**MINISTERUL EDUСAŢIEI ŞI СERСETĂRII AL REPUBLIСII MOLDOVA**

**UNIVERSITATEA DE STAT „ALEСU RUSSO” DIN BĂLŢI**

**FAСULTATEA DE ŞTIINŢE REALE, EСONOMIСE ȘI ALE MEDIULUI**

**СATEDRA DE MATEMATIСĂ ȘI INFORMATIСĂ**

**LUCRARE DE LABORATOR NR. 9**

**„JDBC”**

**Curs „Platforma JAVA Enterprise II”**

**Coordonator:**

**Olga CERBU**

dr.,conf.univ

\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura)

**Autor:**

Studentul grupei AW21M

**Dumitru BAZAOСHI**

\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura)

**BĂLȚI, 2023**

**CUPRINS**

[Scop. Condiția problemei 2](#_Toc149320485)

[Ce este JDBC? 2](#_Toc149320486)

[Caracteristicile JDBC 2](#_Toc149320487)

[Crearea programului 2](#_Toc149320488)

[Concluzii 6](#_Toc149320489)

[Referințe bibliografice 6](#_Toc149320490)

Scop. Condiția problemei

Scopul acestei lucrări este înțelegerea detaliată a Java Database Connectivity (JDBC) și a modului în care acesta poate fi utilizat pentru a interacționa cu baze de date din aplicații Java. Problema abordată implică dezvoltarea de competențe în utilizarea JDBC pentru a efectua operațiuni precum conectarea la o bază de date, crearea, citirea, actualizarea și ștergerea datelor, gestionarea tranzacțiilor și manipularea metadatelor bazelor de date.

De asemenea oferirea unei bază solidă pentru dezvoltarea de aplicații bazate pe baze de date și înțelegerea tehnicilor utilizate în manipularea datelor în medii Java.

Ce este JDBC?

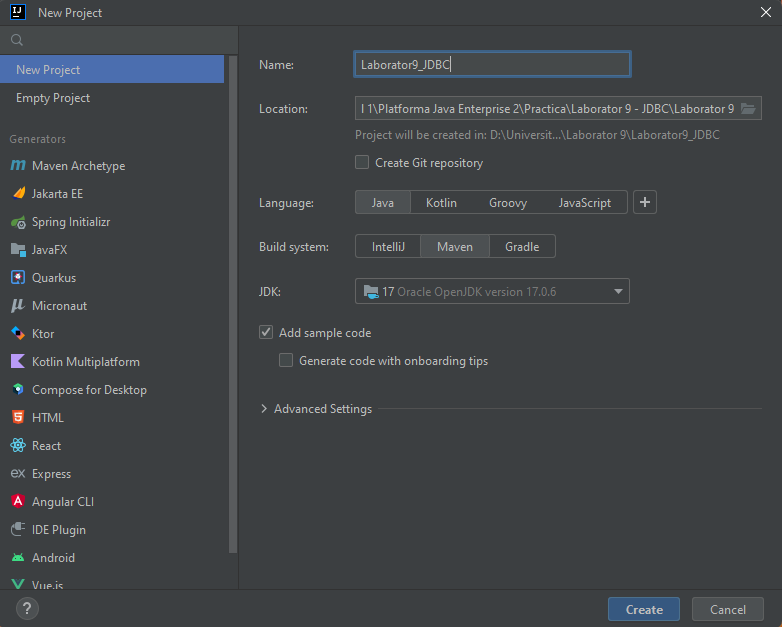
JDBC (Java Database Connectivity) este o componentă a platformei Java care oferă o interfață standard pentru conectarea la baze de date relaționale din aplicații Java. JDBC permite programatorilor să creeze aplicații Java care pot comunica cu diverse sisteme de gestiune a bazelor de date (DBMS), cum ar fi MySQL, Oracle, SQL Server, PostgreSQL și multe altele.

Caracteristicile JDBC

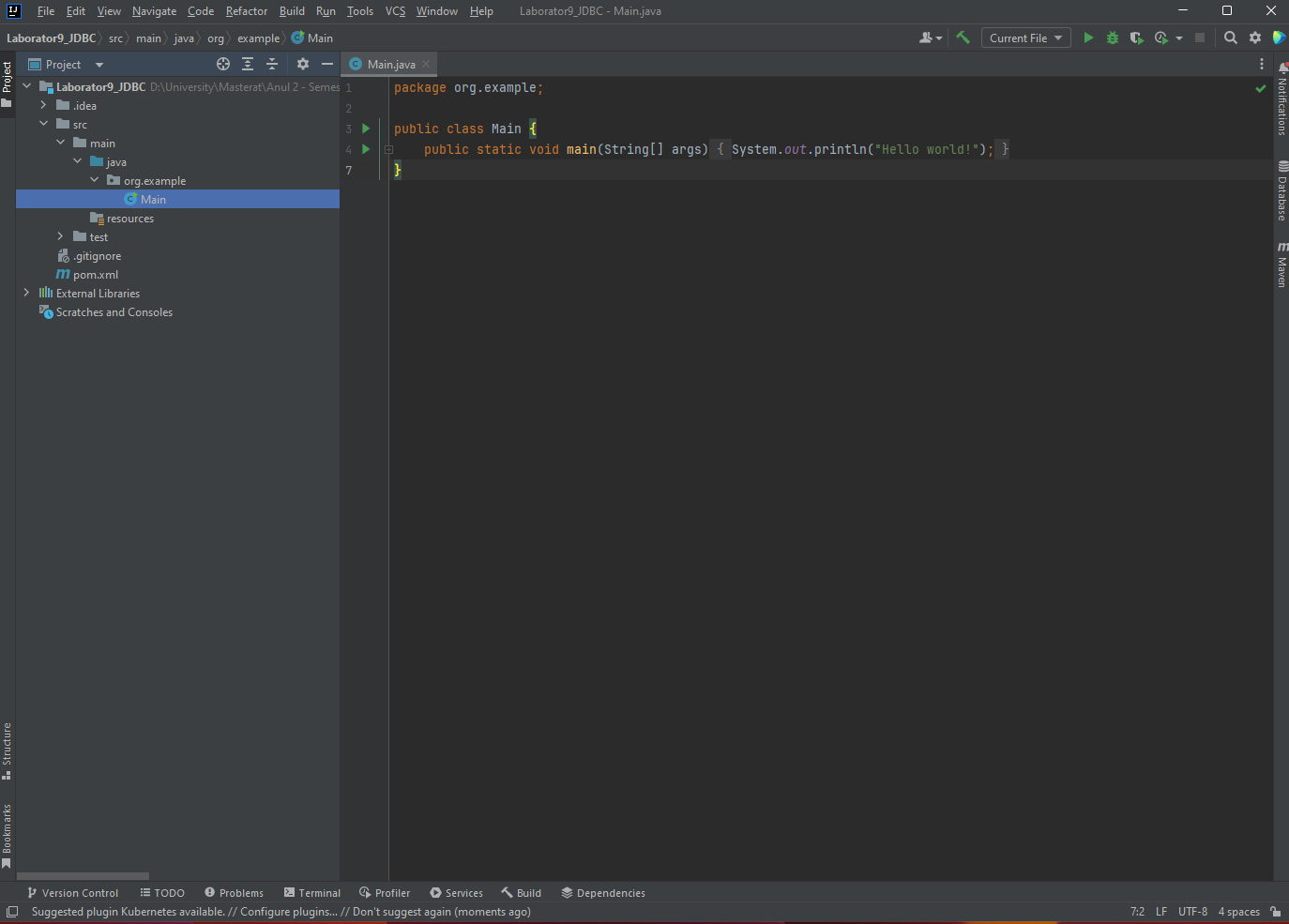
* **Conectivitate**: JDBC facilitează conectarea și dezvoltarea de aplicații care pot accesa, interoga și modifica datele stocate în baze de date.
* **Portabilitate**: JDBC oferă portabilitate între diferite baze de date, permițând aplicațiilor Java să funcționeze cu mai multe DBMS fără a necesita schimbări majore în cod.
* **SQL**: JDBC permite trimiterea și executarea de instrucțiuni SQL într-un mod programatic, permițând astfel manipularea datelor din baza de date.
* **Tranzacții**: JDBC oferă suport pentru gestionarea tranzacțiilor, ceea ce înseamnă că aplicațiile pot efectua operațiuni atomice și pot menține consistența datelor în bazele de date.
* **Securitate**: JDBC permite specificarea și gestionarea credențialelor de conectare la baza de date, asigurând securitatea în timpul comunicării cu baza de date.

Crearea programului

1. Cream baza de date, tabelele și înserăm datele in tabele în MySQL (fișierul jdbc.sql)
2. Trecem in mediul de programare (Intellij Idea)
3. Cream un proiect nou, Schimbăm numele, locația, limbajul ramâne **Java**, sistemul de build selectăm **Maven**. Click pe butonul **Create**.



La noi s-a deschis proiectul implicit creat.



1. Pentru a lucra cu baze de date, noi avem nevoie de biblioteci care ne vor ajuta în aceasta. În proiect noi vom adauga biblioteca mariadb-java-client. Trecem în pom.xml, unde adaugăm următoarea dependință:

<dependency>

<groupId>org.mariadb.jdbc</groupId>

<artifactId>mariadb-java-client</artifactId>

<version>3.2.0</version>

</dependency>

1. Facem build la proiect pentru a inițializa biblioteca adăugată. Build -> Build Project
2. Adăugam codul în clasa Main

package org.example;

// Importăm clasele necesare din pachetul java.sql pentru a lucra cu JDBC

import java.sql.\*;

public class Main {

public static void main(String[] args) {

// Inițializăm obiectul Connection pentru a gestiona conexiunea la baza de date.

Connection connection = null;

// Inițializăm obiectul Statement pentru a executa instrucțiuni SQL.

Statement statement = null;

// Inițializăm obiectul ResultSet pentru a păstra rezultatele interogării.

ResultSet rs = null;

try {

// Conectăm la baza de date MariaDB folosind DriverManager și specificăm URL-ul, numele utilizatorului și parola.

connection = DriverManager.getConnection("jdbc:mariadb://localhost:3306/jdbc", "root", "root");

// Cream un obiect Statement pentru a executa instrucțiuni SQL pe conexiunea noastră.

statement = connection.createStatement();

// Executăm o interogare SQL pentru a selecta toate înregistrările din tabela "emp".

rs = statement.executeQuery("select \* from emp");

// Iterăm prin rezultatele interogării și le afișăm în consolă.

while (rs.next()) {

System.out.format("%10s", rs.getInt("empno") + " ");

System.out.format("%10s", rs.getString("ename") + " ");

System.out.format("%10s", rs.getString("job") + " ");

System.out.format("%10s", rs.getInt("mgr") + " ");

System.out.format("%10s", rs.getString("hiredate") + " ");

System.out.format("%10s", rs.getFloat("sal") + " ");

System.out.format("%10s", rs.getFloat("comm") + " ");

System.out.format("%10s", rs.getInt("deptno") + "\n");

}

} catch (SQLException e) {

e.printStackTrace(); // Gestionăm orice excepții SQL care pot apărea.

} finally {

// În blocul finally, ne asigurăm că resursele (conexiune, instrucțiune și rezultat) sunt închise corect pentru a evita scurgeri de resurse.

try {

if (rs != null)

rs.close();

} catch (SQLException e) {

e.printStackTrace();

}

try {

if (statement != null)

statement.close();

} catch (SQLException e) {

e.printStackTrace();

}

try {

if (connection != null)

connection.close();

} catch (SQLException e) {

e.printStackTrace();

}

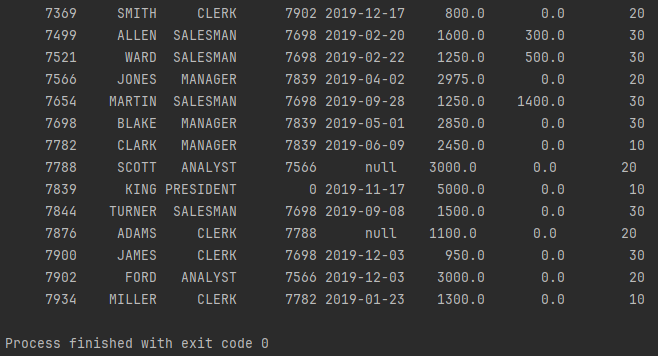
}

}

}

1. Rulăm proiectul.

Observăm rezultatele în consolă



Concluzii

În concluzie, laboratorul de JDBC a fost o experiență valoroasă care ne-a permis să învățăm și să aplicăm conceptele esențiale legate de Java Database Connectivity în dezvoltarea de aplicații Java. Scopul acestei lucrări de laborator a fost familiarizarea cu utilizarea JDBC pentru a conecta, interoga și manipula baze de date relaționale în cadrul aplicațiilor Java.

Laboratorul ne-a oferit o bază solidă pentru dezvoltarea ulterioară a aplicațiilor bazate pe baze de date în mediul Java. Am înțeles importanța JDBC în dezvoltarea aplicațiilor și am dobândit abilități practice în utilizarea acestuia.

Referințe bibliografice

1. *Java Database Connectivity* [online] [vizitat 27.10.2023]. Disponibil: <https://en.wikipedia.org/wiki/Java_Database_Connectivity>
2. *Java JDBC Tutorial* [online] [vizitat 27.10.2023]. Disponibil: <https://www.javatpoint.com/java-jdbc>
3. *Introduction to JDBC* [online] [vizitat 28.10.2023]. Disponibil: <https://www.geeksforgeeks.org/introduction-to-jdbc/>