PRIMER – JDBC

FAKULTET

Šema relacione baze podataka je:

SKOLSKA_GODINA (IDSko)
KATEDRA (IDKat, Naziv, IDRad)
RADNIK (IDRad, Ime, Staz, MatBr)
NASTAVNIK (IDRad, Zvanje)
NENASTAVNIK (IDRad, Posao)
PREDMET (IDPre, Naziv, Godina, Semestar, TrebaUslov)
STUDENT (IDStu, Ime, Status, Godina, BrojOcena, ZbirOcena)
SAMOFINASIRAJUCI (IDStu, Procenat, IDRad)
ROK (IDRok, Naziv, DatumOd, DatumDo, Status, IDSko, ProsekOcena)
ISPIT (IDIsp, Datum, Vreme, IDRok, IDPre)
PRIJAVA (IDIsp,IDStu, Datum)
POLAGANJE (IDIsp,IDStu, Status, Datum, Ocena, IDRad)
SLUSANOST (IDStu,IDPre, IDGod)
USLOV (IDStu,IDPre)
PRIPADA_P (IDPre, IDKat)
PRIPADA_N (IDNas, IDKat)
PREDAJE (IDPre,IDRad)

Sastaviti Java metodu koja koristeći JDBC implementira metodu PrijavaJednogIspita kojom se za jednog studenta vrši prijava jednog ispita, uz sve neophodne prethodne provere. Pri tome se zadaju ID-ovi postojećeg studenta i postoje}eg predmeta. Pretpostaviti da konekcija već uspostavljena.

```
import java.sql.*;
import java.util.*;

public class Fakultet{

	Connection dbConnection = null;

	public Fakultet(){}

	public booelan Init(){

			try {
				Class.forName("net.sourceforge.jtds.jdbc.Driver");

			}
			catch (ClassNotFoundException e){
				e.printStackTrace(System.out);
				return false;
			}

			try {
				dbConnection =
DriverManager.getConnection("jdbc:jtds:sqlserver://66.138.86.217:3071/testdb", "testdata",
"sa");
			}
			catch (SQLException e) {
				e.printStackTrace(System.out);
				return false;
			}

			return true;
	}

	public boolean EndSession(){dbConnection.close(); dbConnection = null;}


	public boolean PrijavaJednogIspita(int IDStu, int IDPre){
		int IDRok, IDIsp;
		boolean Ishod;

		ProveraOtvorenostiRoka(IDRok, Ishod);
		If (Ishod){
			IDIsp = OdredjivanjeIspita(IDRok, IDPre);
			Ishod = ProveraSlusanosti(IDStu, IDPre);
			If (Ishod){
				Ishod = ProveraUslova(IDStu, IDPre);
				If (Ishod){
					Ishod = ProveraNemaPolozeno(IDStu, IDPre);
					If (Ishod){
						try {

						dbConnection.setAutoCommit(false);

dbsetTransactionIsolation(Connection.TRANSACTION_SERIALIZABLE);

						PreparedStatement dbPreStmt =
						dbConnection.prepareStatement("INSERT INTO PRIJAVA
						VALUES(?, ?, ?)");
						dbPreStmt.setInt(1, IDIsp);
						dbPreStmt.setInt(2, IDStu);
						dbPreStmt.setTimestamp(3, new Timestamp((new
Date()).getTime()));
```

```
                                        int rowCount = dbPreStmt.executeUpdate();
                                        System.out.println("rowCount=" + rowCount);

                                        dbConnection.commit();
                                        dbConnection.setAutoCommit(true);

                                        dbPreStmt.close();
                                        return true;
                                  }
                                  catch (SQLException e) {
                                        e.printStackTrace(System.out);
                                        dbConnection.rollback();
                                        dbConnection.setAutoCommit(true);
                                        dbPreStmt.close();
                                        return false;
                                  }
                            }else {System.out.println("Vec polozeno"); return false;}
                      }else {System.out.println("Nema uslov"); return false;}
                }else {System.out.println("Nije slusano"); return false;}
          }else {System.out.println("Nema roka"); return false;}
    }

    public void ProveraOtvorenosti(int IDRok, boolean Ishod){
          IDRok = null;
          Ishod = false;
          try{
                Statement dbStmt = dbConnection.createStatement();

                ResultSet resultSet = dbStmt.executeQuery("SELECT IDRok FROM ROK WHERE
                Status = 'o'");

                while(resultSet.next()){
                      IDRok = resultSet.getInt(1);
                      Ishod = true;
                }
                resultSet.close();
                dbStmt.close();
          }
          catch (SQLException e) {
                e.printStackTrace(System.out);
                if (resultSet) resultSet.close();
                dbStmt.close();
          }
    }

    public int OdredjivanjeIspita(int IDRok, int IDPre){
          int IDIsp = 0;
          try {
                PreparedStatement dbPreStmt = dbConnection.prepareStatement("SELECT
                IDISp FROM ISPIT WHERE IDRok = ? AND IDPre = ?");

                dbPreStmt.setInt(1, IDRok);
                dbPreStmt.setInt(2, IDPre);

                ResultSet resultSet = dbPreStmt.executeQuery();
                while(resultSet.next()){
                      IDIsp = resultSet.getInt(1);
                }
                resultSet.close();
                dbPreStmt.close();
                return IDIsp;
          }
          catch (SQLException e) {
```

```
            e.printStackTrace(System.out);
            if (resultSet) resultSet.close();
            dbPreStmt.close();
            return IDIsp;
      }
}


public boolean ProveraSlusanosti(int IDStu, int IDPre){
      try {
            PreparedStatement dbPreStmt = dbConnection.prepareStatement("SELECT *
            FROM SLUSANOST WHERE IDStu= ? AND IDPre= ? ");

            dbPreStmt.setInt(1, IDStu);
            dbPreStmt.setInt(2, IDPre);

            ResultSet resultSet = dbPreStmt.executeQuery();
            while(resultSet.next()){
                  dbPreStmt.close();
                  return true;
            }
            resultSet.close();
            dbPreStmt.close();
            return false;
      }
      catch (SQLException e) {
            e.printStackTrace(System.out);
            if (resultSet) resultSet.close();
            dbPreStmt.close();
            return false;
      }
}

public boolean ProveraUslova(int IDStu, int IDPre){
boolean Ishod = false, trebaUslov = false;
try {
            PreparedStatement dbPreStmt = dbConnection.prepareStatement("SELECT
            TrebaUslov FROM PREDMET WHERE IDPre= ? ");

            dbPreStmt.setInt(1, IDPre);

            ResultSet resultSet = dbPreStmt.executeQuery();
            while(resultSet.next()){
                  trebaUslov = ResultSet.getBoolean(1);
            }
            resultSet.close();
            dbPreStmt.close();

            dbPreStmt = dbConnection.prepareStatement("SELECT * FROM USLOV WHERE
            IDStu= ? AND IDPre= ? ");
            dbPreStmt.setInt(1, IDStu);
            dbPreStmt.setInt(2, IDPre);

            ResultSet resultSet = dbPreStmt.executeQuery();
            while(resultSet.next()){
                  Ishod = true;
            }
            resultSet.close();
            dbPreStmt.close();
            return Ishod;

      }
      catch (SQLException e) {
```

```
            e.printStackTrace(System.out);
            if (resultSet) resultSet.close();
            dbPreStmt.close();
            return Ishod;
      }
}

public boolean ProveraNemaPolozeno(int IDStu, int IDPre){
try {
            PreparedStatement dbPreStmt = dbConnection.prepareStatement("SELECT *
            FROM POLAGANJE P, ISPIT I WHERE P.IDStu = ? AND P.Ocena > 5 AND
            P.Status = 'V' AND P.IDIsp = I.IDIsp AND I.IDPre = ? ");

            dbPreStmt.setInt(1, IDStu);
            dbPreStmt.setInt(2, IDPre);

            ResultSet resultSet = dbPreStmt.executeQuery();
            while(resultSet.next()){
                  dbPreStmt.close();
                  return false;
            }
            resultSet.close();
            dbPreStmt.close();
            return true;
      }
      catch (SQLException e) {
            e.printStackTrace(System.out);
            if (resultSet) resultSet.close();
            dbPreStmt.close();
            return false;
      }
}
```