Short Tutorial about

Java Database Connectivity (JDBC)

from

mraab

Content

List of figures

List of tables

List of abbreviations

Begriff	Erklärung
MySQL	My Structered Query Language
JDBC	Java Database Connectivity

table 1: list of abbreviations

1 Java Database Connectivity (JDBC)

JDBC is a database interface and is implemented for Java-platforms. The main task is the connection between the database and the IDE Java Eclipse. JDBC executes database models from different manufacturer. The figure 1 shows how the implemented program and the database connection works.

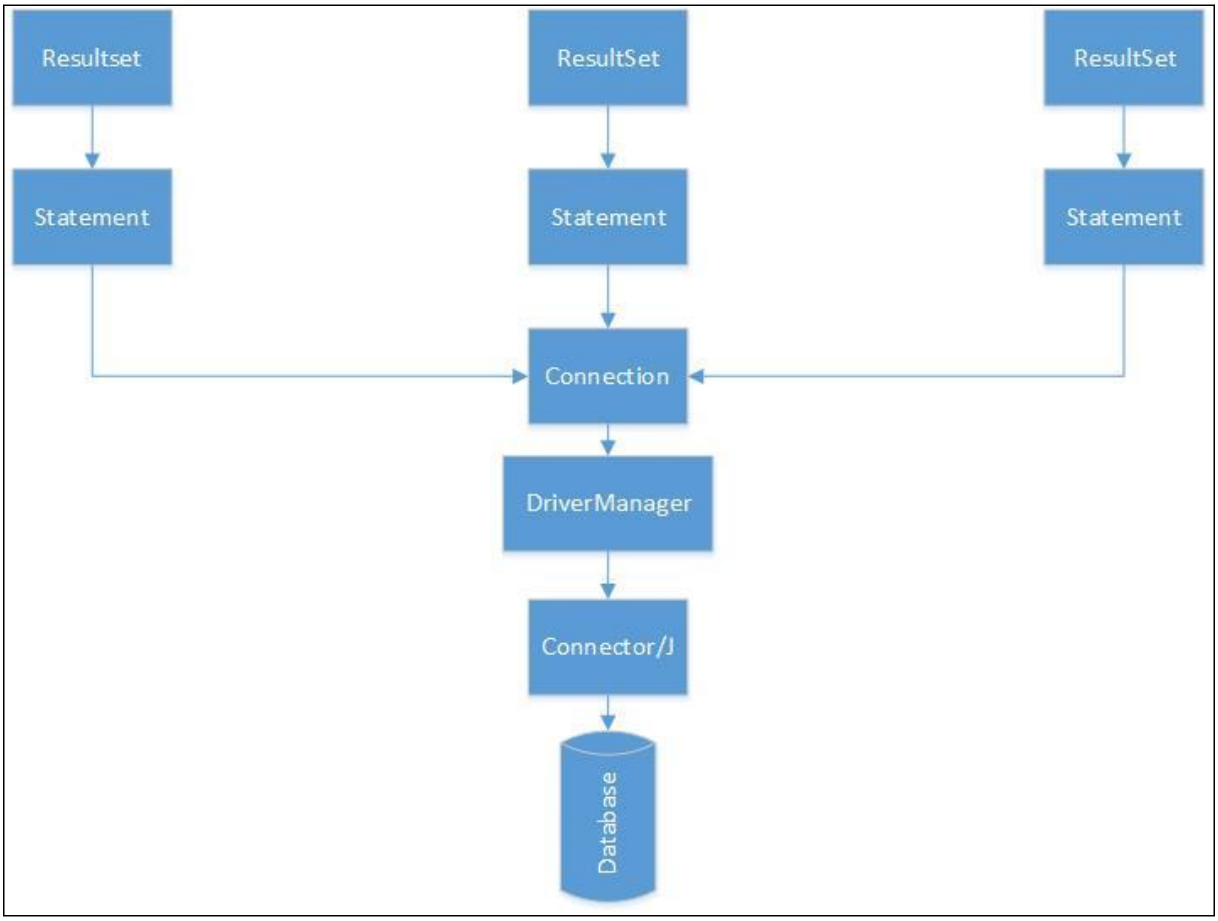


figure 1: overview connection

2 approach

2.1 Download MySQL 5.6.17

Download the zip-archive MySQL 5.6.17 for Windows (32- or 64-bit-version).

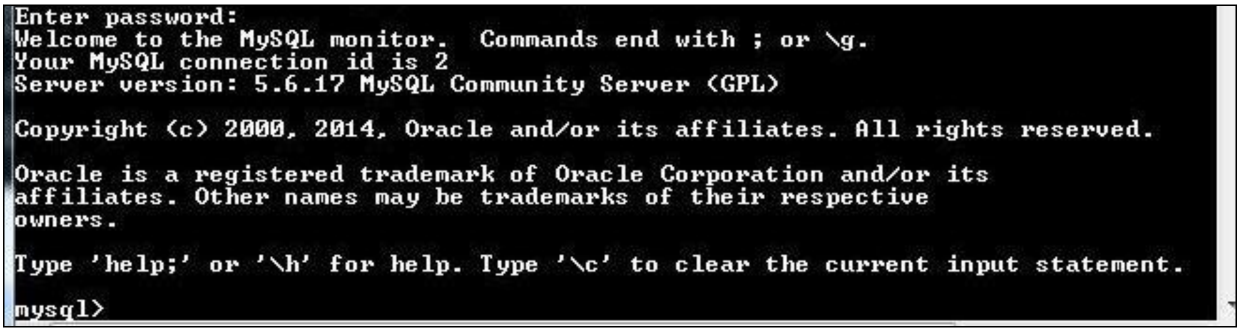
Download-Link: http://dev.mysql.com/downloads/mysql/ erhältlich.

2.2 Installation MySQL 5.6.17

- Unzip the archive
- Start the mysqld.exe to start the sql-services in the background
- Start the mysql.exe or use the command line interface and type the commands: *„cd <path>/mysql/bin“ and „mysql –install“*

2.3 Create database connection with MySQL

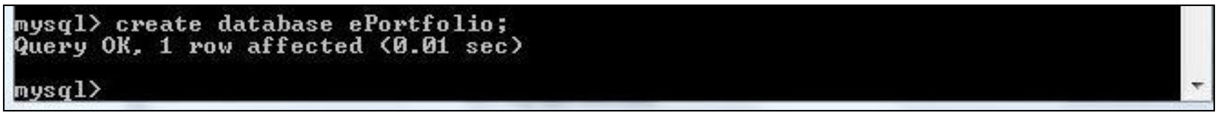
- Navigate with the cd command to the bin folder of mysql 5.6.17
- Login as admin/root to the command line interface *„mysql -u root -p “*
- Enter password with a blank (figure 2)



```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.6.17 MySQL Community Server (GPL)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql>
```

figure 2: Login SQL-Server

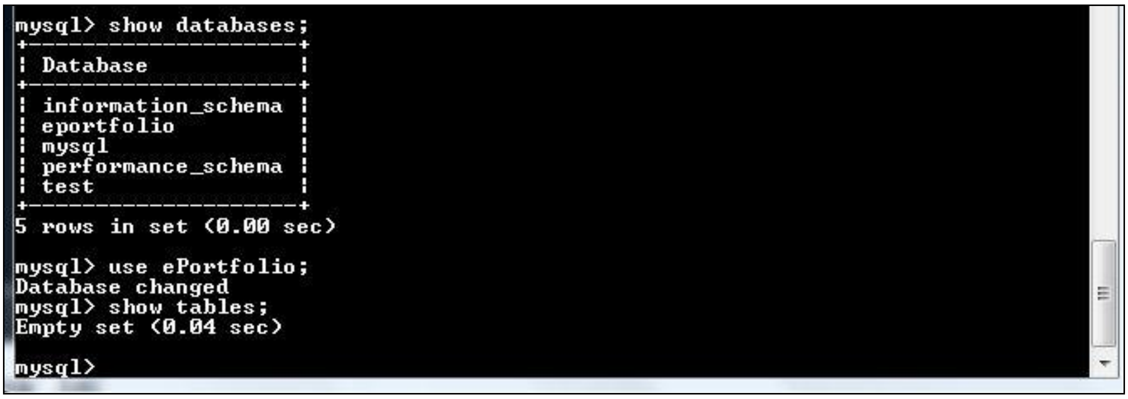
- Now it is possible to enter SQL-commands to the command line interface.
- To create a database use: *„create database databaseName“* (figure 3)



```
mysql> create database ePortfolio;
Query OK, 1 row affected (0.01 sec)

mysql>
```

figure 3: create Database

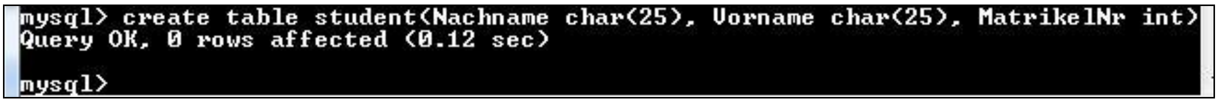
- Use *„show databases“* to display all existing databases.
- After entering the *„use ePortfolio“* command, only sql-commands for the database named ePortfolio are expected.
- The *„show tables“* command displays all tables of the database. (figure 4)



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| eportfolio         |
| mysql              |
| performance_schema |
| test               |
+--------------------+
5 rows in set (0.00 sec)

mysql> use ePortfolio;
Database changed
mysql> show tables;
Empty set (0.04 sec)

mysql>
```

figure 4: show databases and use database

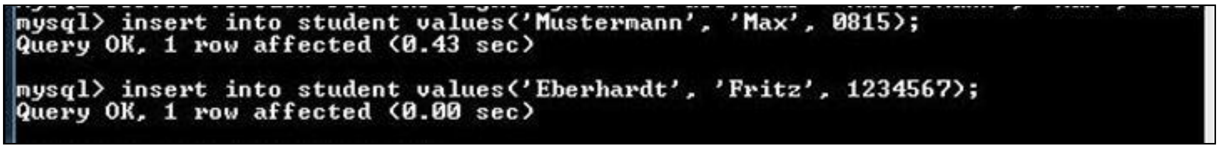
- Now you have to create tables and the columns of the table (figure 5)



```
mysql> create table student(Nachname char(25), Vorname char(25), MatrikelNr int)
Query OK, 0 rows affected (0.12 sec)

mysql>
```

figure 5: create table

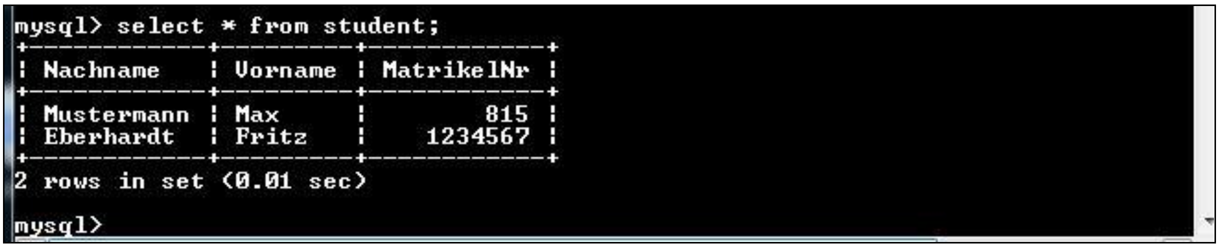
- To fill the tables you have to use the command in figure 6.



```
mysql> insert into student values('Mustermann', 'Max', 0815);
Query OK, 1 row affected (0.43 sec)

mysql> insert into student values('Eberhardt', 'Fritz', 1234567);
Query OK, 1 row affected (0.00 sec)
```

figure 6: insert into table

- To test the database use a simple sql-query shown in figure 7.

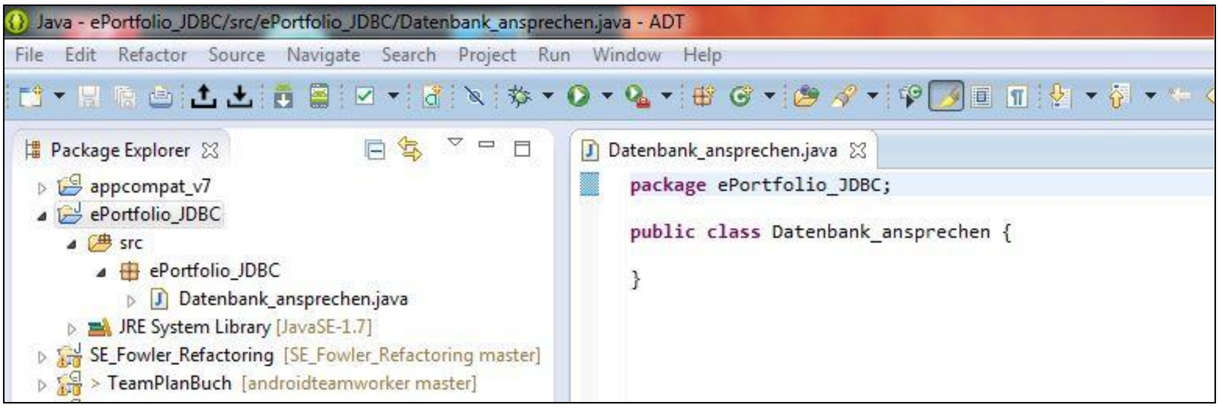


```
mysql> select * from student;
+------------+---------+------------+
| Nachname   | Vorname | MatrikelNr |
+------------+---------+------------+
| Mustermann | Max     |        815 |
| Eberhardt  | Fritz   |    1234567 |
+------------+---------+------------+
2 rows in set (0.01 sec)

mysql>
```

figure 7: Select query

2.4 Database connection with java eclipse

- Create a new java project or use an old project (figure 8).



Java - ePortfolio_JDBC/src/ePortfolio_JDBC/Datenbank_ansprechen.java - ADT
File Edit Refactor Source Navigate Search Project Run Window Help
Package Explorer
appcompat_v7
ePortfolio_JDBC
src
ePortfolio_JDBC
Datenbank_ansprechen.java
JRE System Library [JavaSE-1.7]
SE_Fowler_Refactoring [SE_Fowler_Refactoring master]
> TeamPlanBuch [androidteamworker master]
Datenbank_ansprechen.java

```
package ePortfolio_JDBC;

public class Datenbank_ansprechen {

}
```

figure 8: Projekt und Klasse anlegen

2.4.1 Download MySQL Connector/J

Download the java driver for the database connection.

Download Connector/J: http://dev.mysql.com/downloads/connector/j/

2.4.2 Datenbank-Treiber der Klassenbibliothek hinzufügen

- Import the driver to the class library of your project.
- Right click on the class
- Navigate to the configure build path (figure 9)

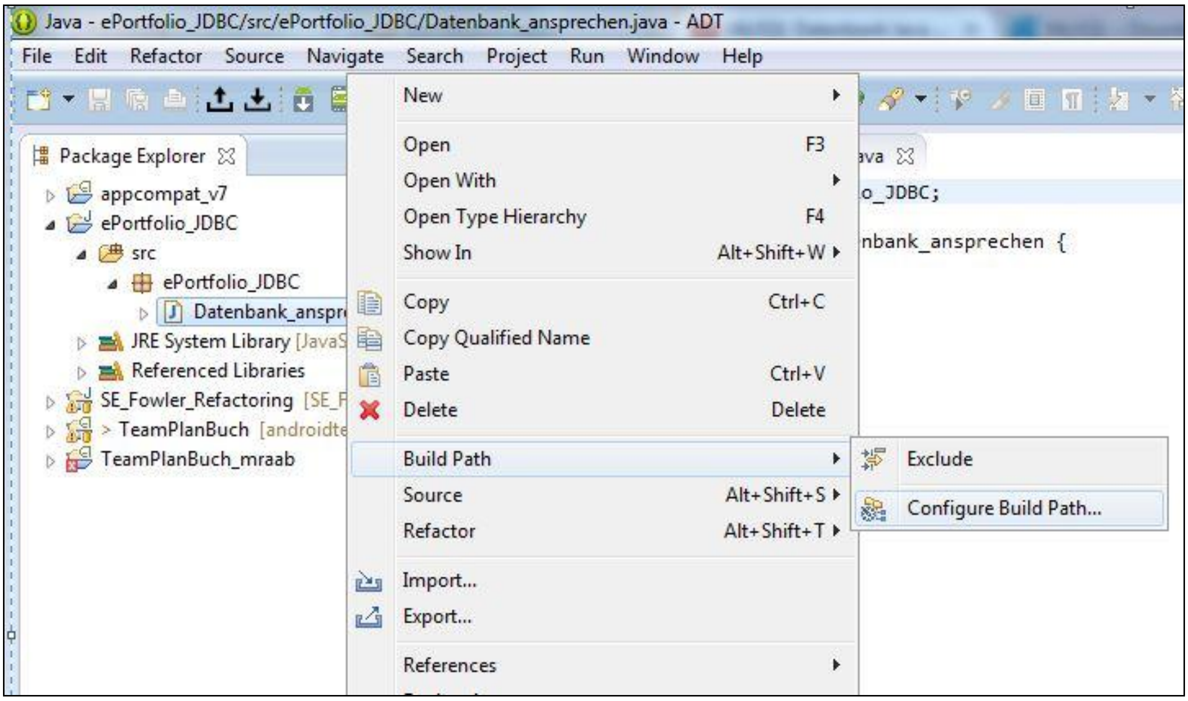
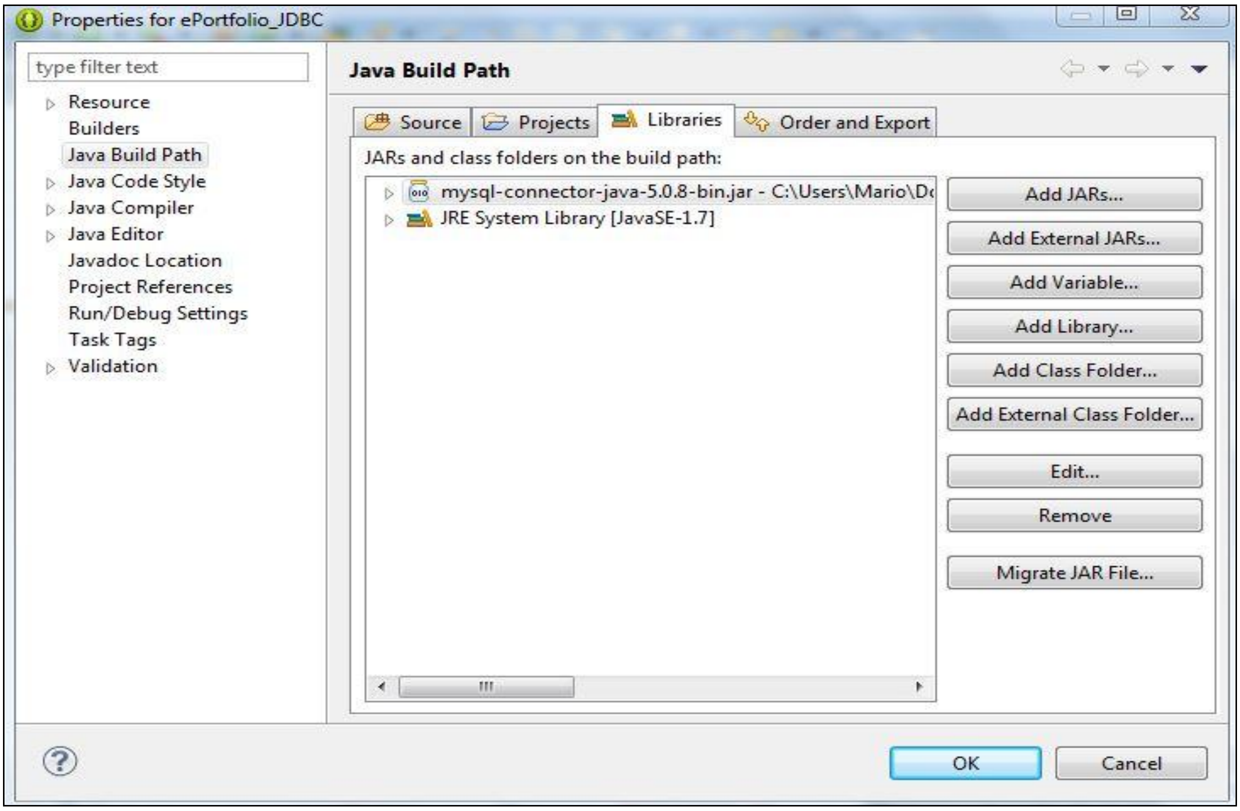


figure 9: Build Path

- Navigate in the task bar to „Java Build Path - Libraries“
- With “Add External JARs…“ you have to specify the path to the mysql-connector-java-bin.jar file. (figure 10)



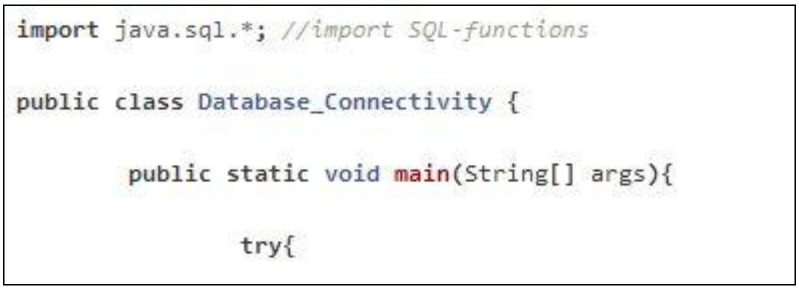
JARs and class folders on the build path:

- mysql-connector-java-5.0.8-bin.jar - C:\Users\Mario\D
- JRE System Library [JavaSE-1.7]

figure 10: Add External JARs…

2.5 Example of a simple sql-query (syntax)

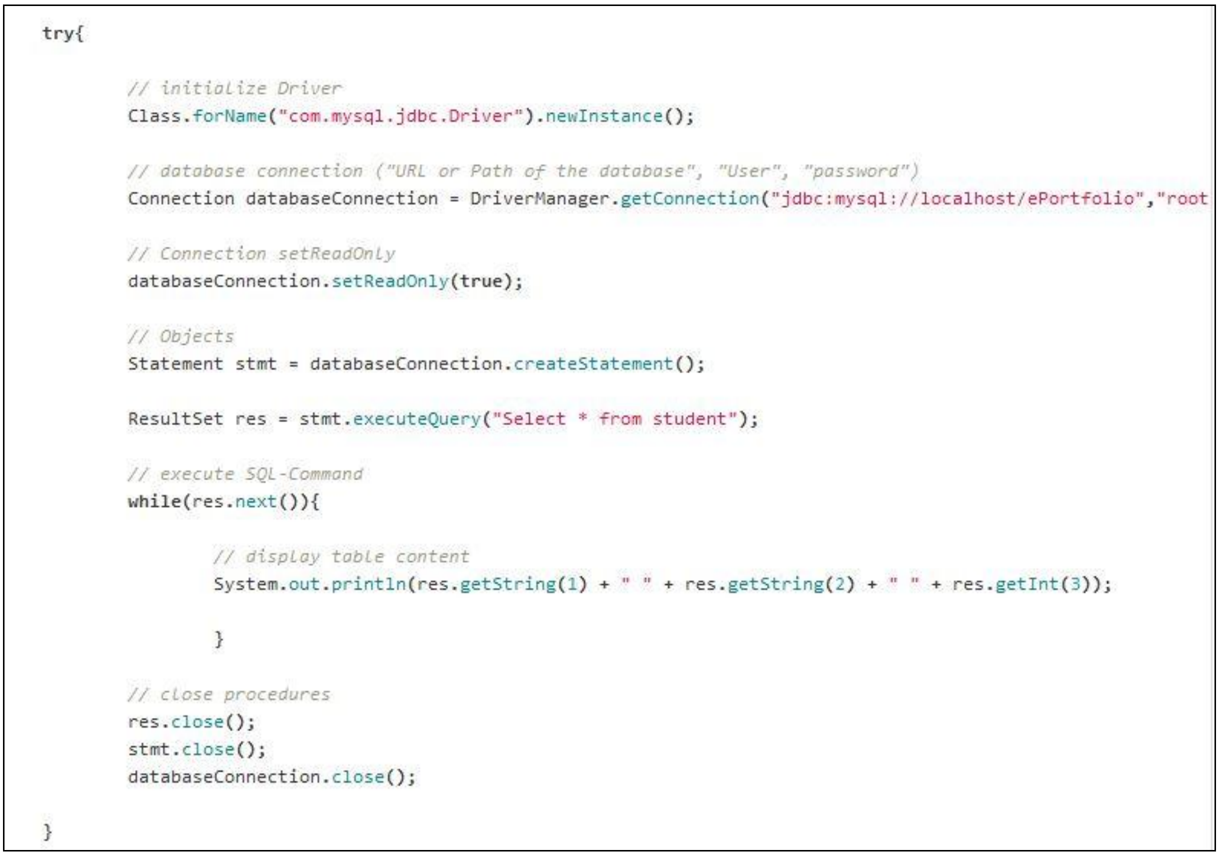
- Import the java-package „java.sql.*“, with this package you are able to use all sql-functions. (figure 11)



```
import java.sql.*; //import SQL-functions

public class Database_Connectivity {

        public static void main(String[] args){

                try{
```

figure 11: import java-package (sql-functions)

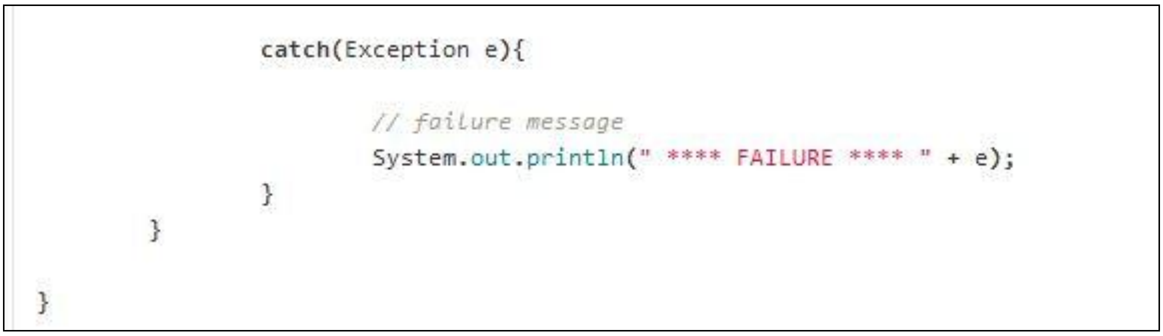
- Class.*forName*(”com.mysql.jdbc.Driver“).newInstance(); initializes the jdbc-driver.
- The database connection needs parameter („path“, “user“, “password“).
- The while-loop executes the select query. (figure 12)



```
try{

        // initialize Driver
        Class.forName("com.mysql.jdbc.Driver").newInstance();

        // database connection ("URL or Path of the database", "User", "password")
        Connection databaseConnection = DriverManager.getConnection("jdbc:mysql://localhost/ePortfolio","root

        // Connection setReadOnly
        databaseConnection.setReadOnly(true);

        // Objects
        Statement stmt = databaseConnection.createStatement();

        ResultSet res = stmt.executeQuery("Select * from student");

        // execute SQL-Command
        while(res.next()){

                // display table content
                System.out.println(res.getString(1) + " " + res.getString(2) + " " + res.getInt(3));

                }

        // close procedures
        res.close();
        stmt.close();
        databaseConnection.close();

}
```

figure 12: try

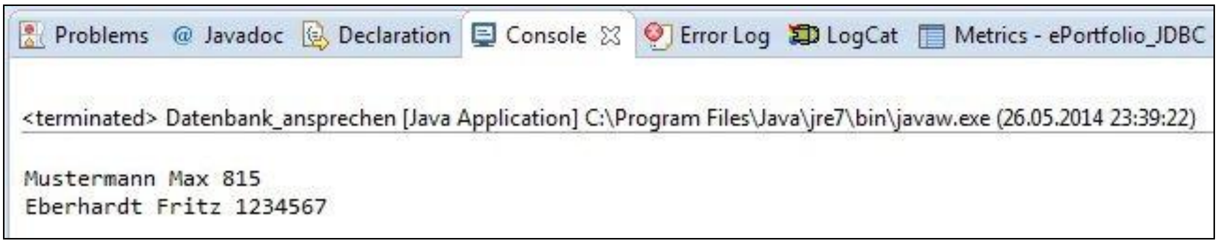
- Catch is executed if a failure occurs (figure 13)



```
                catch(Exception e){

                        // failure message
                        System.out.println(" **** FAILURE **** " + e);
                }
        }

}
```

figure 13: catch

- Output is shown in figure



```
<terminated> Datenbank_ansprechen [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (26.05.2014 23:39:22)

Mustermann Max 815
Eberhardt Fritz 1234567
```

figure 14: output

3 Forum

Link to Git Repository

- https://github.com/mraab89/Tutorial_Java_Database_Connectivity_JDBC

Link to forum stackoverflow

- http://stackoverflow.com/questions/17426052/error-2003-hy000-cant-connect-to-mysql-server-on-localhost-10061