sqlite3 command line**

Table-name Data-base name







**Creating data-bases from Java**

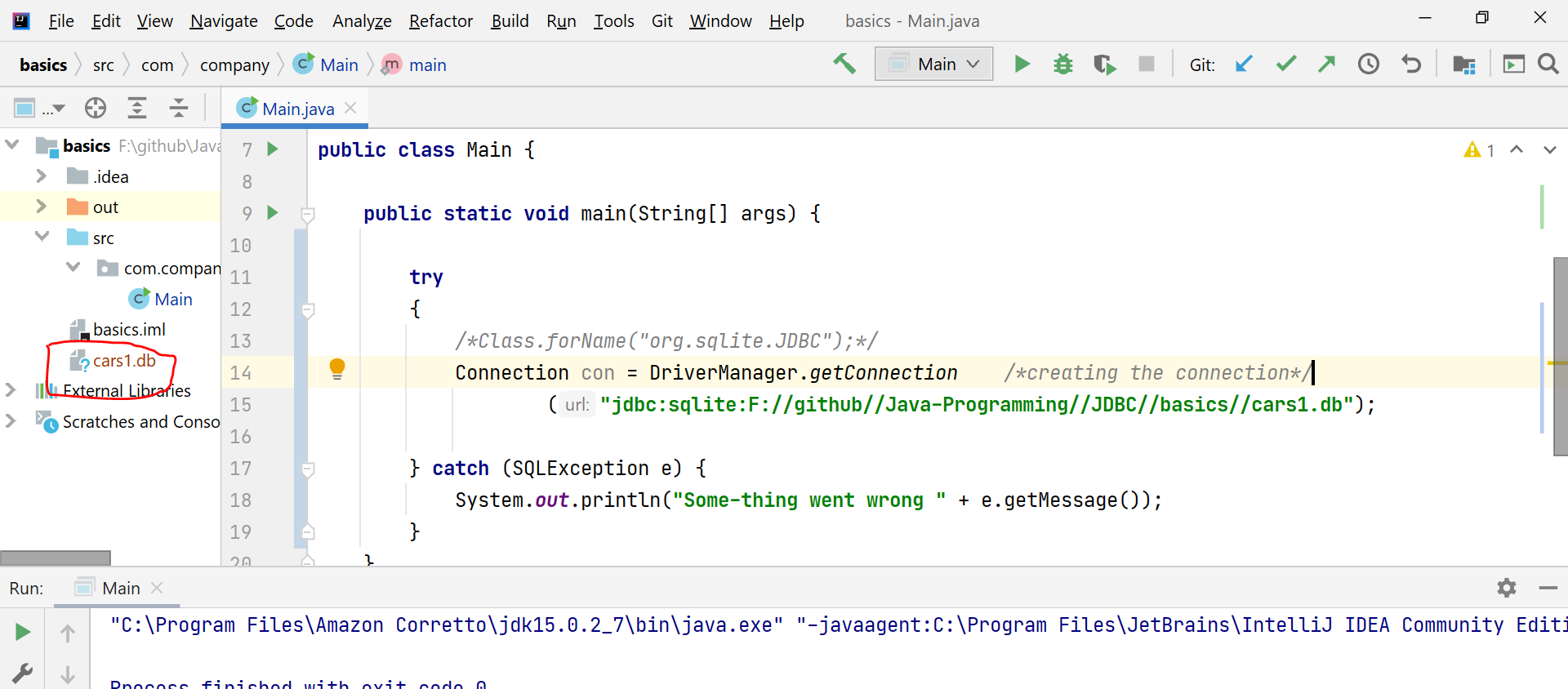
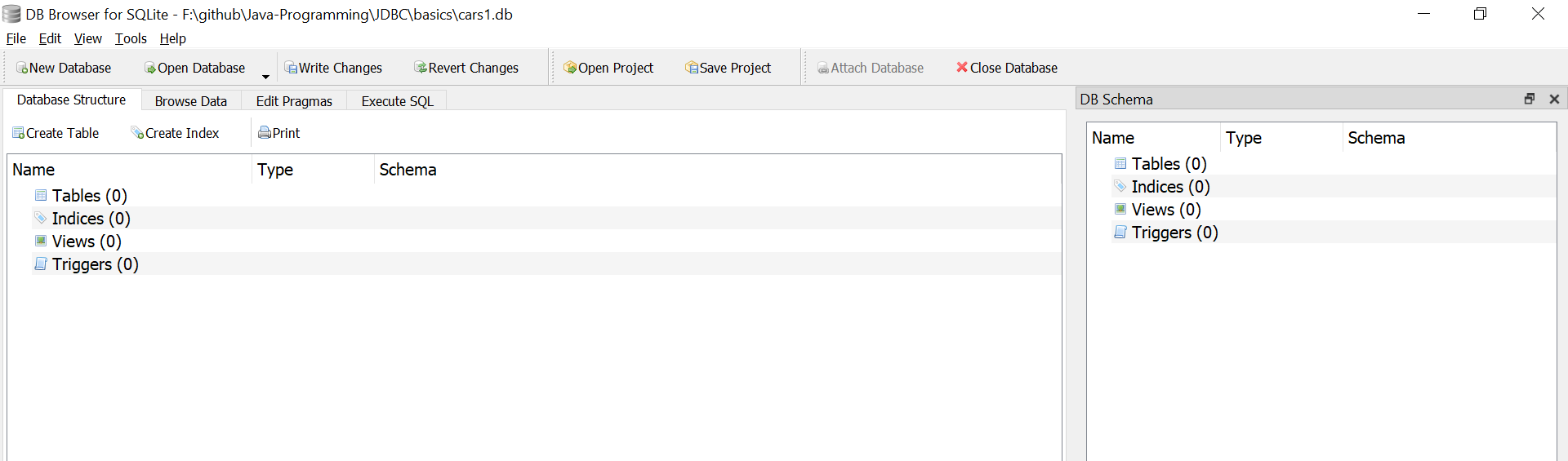
1) Add java-driver into our project.  
Inside the project, give file 🡪 Project structure, Click Libraries under Project Settings



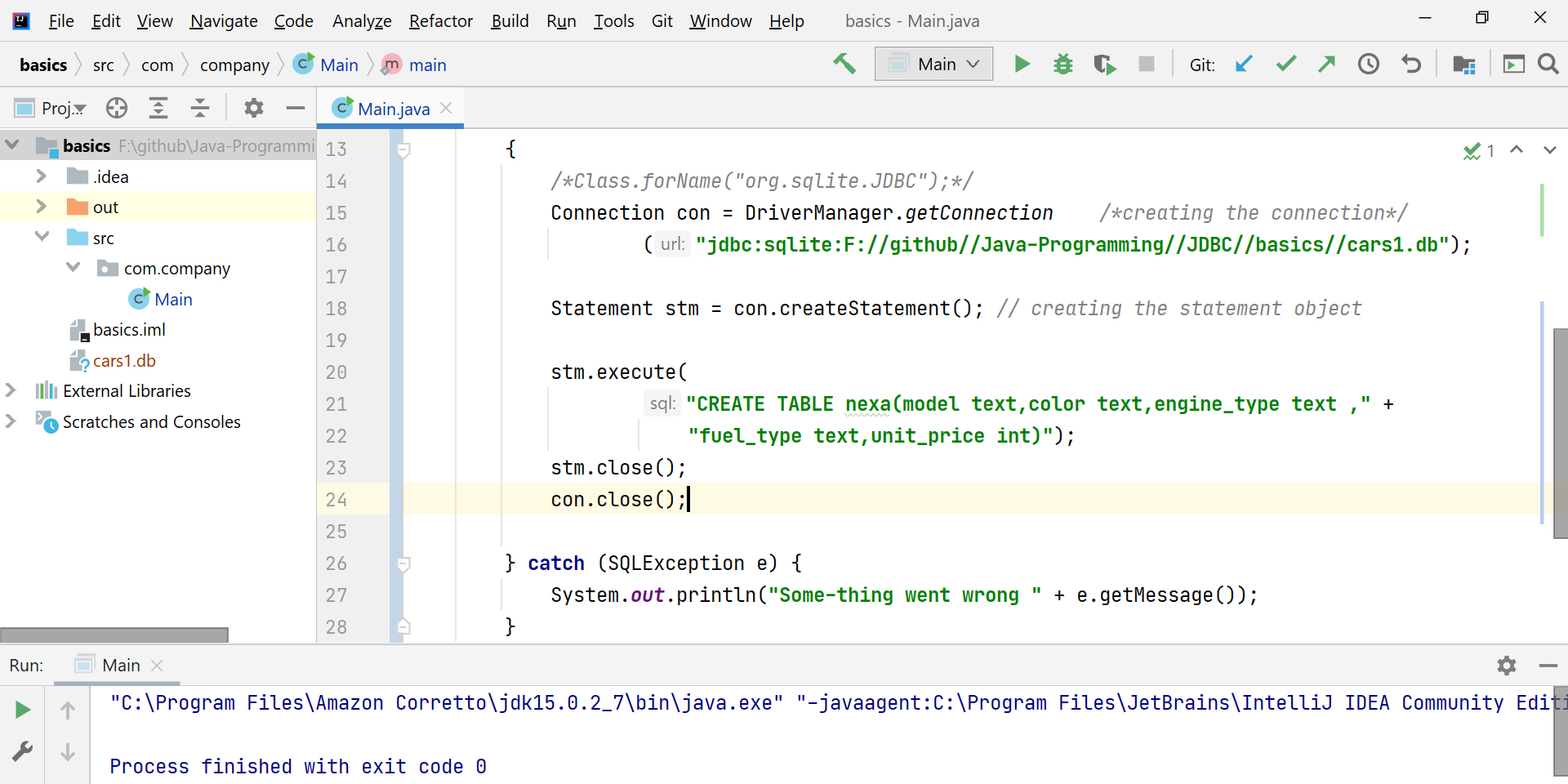
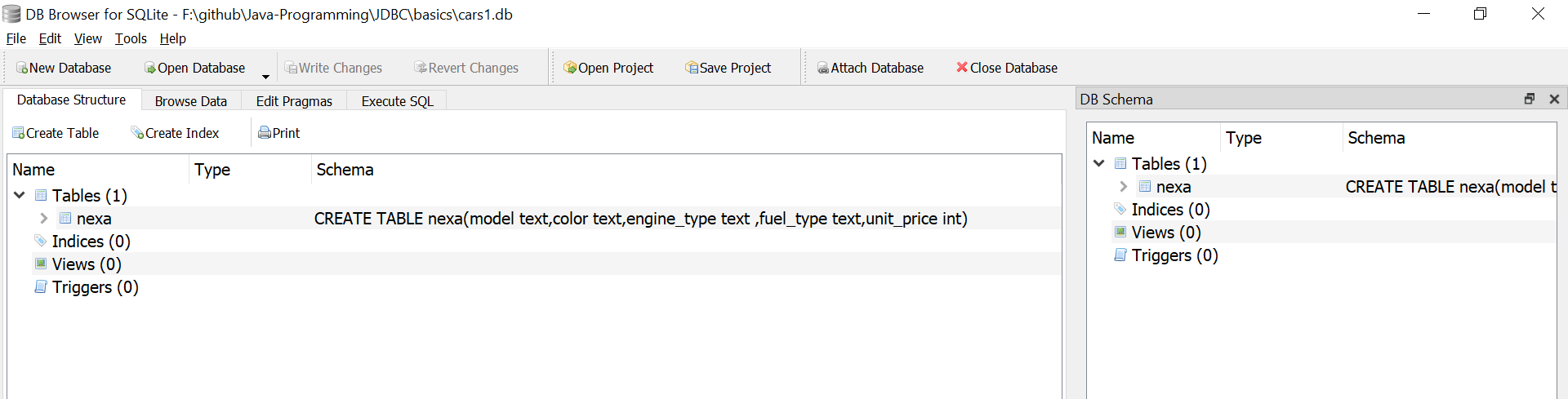
Click Apply and then OK

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   


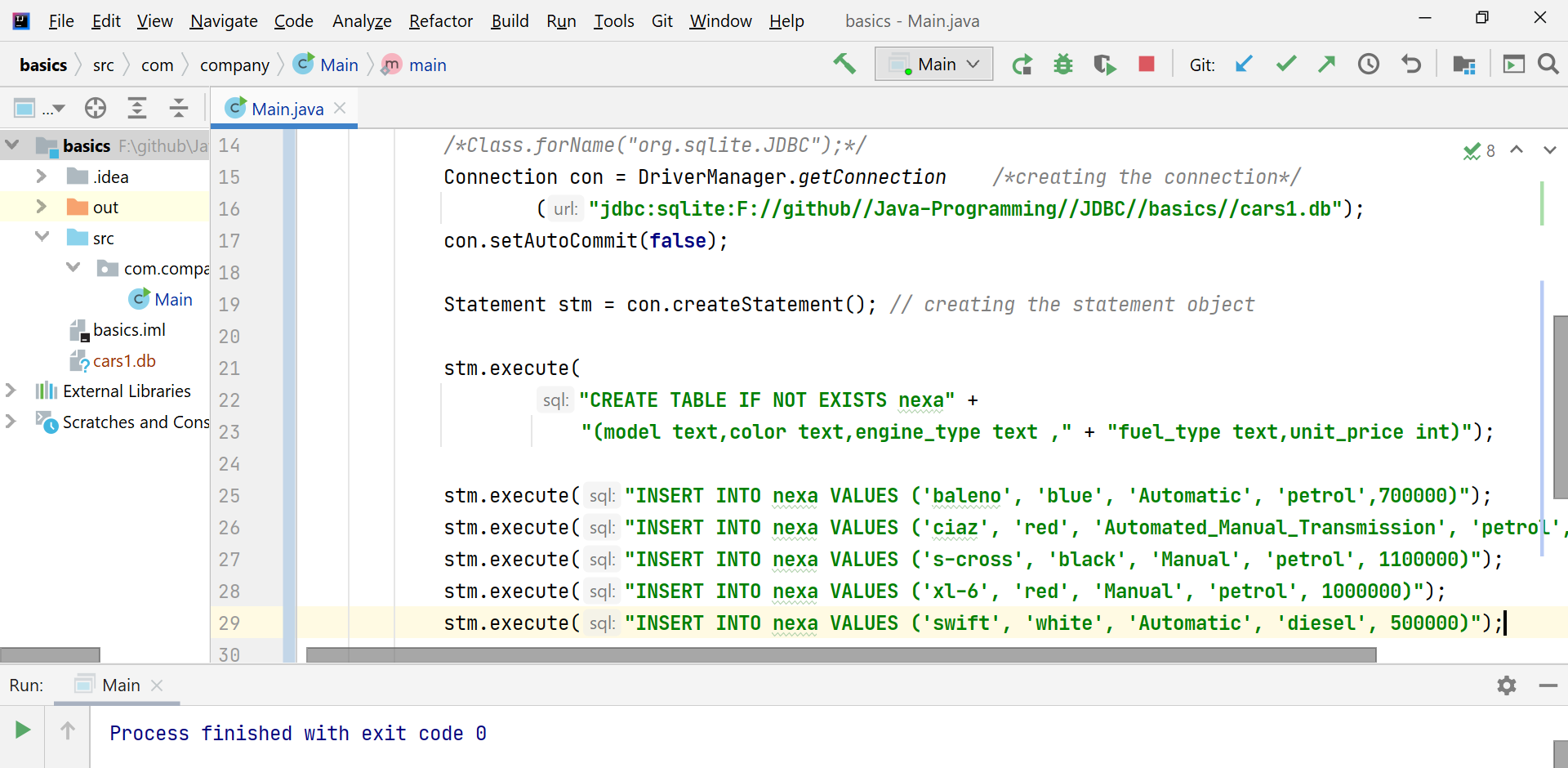
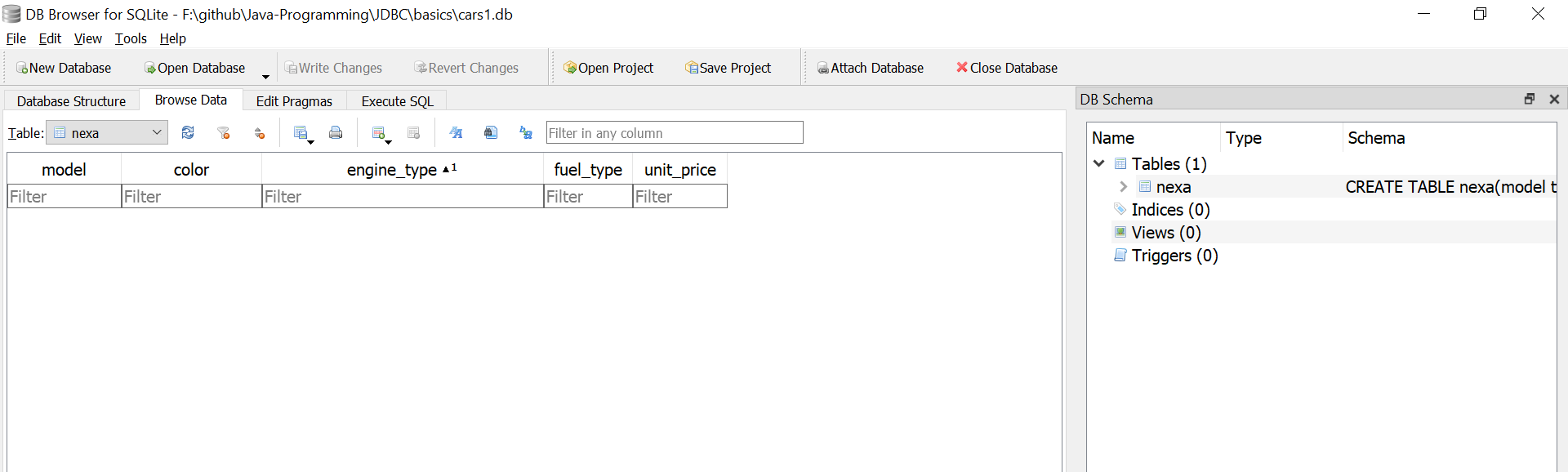
cars1 table is viewed in db browser sqlite.

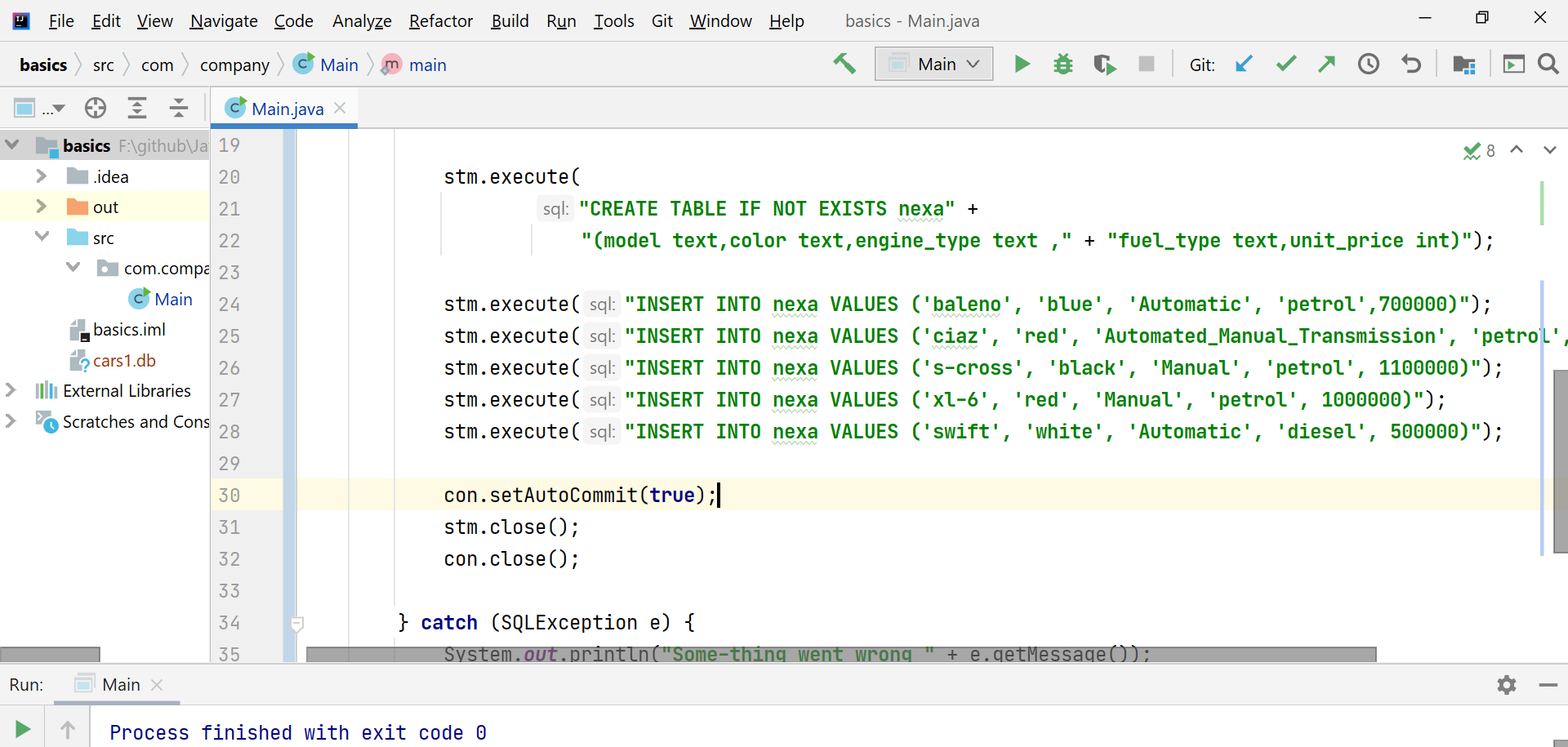
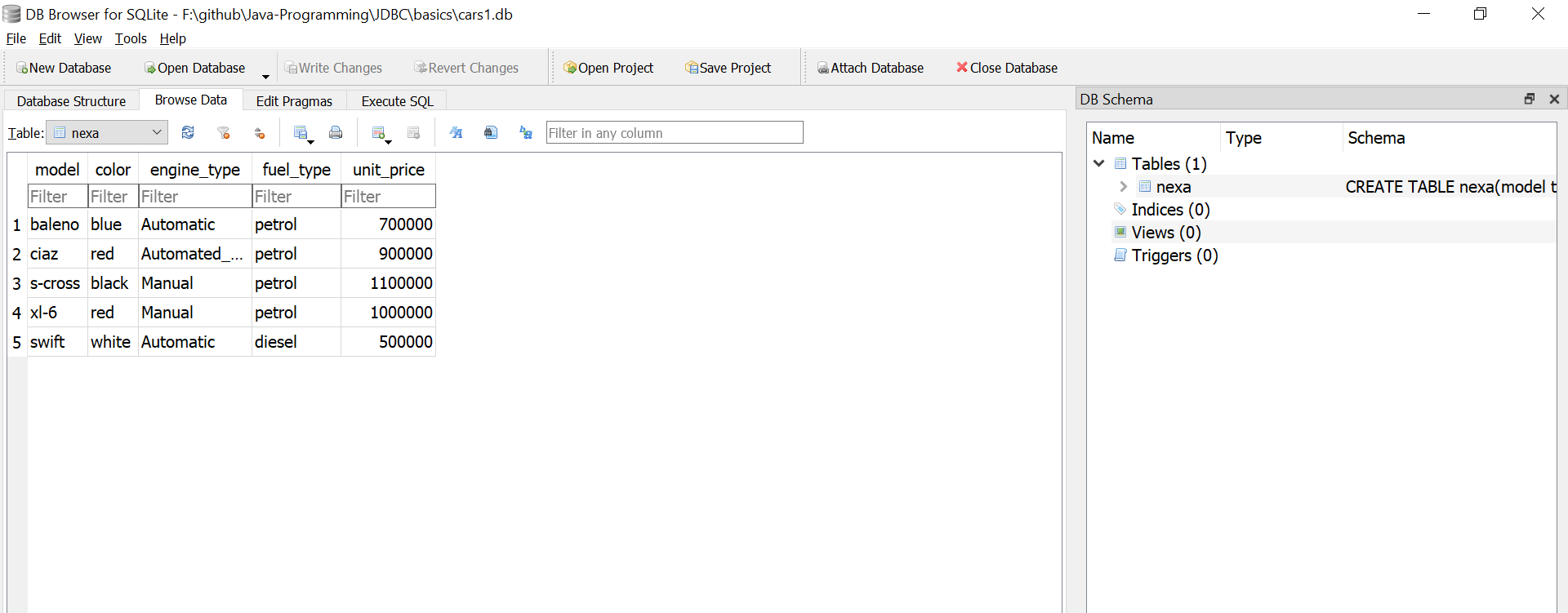
**C** 🡪 Table creation  
  


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”)

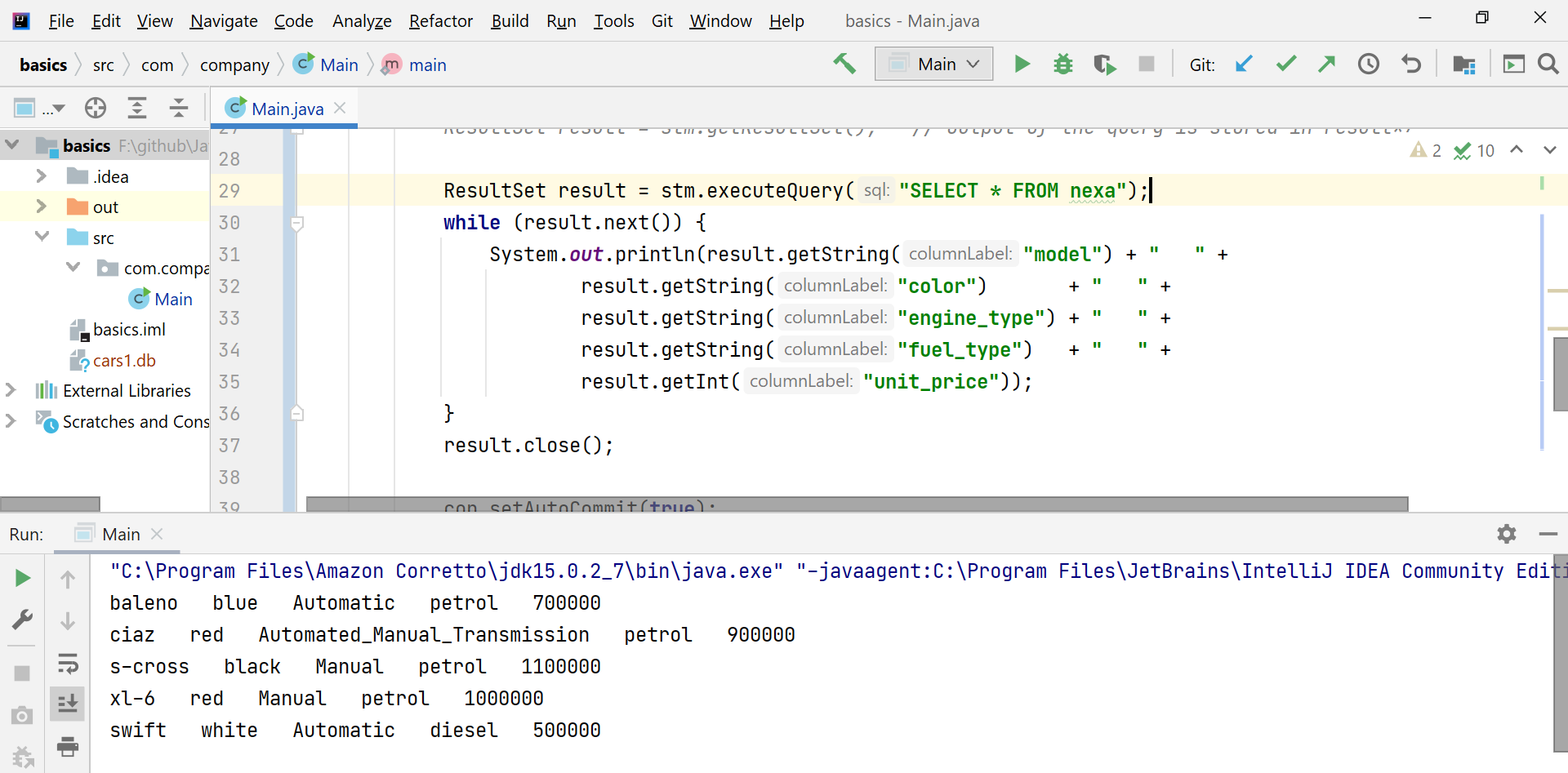
Inserting the values into the data-base by auto-committing(false)  


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


**D** 🡪 Displaying the results



Using “statement.executeQuery()”



# **Data-base creation using derby**

CREATE TABLE "table\_1" (

"player\_ID" INTEGER not null primary key,

"player\_name" VARCHAR(30) ,

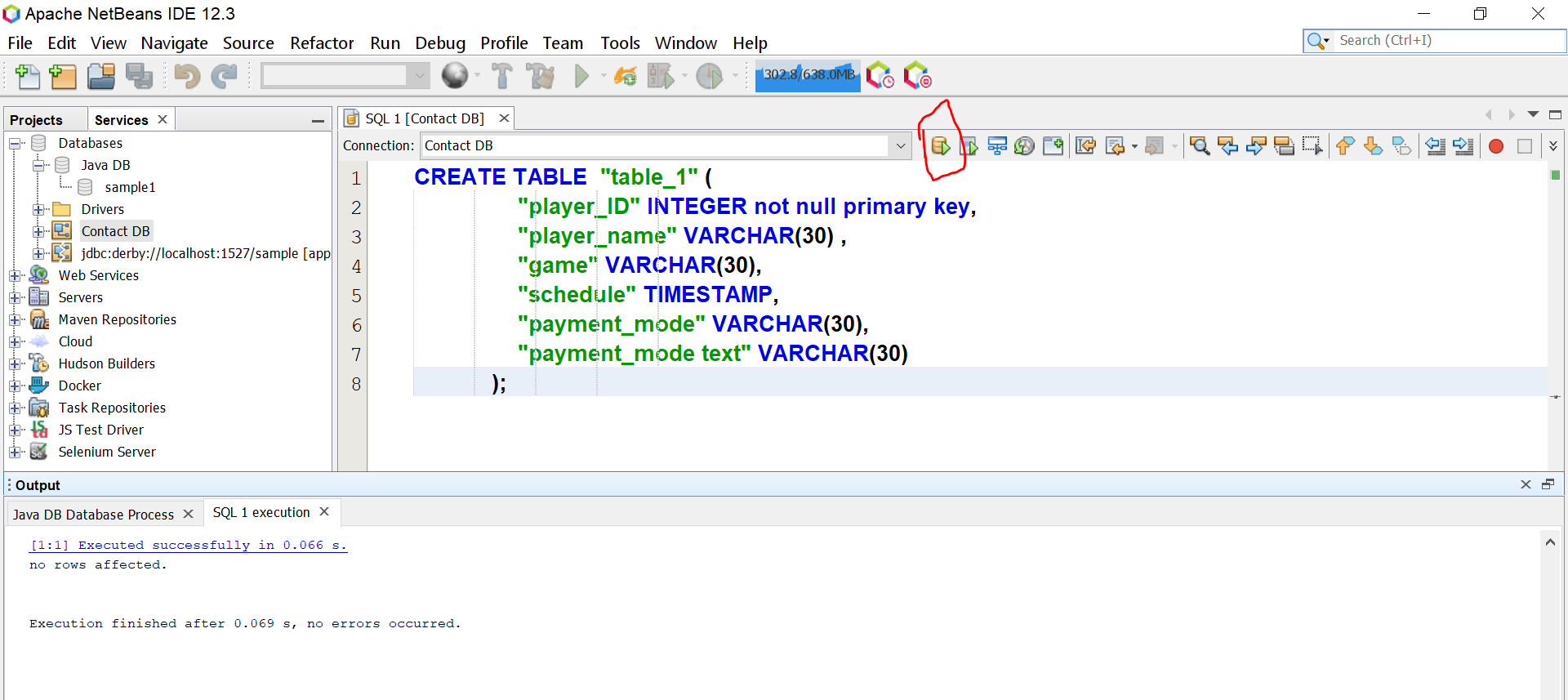
"game" VARCHAR(30),

"schedule" TIMESTAMP,

"payment\_mode" VARCHAR(30),

"payment\_mode text" VARCHAR(30)

);



Lets check whether the table is created/not

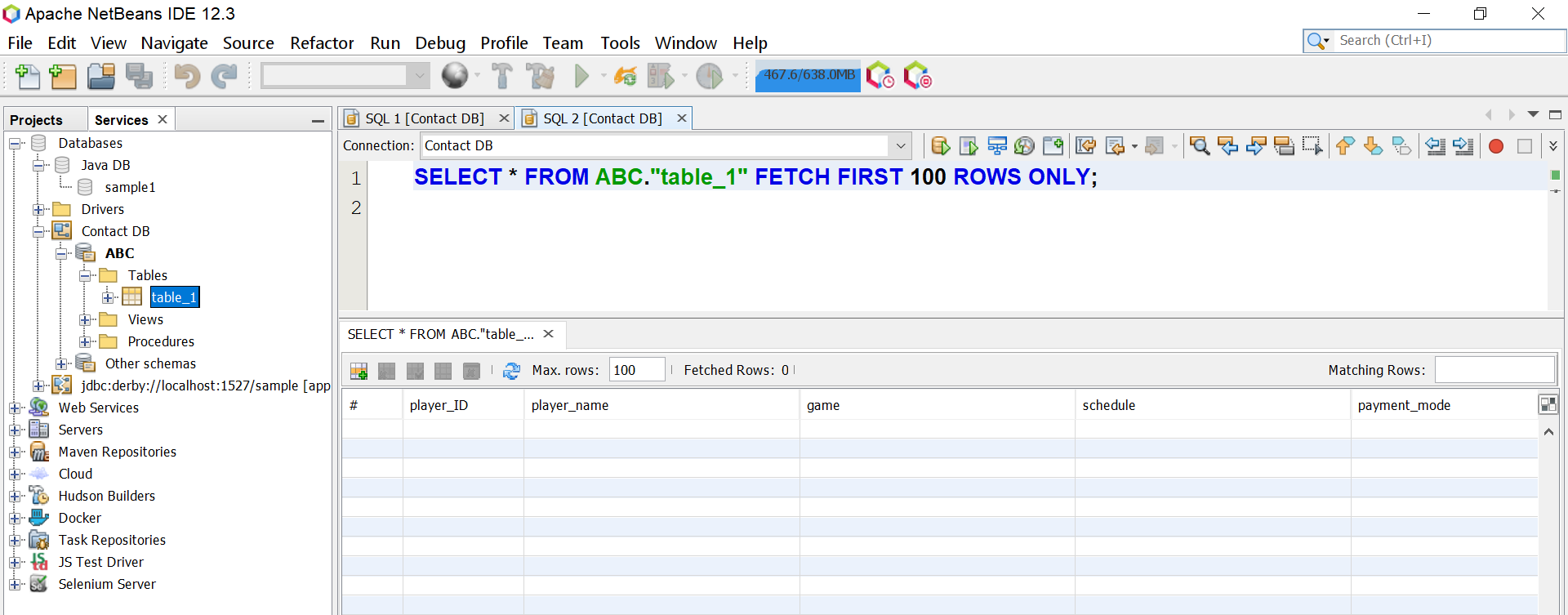


Table creation is successful.

Enter the records into the table

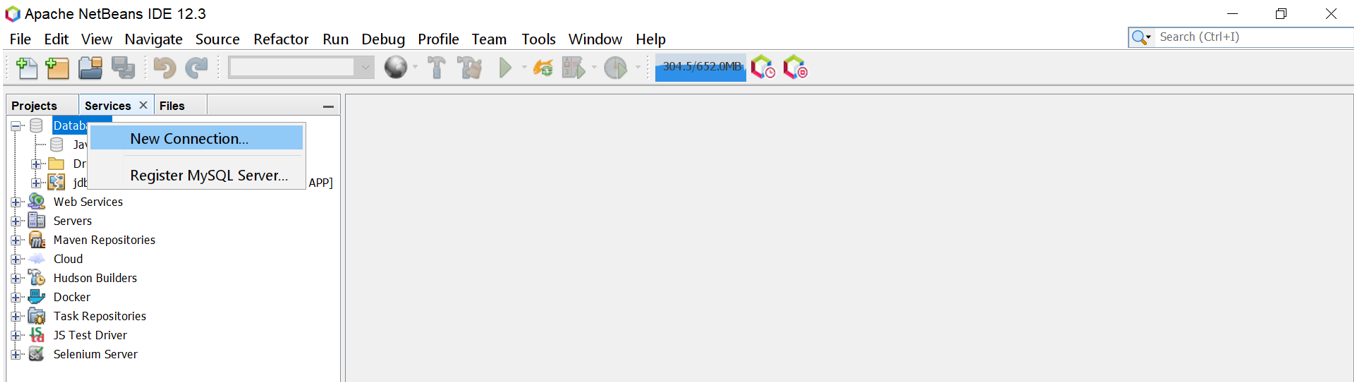
INSERT INTO APPtable\_1 VALUES(1,'Prashanth' , 'cricket' , '2020-10-25 9:10:44' , 'card'),

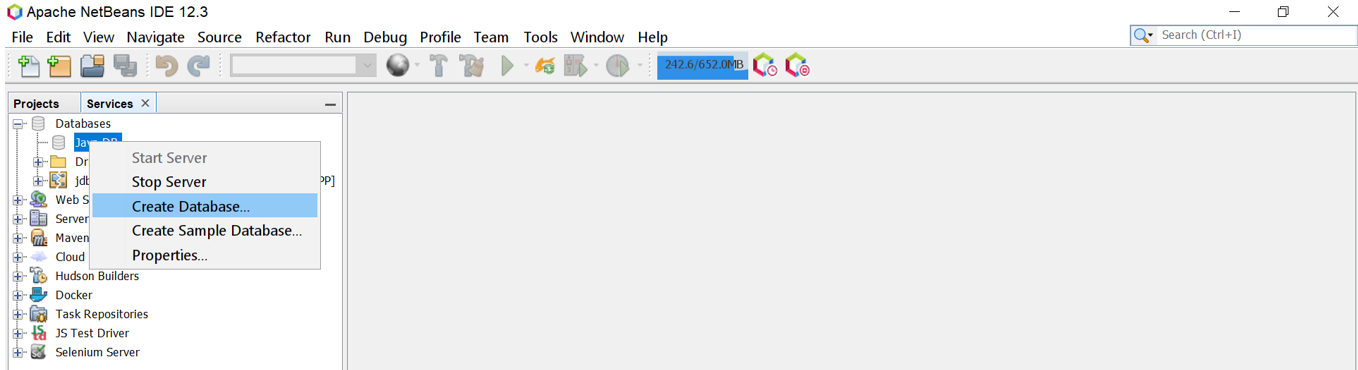
INSERT INTO APP.table\_1 VALUES(2,'Saravanan' , 'base-ball' , '2020-10-25 10:10:44' , 'cash'),

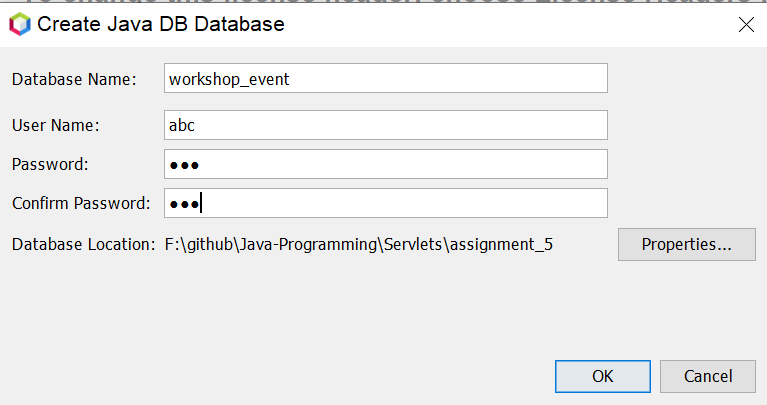
INSERT INTO APP.table\_1 VALUES(3,'Kishore' , 'cricket' , '2020-10-25 11:10:44' , 'demand draft'),

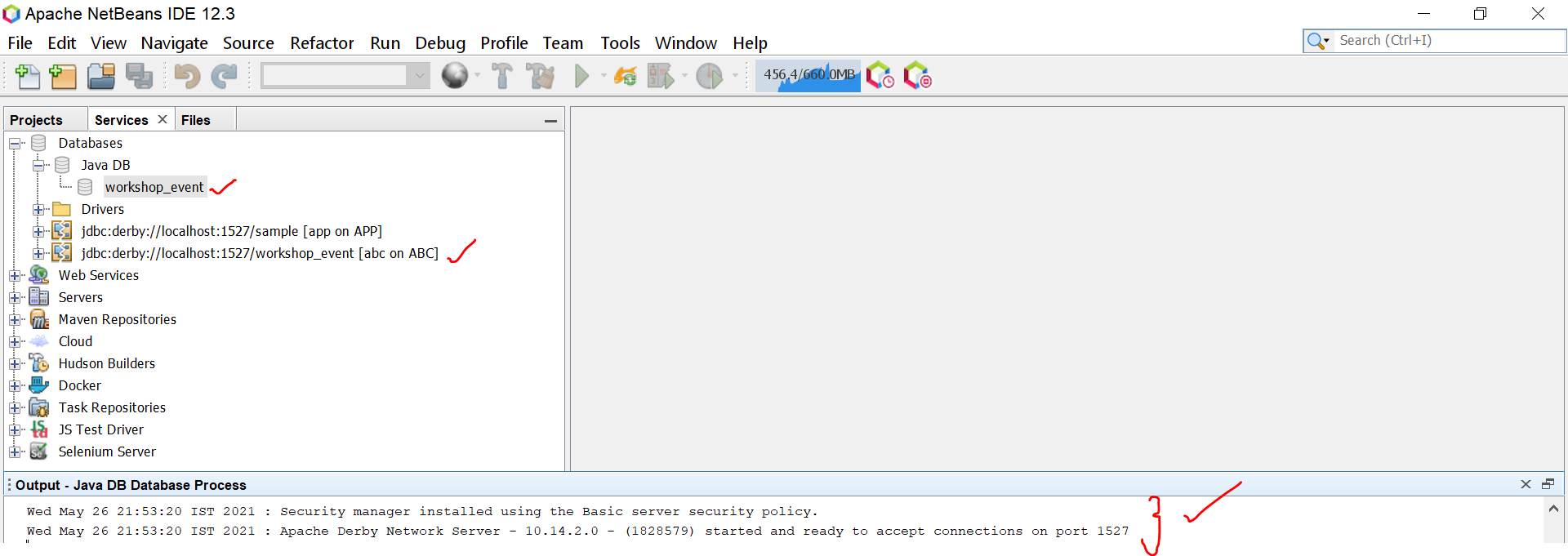
INSERT INTO APP.table\_1 VALUES(4,'Mothish' , 'athletics' , '2020-10-25 14:10:44' , 'cash'),

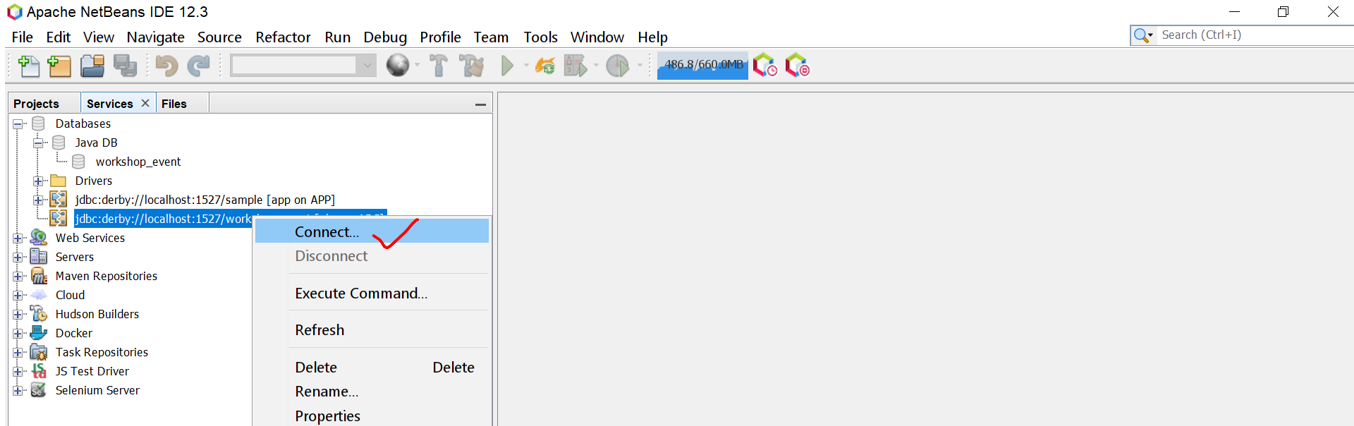
INSERT INTO APP.table\_1 VALUES(5,'Sabari' , 'foot-ball' , '2020-10-25 15:10:44' , 'cheque')

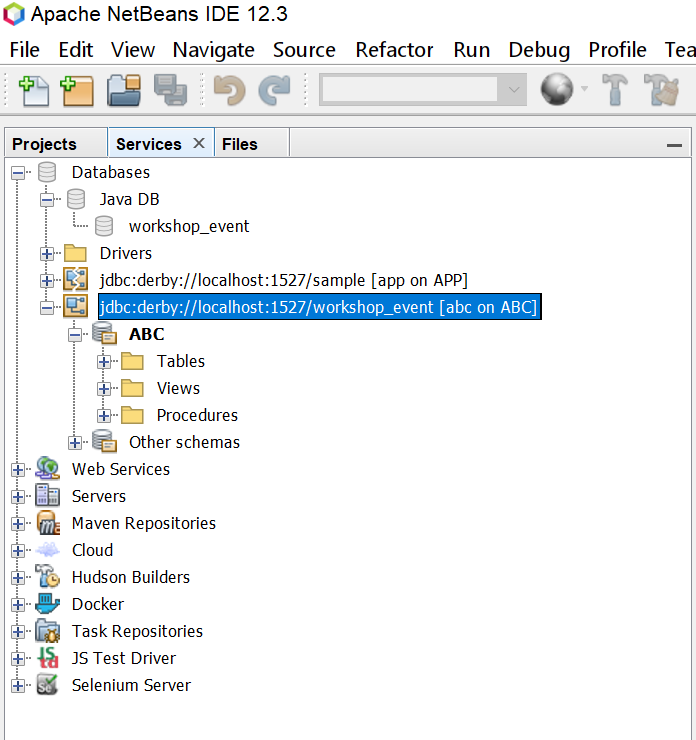
You can configure a new data-base also (like mysql,…..) by creating a new connection.

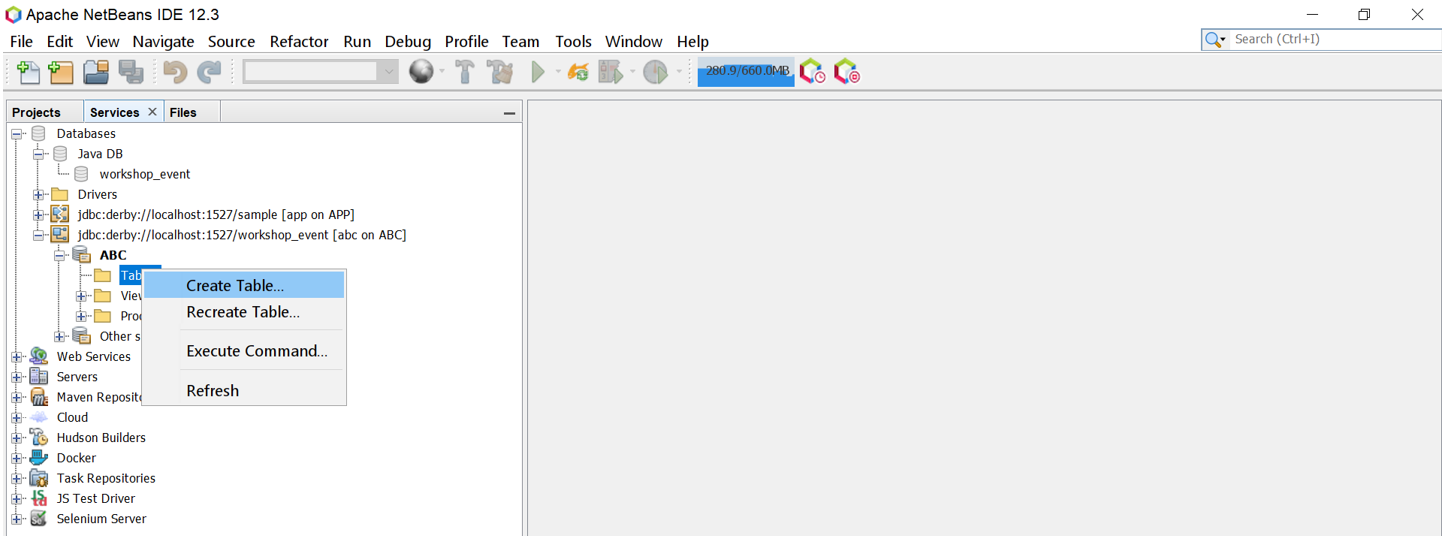
Here in glass-fish server , JavaDerby data-base is there by-default. So we can create the data-base.

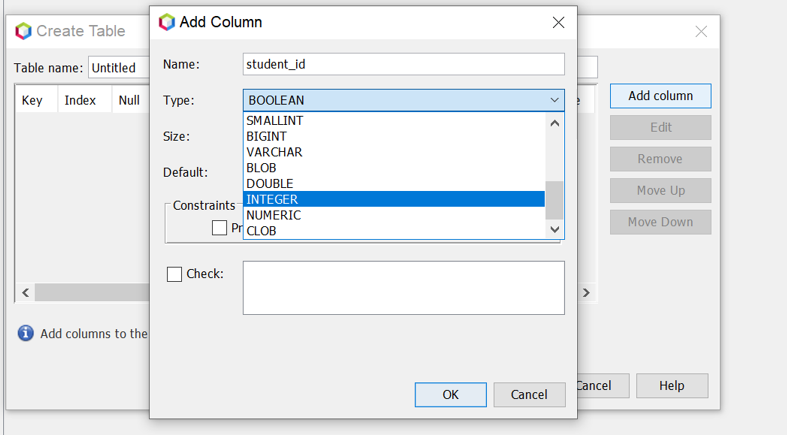
Creating the data-base with user name and password  


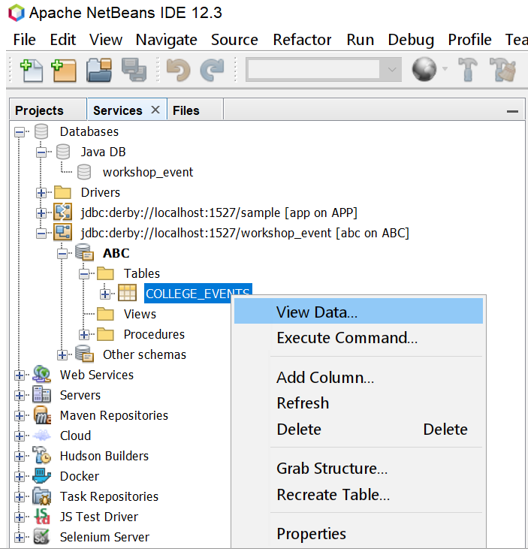
Data-base have created successfully and started to run  


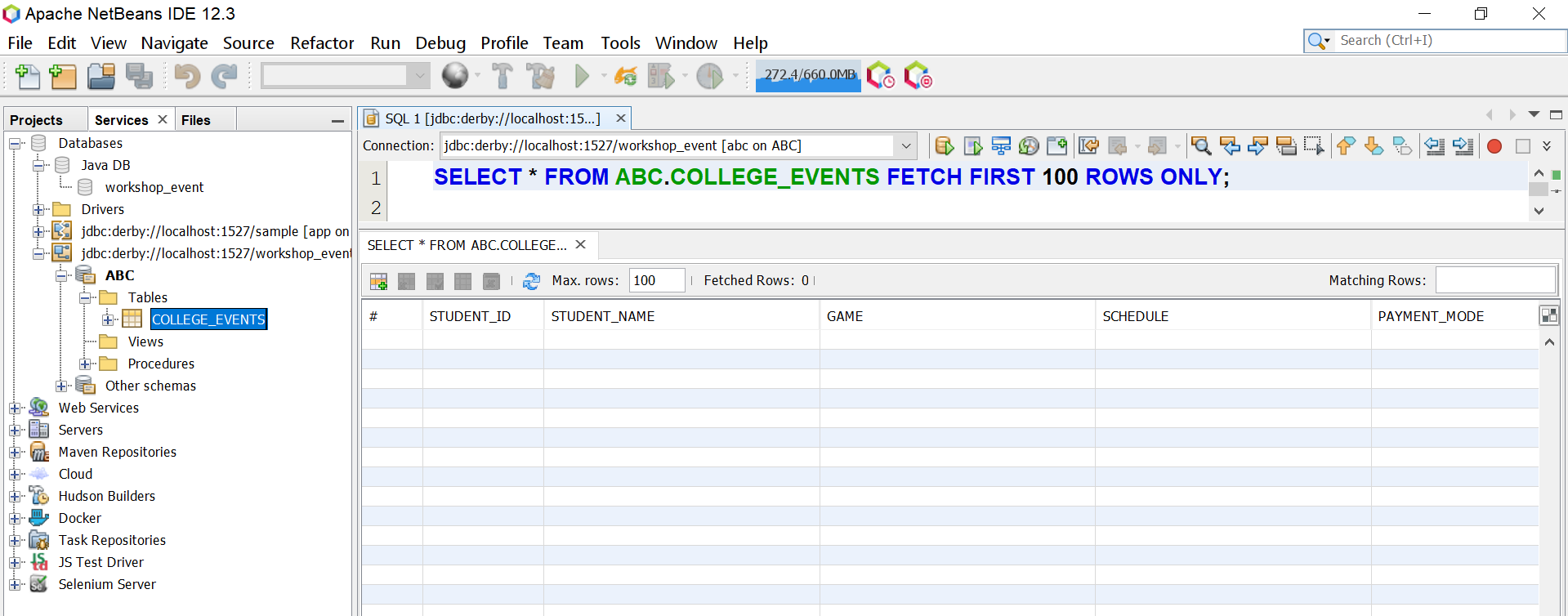
Click “connect” to connect the data-base.

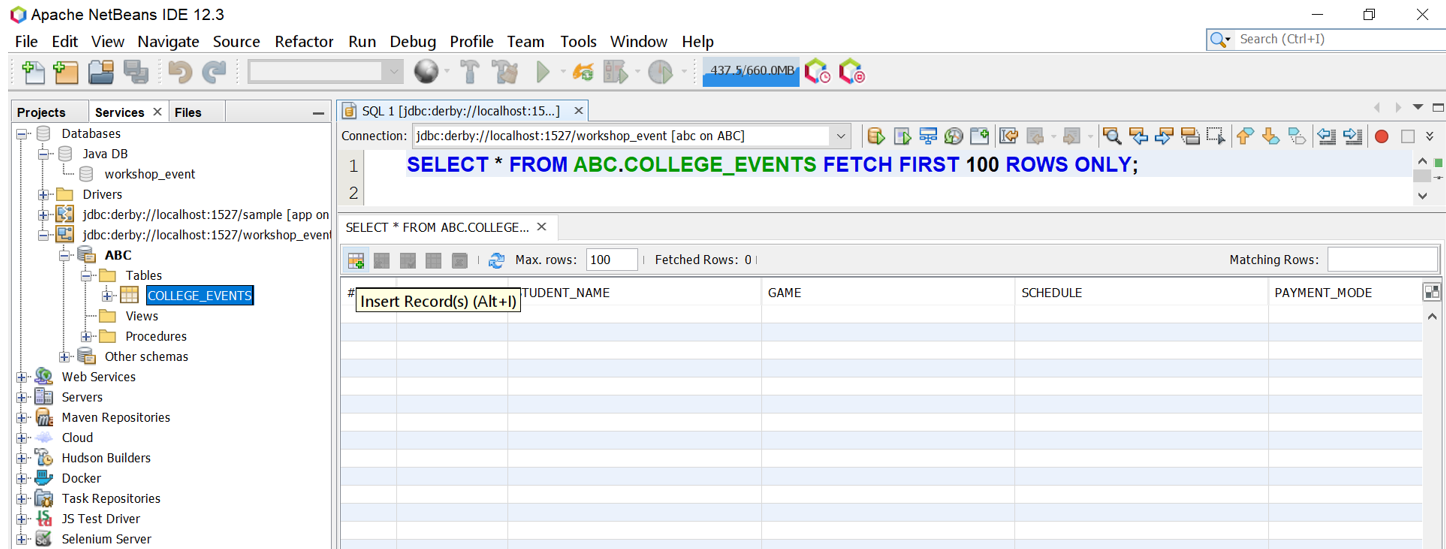
After the data-base connection is made, click the + to see the tables inside the data-base.

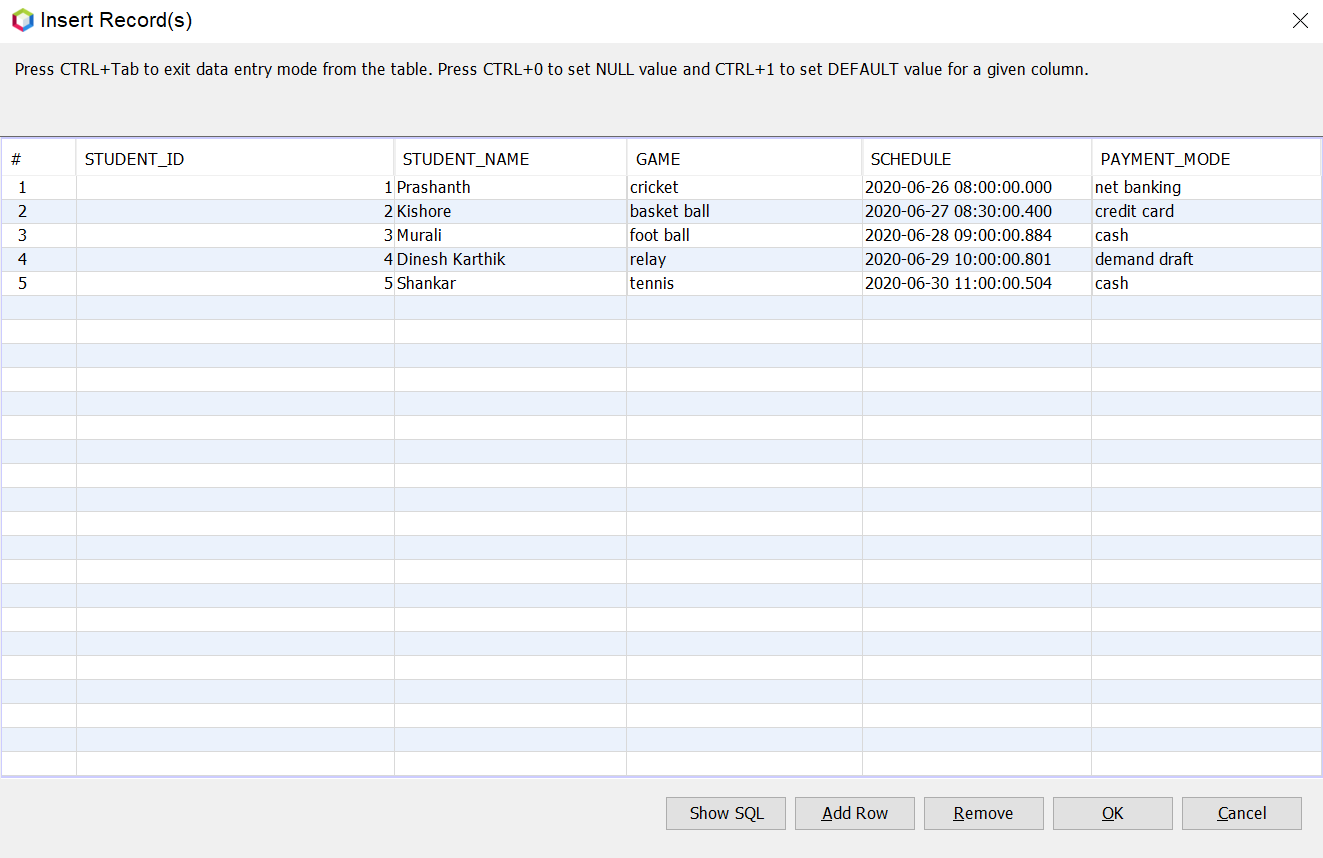
Create the table  


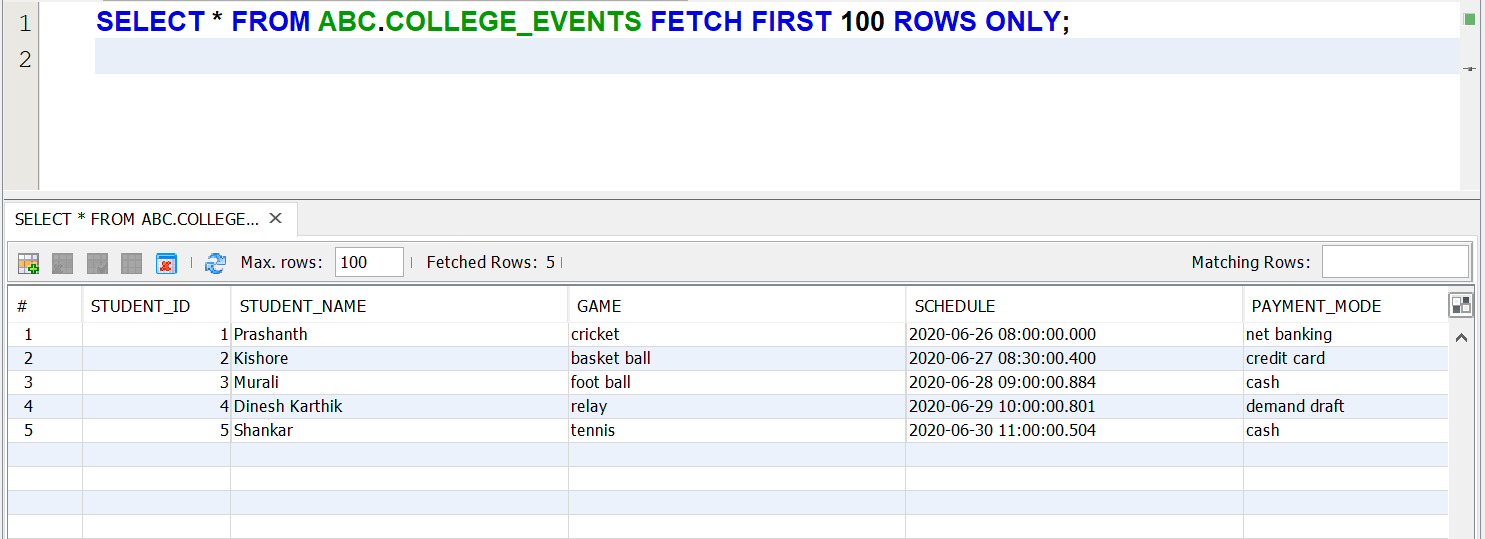


Click to view the table structure.  




Inserting the records into the table

  
click ok.

  
view data.