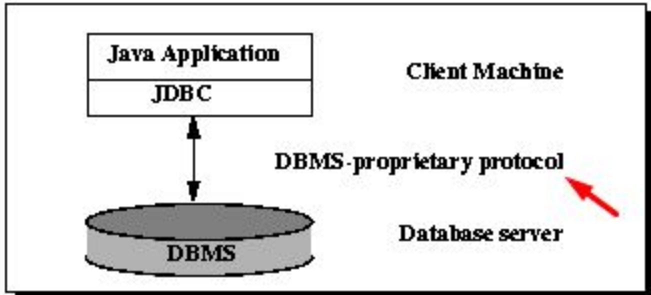
AfroDev

Java Básico - Aula 11 - Parte 1

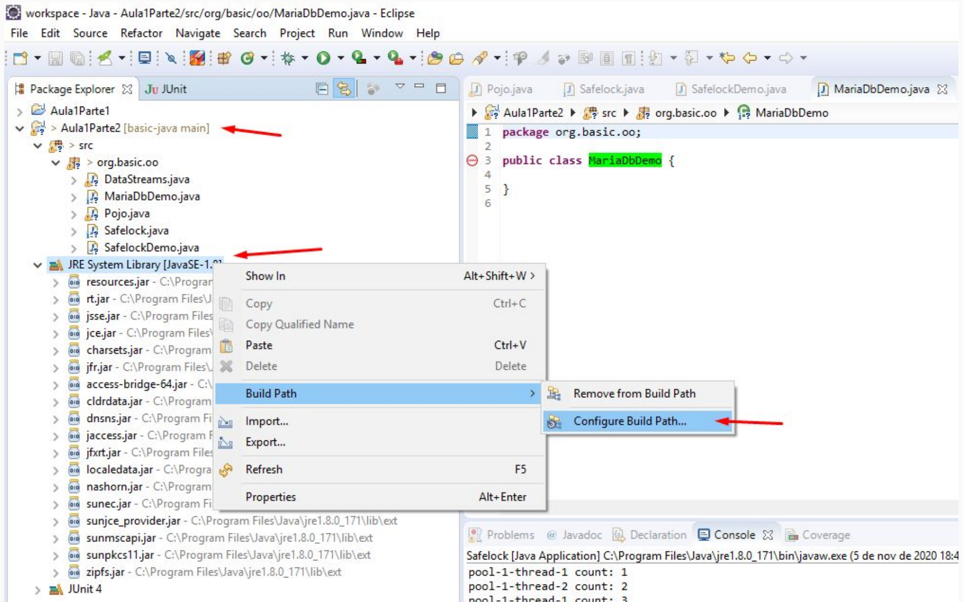
MariaDB - Connector/J

- Vamos entender o conector: https://mariadb.com/kb/en/about-mariadb-connector-j/
- Maneiras de instalar: https://mariadb.com/kb/en/installing-mariadb-connectorj/
- Download (todas as versões): https://downloads.mariadb.org/connector-java/
- Download da 2.6: https://downloads.mariadb.org/connector-java/2.6.2/
- Link direto: https://downloads.mariadb.com/Connectors/java/connector-java-2.6.2/mariadb-java-client-2.6.2.jar



MariaDB - Connector/J

- Agora precisamos importar o JAR do conector no nosso projeto JAVA na IDE

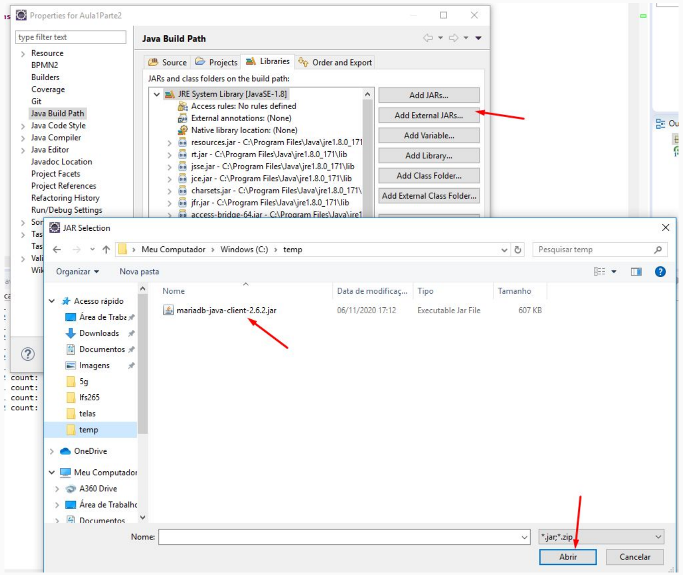

workspace - Java - Aula1Parte2/src/org/basic/oo/MariaDbDemo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Aula1Parte1
> Aula1Parte2 [basic-java main]
> src
> org.basic.oo
DataStreams.java
MariaDbDemo.java
Pojo.java
Safelock.java
SafelockDemo.java
JRE System Library [JavaSE-1.8]
resources.jar - C:\Progran
rt.jar - C:\Program Files\J
jsse.jar - C:\Program Files
jce.jar - C:\Program Files\
charsets.jar - C:\Program
jfr.jar - C:\Program Files\
access-bridge-64.jar - C:\
cldrdata.jar - C:\Program
dnsns.jar - C:\Program Fi
jaccess.jar - C:\Program F
jfxrt.jar - C:\Program Files
localedata.jar - C:\Progra
nashorn.jar - C:\Program
sunec.jar - C:\Program Fi
sunjce_provider.jar - C:\Program Files\Java\jre1.8.0_171\lib\ext
sunmscapi.jar - C:\Program Files\Java\jre1.8.0_171\lib\ext
sunpkcs11.jar - C:\Program Files\Java\jre1.8.0_171\lib\ext
zipfs.jar - C:\Program Files\Java\jre1.8.0_171\lib\ext
JUnit 4
Show In Alt+Shift+W >
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path >
Import...
Export...
Refresh F5
Properties Alt+Enter
Remove from Build Path
Configure Build Path...
Pojo.java
Safelock.java
SafelockDemo.java
MariaDbDemo.java
Aula1Parte2 > src > org.basic.oo > MariaDbDemo
Problems
Javadoc
Declaration
Console
Coverage

```
package org.basic.oo;

public class MariaDbDemo {

}
```

```
Safelock [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (5 de nov de 2020 18:4
pool-1-thread-1 count: 1
pool-1-thread-2 count: 2
pool-1-thread-1 count: 3
```

MariaDB - Connector/J

- Agora precisamos importar o JAR do conector no nosso projeto JAVA na IDE



JARs and class folders on the build path:

- JRE System Library [JavaSE-1.8]
  - Access rules: No rules defined
  - External annotations: (None)
  - Native library location: (None)
  - resources.jar - C:\Program Files\Java\jre1.8.0_171
  - rt.jar - C:\Program Files\Java\jre1.8.0_171\lib
  - jsse.jar - C:\Program Files\Java\jre1.8.0_171\lib
  - jce.jar - C:\Program Files\Java\jre1.8.0_171\lib
  - charsets.jar - C:\Program Files\Java\jre1.8.0_171\
  - jfr.jar - C:\Program Files\Java\jre1.8.0_171\lib
  - access-bridge-64.jar - C:\Program Files\Java\jre1

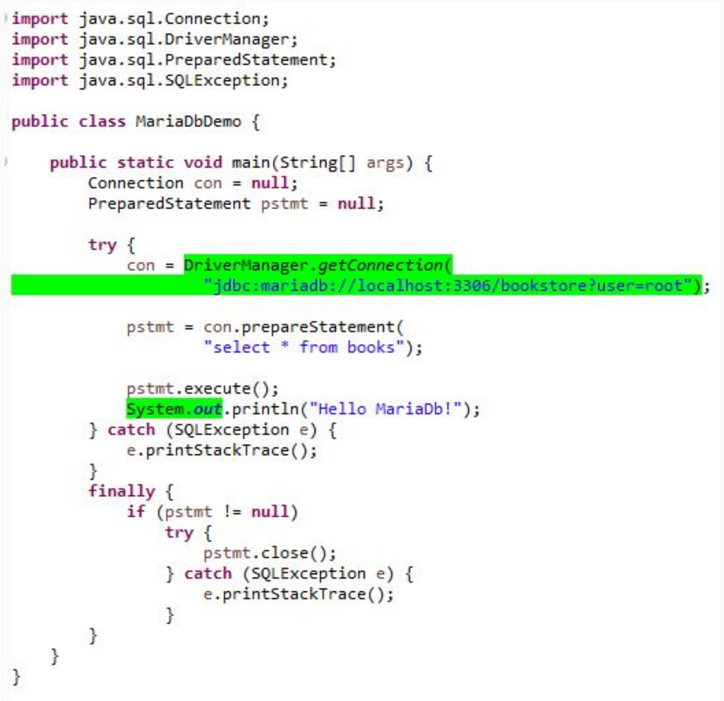
Meu Computador > Windows (C:) > temp

Nome	Data de modificaç...	Tipo	Tamanho
mariadb-java-client-2.6.2.jar	06/11/2020 17:12	Executable Jar File	607 KB

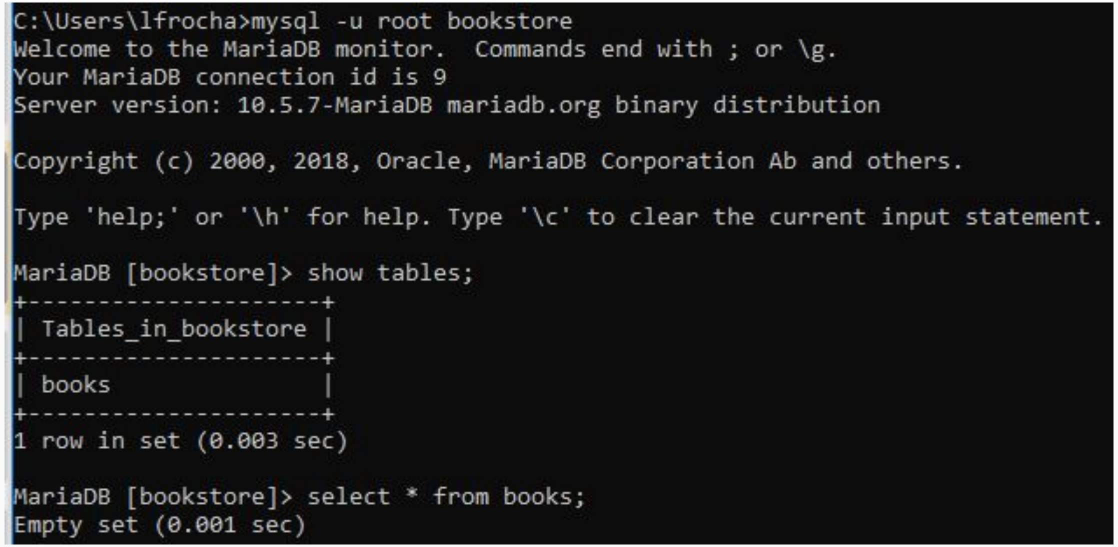
MariaDB - Connector/J - Teste

- Vamos fazer um teste de conectividade do nosso programa JAVA com o banco MariaDB via JDBC
- Guia JDBC: https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html
- Guia MariaDB: https://mariadb.com/kb/en/about-mariadb-connector-j/

MariaDB - Connector/J - Teste OK



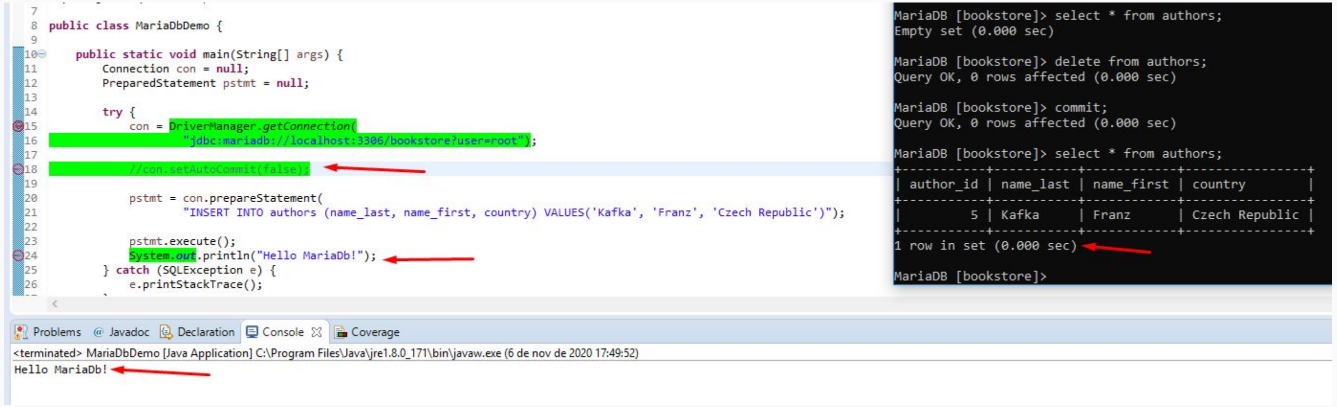
```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class MariaDbDemo {

    public static void main(String[] args) {
        Connection con = null;
        PreparedStatement pstmt = null;

        try {
            con = DriverManager.getConnection(
                "jdbc:mariadb://localhost:3306/bookstore?user=root");

            pstmt = con.prepareStatement(
                "select * from books");

            pstmt.execute();
            System.out.println("Hello MariaDb!");
        } catch (SQLException e) {
            e.printStackTrace();
        }
        finally {
            if (pstmt != null)
                try {
                    pstmt.close();
                } catch (SQLException e) {
                    e.printStackTrace();
                }
        }
    }
}
```



```
C:\Users\lfrocha>mysql -u root bookstore
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 10.5.7-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [bookstore]> show tables;
+---------------------+
| Tables_in_bookstore |
+---------------------+
| books               |
+---------------------+
1 row in set (0.003 sec)

MariaDB [bookstore]> select * from books;
Empty set (0.001 sec)
```

MariaDB - Transação e AutoCommit

- A conexão mantém a propriedade autoCommit
- O MariaDB, por default, é true
- O que isso significa?



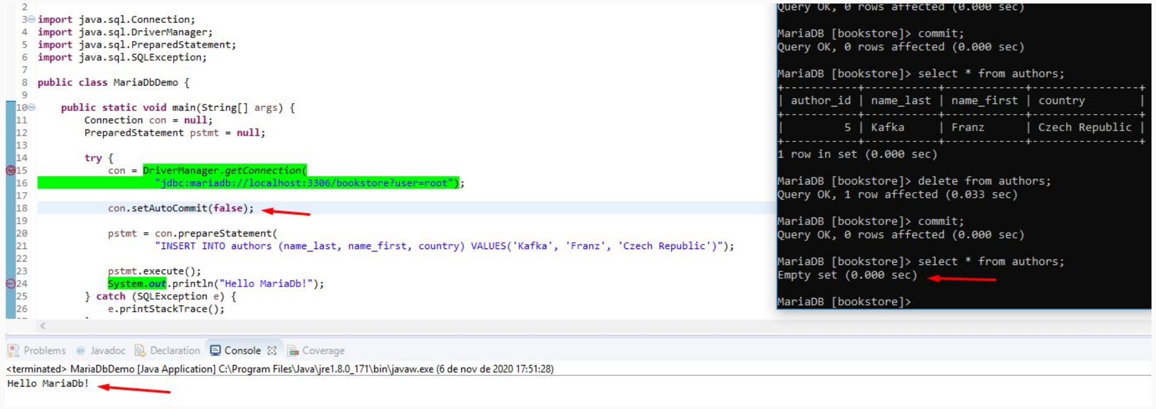
```
public class MariaDbDemo {

    public static void main(String[] args) {
        Connection con = null;
        PreparedStatement pstmt = null;

        try {
            con = DriverManager.getConnection(
                "jdbc:mariadb://localhost:3306/bookstore?user=root");

            //con.setAutoCommit(false);

            pstmt = con.prepareStatement(
                "INSERT INTO authors (name_last, name_first, country) VALUES('Kafka', 'Franz', 'Czech Republic')");

            pstmt.execute();
            System.out.println("Hello MariaDb!");
        } catch (SQLException e) {
            e.printStackTrace();
```

```
MariaDB [bookstore]> select * from authors;
Empty set (0.000 sec)

MariaDB [bookstore]> delete from authors;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> commit;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> select * from authors;
+-----------+-----------+------------+----------------+
| author_id | name_last | name_first | country        |
+-----------+-----------+------------+----------------+
|         5 | Kafka     | Franz      | Czech Republic |
+-----------+-----------+------------+----------------+
1 row in set (0.000 sec)

MariaDB [bookstore]>
```

```
<terminated> MariaDbDemo [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (6 de nov de 2020 17:49:52)
Hello MariaDb!
```

MariaDB - Transação e AutoCommit

- Vamos setar para false
- Use SEMPRE assim!!!



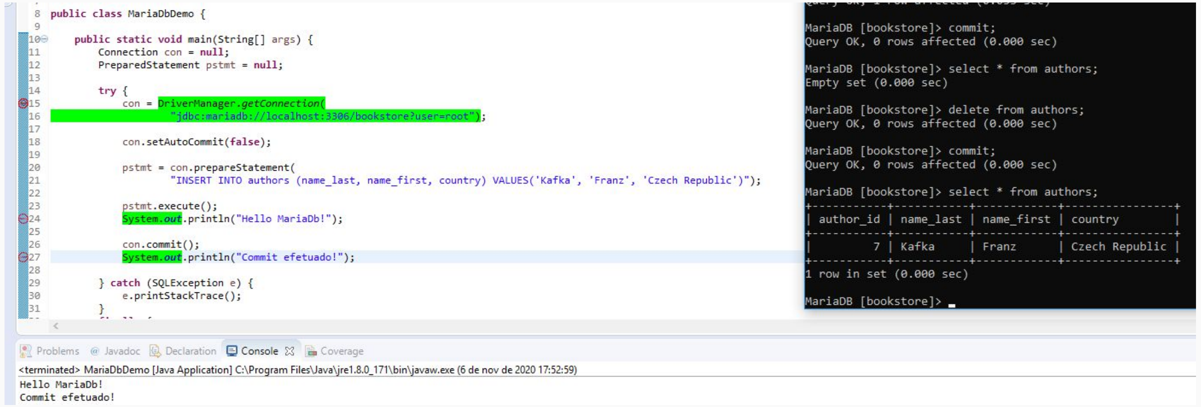
```java

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class MariaDbDemo {

    public static void main(String[] args) {
        Connection con = null;
        PreparedStatement pstmt = null;

        try {
            con = DriverManager.getConnection(
                    "jdbc:mariadb://localhost:3306/bookstore?user=root");

            con.setAutoCommit(false);

            pstmt = con.prepareStatement(
                    "INSERT INTO authors (name_last, name_first, country) VALUES('Kafka', 'Franz', 'Czech Republic')");

            pstmt.execute();
            System.out.println("Hello MariaDb!");
        } catch (SQLException e) {
            e.printStackTrace();
```

```
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> commit;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> select * from authors;
+-----------+-----------+------------+----------------+
| author_id | name_last | name_first | country        |
+-----------+-----------+------------+----------------+
|         5 | Kafka     | Franz      | Czech Republic |
+-----------+-----------+------------+----------------+
1 row in set (0.000 sec)

MariaDB [bookstore]> delete from authors;
Query OK, 1 row affected (0.033 sec)

MariaDB [bookstore]> commit;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> select * from authors;
Empty set (0.000 sec)

MariaDB [bookstore]>
```

```
<terminated> MariaDbDemo [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (6 de nov de 2020 17:51:28)
Hello MariaDb!
```

MariaDB - Transação e AutoCommit

- Agora vamos de fato efetivar (comitar) a transação



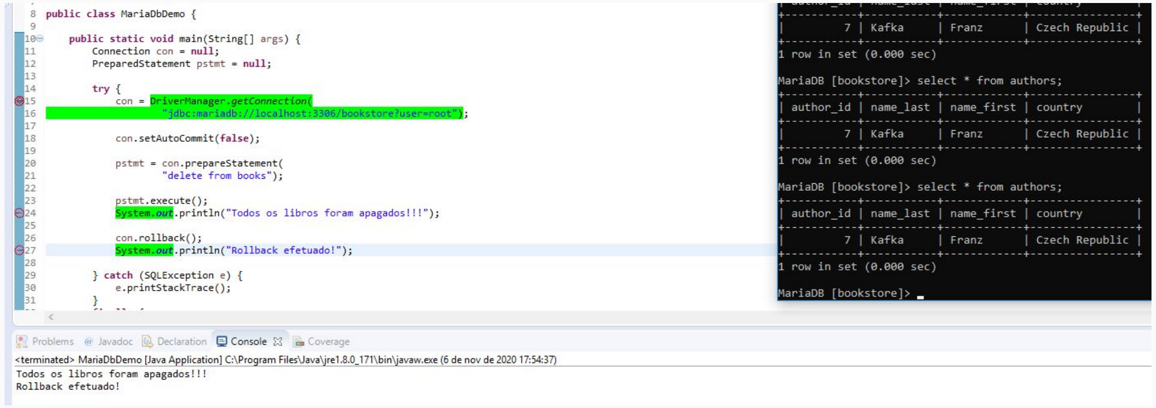
```java
public class MariaDbDemo {

    public static void main(String[] args) {
        Connection con = null;
        PreparedStatement pstmt = null;

        try {
            con = DriverManager.getConnection(
                    "jdbc:mariadb://localhost:3306/bookstore?user=root");

            con.setAutoCommit(false);

            pstmt = con.prepareStatement(
                    "INSERT INTO authors (name_last, name_first, country) VALUES('Kafka', 'Franz', 'Czech Republic')");

            pstmt.execute();
            System.out.println("Hello MariaDb!");

            con.commit();
            System.out.println("Commit efetuado!");

        } catch (SQLException e) {
            e.printStackTrace();
        }
```

```
MariaDB [bookstore]> commit;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> select * from authors;
Empty set (0.000 sec)

MariaDB [bookstore]> delete from authors;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> commit;
Query OK, 0 rows affected (0.000 sec)

MariaDB [bookstore]> select * from authors;
+-----------+-----------+------------+----------------+
| author_id | name_last | name_first | country        |
+-----------+-----------+------------+----------------+
|         7 | Kafka     | Franz      | Czech Republic |
+-----------+-----------+------------+----------------+
1 row in set (0.000 sec)

MariaDB [bookstore]> _
```

```
<terminated> MariaDbDemo [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (6 de nov de 2020 17:52:59)
Hello MariaDb!
Commit efetuado!
```

MariaDB - Transação e AutoCommit

- Agora vamos desfazer (rollback) a transação



```java
public class MariaDbDemo {

    public static void main(String[] args) {
        Connection con = null;
        PreparedStatement pstmt = null;

        try {
            con = DriverManager.getConnection(
                    "jdbc:mariadb://localhost:3306/bookstore?user=root");

            con.setAutoCommit(false);

            pstmt = con.prepareStatement(
                    "delete from books");

            pstmt.execute();
            System.out.println("Todos os libros foram apagados!!!");

            con.rollback();
            System.out.println("Rollback efetuado!");

        } catch (SQLException e) {
            e.printStackTrace();
        }
```

```
+-----------+-----------+------------+----------------+
|         7 | Kafka     | Franz      | Czech Republic |
+-----------+-----------+------------+----------------+
1 row in set (0.000 sec)

MariaDB [bookstore]> select * from authors;
+-----------+-----------+------------+----------------+
| author_id | name_last | name_first | country        |
+-----------+-----------+------------+----------------+
|         7 | Kafka     | Franz      | Czech Republic |
+-----------+-----------+------------+----------------+
1 row in set (0.000 sec)

MariaDB [bookstore]> select * from authors;
+-----------+-----------+------------+----------------+
| author_id | name_last | name_first | country        |
+-----------+-----------+------------+----------------+
|         7 | Kafka     | Franz      | Czech Republic |
+-----------+-----------+------------+----------------+
1 row in set (0.000 sec)

MariaDB [bookstore]> _
```

```
<terminated> MariaDbDemo [Java Application] C:\Program Files\Java\jre1.8.0_171\bin\javaw.exe (6 de nov de 2020 17:54:37)
Todos os libros foram apagados!!!
Rollback efetuado!
```

MariaDB - CRUD

- Para escrita (create, update e delete): podemos usar **Statement** ou **PreparedStatement**. De qualquer forma, vamos sempre invocar o método **executeUpdate**
- Para leitura (read): de novo vamos usar **Statement** ou **PreparedStatement**, mas invocando o método **getResultSet**
    - Um resultSet é um cursor do resultado obtido pelo banco. Assim conseguimos transportar os dados da base e criarmos nossos objetos Java com eles.

MariaDB - CRUD - Exemplo

- INSERT: https://docs.oracle.com/javase/tutorial/jdbc/basics/tables.html#populate
- UPDATE: https://docs.oracle.com/javase/tutorial/jdbc/basics/prepared.html#overview_ps,
- DELETE: vimos no exemplo do rollback. Vamos ver novamente?
- SELECT: https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html

MariaDB - Exercício

- Utilizando JDBC, fazer CRUD para a tabela/entidade “authors” do tutorial do MariaDB
- Utilizando JDBC, fazer CRUD para a tabela/entidade “books” do tutorial do MariaDB

Revisão

- Vimos que com JDBC conseguimos fazer a mesma coisa que fazemos com o *client* “mysql”
- Isso faz com que o nosso programa JAVA faça qualquer CRUD utilizando JDBC
- O resultado desse CRUD pode virar uma aplicação que gere valor para o cliente final
- As camadas ficam desacopladas e os dados protegidos