OBJEKTUMORIENTÁLT PROGRAMOZÁS
11. GYAKORLAT

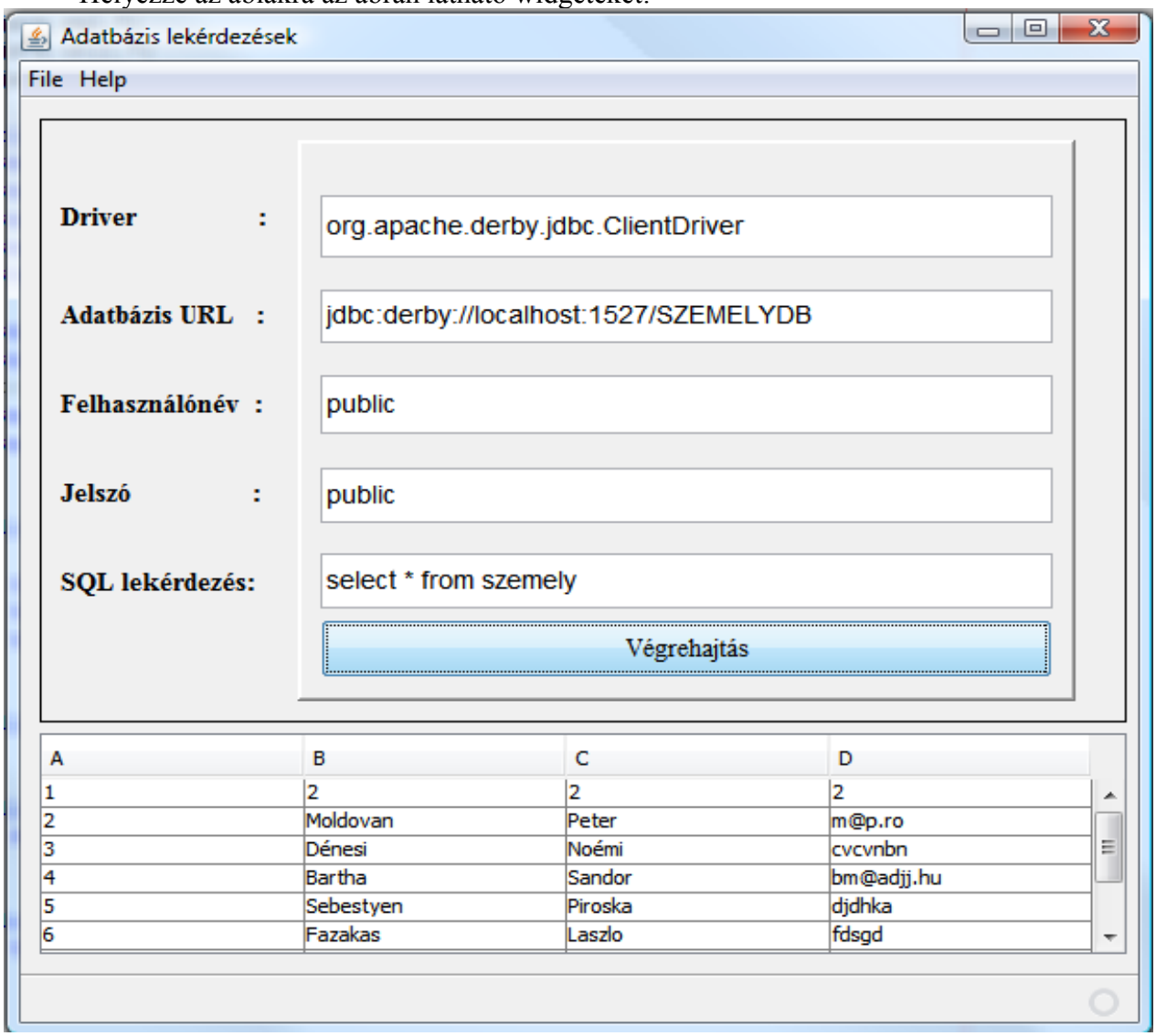
Célok:

- *Grafikus interfészek készítése*
- *JDBC – Java adatbáziskezelés*
- *JTable, TableModel*

1. feladat GUI készítése

- Készítsen egy **Java Desktop Application** típusú projektet.

- Helyezze az ablakra az ábrán látható widgeteket:



2. feladat Adatbázisszerver indítása

```
      Runtime window (felület bal felső sarkában levő ablak: Projects, Files,
Services ... )
      ---> Services --->Databases ---> JavaDB  jobb egérkattintás és Start
Server
```

OBJEKTUMORIENTÁLT PROGRAMOZÁS
11. GYAKORLAT

---

3. feladat Adatbázisdriver, modell elemek: QueryTableModel

3.1 A Derby JDBC driver hozzáadása a projekthez.

`Project – Properties –Libraries – Add Jar/Folder`

Keresse meg a fájlrendszerben a **derbyclient.jar** állományt!
(egy javadb/lib mappában kell lennie)

3.2 QueryTableModel

A **JTable** widget tartalmát egy táblamodell fogja meghatározni. Ebben az osztályban kapcsolódunk az adatbázishoz, és ugyancsak itt fogjuk végrehajtani a lekérdezéseket is.

http://www.ms.sapientia.ro/~manyi/teaching/oop/QueryTableModel.java

3.3 táblamodell példányosítása

A GUI osztályba helyezze el a következő sort:

```
private QueryTableModel qtm = new QueryTableModel();
```

A GUI Designer segítségével kösse össze a JTable komponenst a modelljével (qtm).

4. feladat Eseménykezelés

A **Végrehajtás** címkéjű nyomógomb lenyomása eredményezze a tábla feltöltését a lekérdezés eredményével! Amennyiben adatbázis kapcsolódási hiba vagy SQL parancs hiba van, jelenjen meg egy üzenetdoboz a megfelelő üzenettel!

5. feladat JTable fejléc

A lekérdezés után a fejléc tartalmazza a ResultSet metaadatait (a lekérdezésben szereplő mezőneveket)

```
this.qtm.doQuery("select * from szemely");
JTableHeader th = jTable1.getTableHeader();
TableColumnModel tcm = th.getColumnModel();

String[] headers = qtm.getHeaders();
for (int i = 0; i < headers.length; ++i) {
    TableColumn tc = tcm.getColumn(i);

    tc.setHeaderValue( headers[ i ]);
    th.repaint();
}
```