I. Készítsen egy konzolos és grafikus alkalmazást Java vagy C# nyelven, amellyel egy könyvtár

működését segíti. Megoldását töltse fel a sb2vr2o kurzus kódú classroomba! A szükséges

forrásokat is itt találja!

1) A konzolos alkalmazás a paraméterként kapott állományok adatait importálja egy

adattáblába (teszt adatok: Kolcsonzesek.csv és Kolcsonzok.csv).

Az adathalmazok pontosvesszővel elválasztva UTF-8 kódolással tartalmazza az adatokat!

Kolcsonzok.csv:

Név;Születési dátum

Kovács Ádám;1990-05-15

…

Kovács Márton;1981-12-19

Kolcsonzesek.csv:

KolcsonzoId;Író;Műfaj;Cím

1;J.K. Rowling;Fantasy;Harry Potter és a Bölcsek Köve

…

10;Michael Crichton;Sci-fi;Jurassic Park

1) Hozzon létre egy Ön által választott adatbáziskezelő-rendszerben egy adatbázist és

abban egy-egy adattáblát, amely alkalmas az adatok tárolására.

Kolcsonzok{

id integer($int32)

nev string; nullable: true

szulIdo string($date-time)

kolcsonzesek [

nullable: true

readOnly: true

Kolcsonzesek{

id integer($int32)

kolcsonzokId integer($int32)

iro string

nullable: true

mufaj string

nullable: true

cim string

nullable: true

}]

}

Kolcsonzesek{

id integer($int32)

kolcsonzokId integer($int32)

iro string

nullable: true

mufaj string

nullable: true

cim string

nullable: true

}

**megoldás:**

Hozzunk létre egy java projektet Vscode-al.

* nyissuk meg a Vscode-ot a feladat mappájában
* F1 -> Java: Create Java Project
* válasszuk a No build tools-t
* Katt a Select Project Locationra
* adjuk meg a projekt nevét
* zárjuk be a hátsó Vscode-ot

Szükségünk lesz a **mariadb-java-client-3.3.3.jar** könyvtárra. Ha nincs meg töltsük le és másoljuk be a projekt *lib*/ könyvtárába.

Nyissuk meg az *src/* mappát és dupla kattintás az App.java fájlra.

Alul a java projects sorban ellenőrizzük, hogy importálta-e a mariadb-java-client-3.3.3.jar fájlt. Ha nem akkor tegyük meg manuálisan.

Csináljuk meg az adatbázist.

* hozzunk létre a gyökérkönyvtárban egy *database* mappát, abban pedig egy *create.sql* fájlt.
* a fájlban írjuk be a következő sorokat:
* create database konyvtar;
* use konyvtar;
* create table kolcsonzok (
* id int not null primary key auto\_increment,
* nev varchar(50) not null,
* szulIdo date not null
* );
* create table kolcsonzesek(
* id int not null primary key auto\_increment,
* kolcsonzoId int not null,
* iro varchar(50) not null,
* mufaj varchar(50) not null,
* cim varchar(50) not null
* )
* nyissuk meg a XAMPP-ot, indítsuk el az *Apache* és *MySQL* szervereket. A böngészőben a localhost/phpmyadmint-t.
* Kattintsunk az SQL fülre és másoljuk be a *create.sql* tartalmát.
* /ha hibát jelez, de az adatbázis létrejött akkor a *create database konyvtar* sort töröljük ki az SQL lekérdezések ablakból/.

2) Ha alkalmazás nem kapott parancssori indításkor paramétert, vagy a paraméterként

megadott helyen és/vagy néven nem található állomány, tájékoztassa a helyes

paraméterezés szintaktikájáról a felhasználót és fejezze be a futást.

Másoljuk be a projekt gyökérkönyvtárába a *Kolcsonzesek.csv* és a *Kolcsonzok.csv* fájlokat.

Kattintsunk a *.vscode* mappára, majd a menűben a *Run -> Add Configuration*-ra.

Létrejött egy *launch.json* fájl.

a *„projectName”: „a projekt neve”*sor után írjuk be: /vesszőt ne felejtsük el/

{

            "type": "java",

            "name": "App",

            "request": "launch",

            "mainClass": "App",

            "projectName": "library\_console\_2f225b47",

            "args": [

                "Kolcsonzok.csv",

                "Kolcsonzesek.csv"

            ]

        }

Nyissuk meg az *App.java* fájlt.

Készítsünk egy metódust, amivel ellenőrizzük a paraméterek számát.

private static void checkArgsNumber(String[] args) {

        if (args.length != 2) {

            System.err.println("Hiba! Nincs két paraméter!");

            System.exit(1001);

        }

    }

A main függvénybe írjuk be a függvény hívást:

checkArgsNumber(args);

Írjunk egy másik függvényt, ami ellenőrzi a fájlok meglétét.

private static void checkFileIsExist(String[] args) {

        File param1 = new File(args[0]);

        File param2 = new File(args[1]);

        if (!param1.exists() || !param2.exists()) {

            System.err.println("Hiba! A fájl(ok) nem található(ak)!”);

            System.exit(1002);

        }

    }

Hozzuk létre a paraméterek osztályait:

Az *src* mappába két új fájl: *Kolcsonzok.java*  és *Kolcsonzesek.java*

A csv fájlokból megtudjuk hogy milyen adattagokat tartalmaznak az osztályok, és ennek megfelelően hozzuk létre.

Írjuk meg az Kolcsonzok osztályt:

public class Kolcsonzok {

    String nev;

    String szulIdo;

}

és a Kolcsonzesek-et is:

public class Kolcsonzesek {

    int kolcsonzoId;

    String iro;

    String mufaj;

    String cim;

}

Hozzunk létre két *ArrayList*-et amik tárolják a Kolcsonzok és a Kolcsonzesek példányait.

A main függvény elé:

static ArrayList<Kolcsonzok> kolcsonzokLista = new ArrayList<>();

static ArrayList<Kolcsonzesek> kolcsonzesekLista = new ArrayList<>();

Írjuk meg a függvényeket, ami feltölti a listákat.

private static void readFile1(String path) throws FileNotFoundException {

        File file = new File(path);

        Scanner sc = new Scanner(file);

        sc.nextLine();

        while (sc.hasNextLine()) {

            String line = sc.nextLine();

            String[] parts = line.split(";");

            Kolcsonzok kolcsonzok = new Kolcsonzok();

            kolcsonzok.nev = parts[0];

            kolcsonzok.szulIdo = parts[1];

            kolcsonzokLista.add(kolcsonzok);

        }

    }

és

private static void readFile2(String path) throws FileNotFoundException {

        File file = new File(path);

        Scanner sc = new Scanner(file);

        sc.nextLine();

        while (sc.hasNextLine()) {

            String line = sc.nextLine();

            String[] parts = line.split(";");

            Kolcsonzesek kolcsonzesek = new Kolcsonzesek();

            kolcsonzesek.kolcsonzoId = Integer.parseInt(parts[0]);

            kolcsonzesek.iro = parts[1];

            kolcsonzesek.mufaj = parts[2];

            kolcsonzesek.cim = parts[3];

            kolcsonzesekLista.add(kolcsonzesek);

        }

    }

Hozzunk létre egy DataService osztályt. -> *src/DataService.java*

Írjuk meg a metódust, ami kapcsolódik az adatbázishoz.

private Connection getConnection() throws SQLException {

        String user = "root";

        String password = "";

        String url = "jdbc:mariadb://localhost:3306/konyvtar";

        return DriverManager.getConnection(url, user, password);

    }

és a másik két metódust, ami feltölti az adatbázisba a fájlok tartalmát:

 public void insertDataKolcsonzok(ArrayList<Kolcsonzok> kolcsonzokLista) {

        Connection connection = null;

        try {

            connection = getConnection();

            String sql = "INSERT INTO kolcsonzok (nev, szulIdo) VALUES (?, ?)";

            int szamlalo = 0;

            for (Kolcsonzok kolcsonzok : kolcsonzokLista) {

                PreparedStatement preparedStatement = connection.prepareStatement(sql);

                preparedStatement.setString(1, kolcsonzok.nev);

                preparedStatement.setString(2, kolcsonzok.szulIdo);

                preparedStatement.execute();

                szamlalo++;

            }

            connection.close();

            System.out.println(szamlalo + " adat rogzítve a kolcsonzok tablaban");

        } catch (SQLException e) {

            System.err.println("Hiba: " + e.getMessage());

        }

    }

    public void insertDataKolcsonzesek(ArrayList<Kolcsonzesek> kolcsonzesekLista) {

        Connection connection = null;

        try {

            connection = getConnection();

            String sql = "INSERT INTO kolcsonzesek (kolcsonzoId, iro, mufaj, cim) VALUES (?, ?, ?, ?)";

            int szamlalo = 0;

            for (Kolcsonzesek kolcsonzesek : kolcsonzesekLista) {

                PreparedStatement preparedStatement = connection.prepareStatement(sql);

                preparedStatement.setInt(1, kolcsonzesek.kolcsonzoId);

                preparedStatement.setString(2, kolcsonzesek.iro);

                preparedStatement.setString(3, kolcsonzesek.mufaj);

                preparedStatement.setString(4, kolcsonzesek.cim);

                preparedStatement.execute();

                szamlalo++;

            }

            connection.close();

            System.out.println(szamlalo + " adat rögzítve a kolcsonzesek tablaban");

        } catch (SQLException e) {

            System.err.println("Hiba: " + e.getMessage());

        }

    }

Kész az első feladat.