# Mérési jegyzőkönyv – Szoftver Laboratórium 5

3. mérés: JDBC

Név:	Kovács Levente Ákos
Neptun kód:	CM6UKU
Feladat kódja:	34-TOZSDE
Mérésvezető neve:	Ádám Anna Kinga
Mérés időpontja:	2016-03-31 10:15-12:00
Mérés helyszíne:	HSZK N
A működő alkalmazás elérhetősége:	http://rapid.eik.bme.hu/~cm6uku/jdbc
Megoldott feladatok:	1,2,3,4,5
Elérhető pontszám (plusz pontok nélkül):	50p

#### Felhasználói útmutató

Az alkalmazás elindítása után szükséges beírni a két felső dobozba a bejelentkezési adatokat

Felhasználónév: cm6uku

Jelszó: Aa36912151

A guin 4 fül látható:

Az 1. fülön (**Log**) az alkalmazás használatakor keletkező loggolt információk olvashatóak.

A 2. fülön (**Search**) a tranzakciók listázása végezhető, ha a szöveg beviteli mezőt üresen hagyjuk akkor kilistáz minden tranzakciót, ha kitöltjük egy 4 jegyű számmal akkor adott évi tranzakciókat listáz ki.

A 3. fülön (**Edit**) az adat bevitel és frissítés végezhető. Minden mezőt jó formátumú adatokkal kell feltölteni, ha az adatbázisban van már a megadott részvénytípus akkor frissíti ellenkező esetben beszúr egy új rekordot, ha az autocommit N-re (kikapcsolt) van állítva akkor beilleszt egy tranzakciót is a megadott befektető id-vel. A változtatás véglegesítéséhez a commit gombot kell megnyomni ha az auto commit ki volt kapcsolva legutóbb.

A 4.fülön (Statistics) a gomb megnyomásával statisztikai adatokat lehet listázni a tranzakciókról.

# Mérési feladatok megoldása

### 1. feladat

# A megoldáshoz használt SQL utasítás

Listázáshoz szükséges sql kód:

```
SELECT b.NEV,rt.CEGNEV,t.MENNYISEG,t.EGYSEGAR
FROM Tranzakcio t
inner join Reszvenytipus rt on rt.RTID=t.RTID
inner join Befekteto b on b.BEFEKTETOID=t.BEFEKTETOID
```

.

Évszerinti listázáshoz szükséges sql kód:

```
SELECT b.NEV, rt.CEGNEV, t.MENNYISEG, t.EGYSEGAR FROM Tranzakcio t inner join Reszvenytipus rt on rt.RTID=t.RTID inner join Befekteto b on b.BEFEKTETOID=t.BEFEKTETOID where EXTRACT(YEAR FROM t.Datum)=?
```

### Magyarázat

A feladat megoldásához a mintának megadott search fület használtam, annyi változtatással, hogy egy label-t raktam a szövegdoboz elé, ami felhívja a figyelmet a 2 funkcióra. A megoldáshoz a view osztály searcheventhandler()-jét, továbbá a controll és a model osztály search metódusát kellett implementálni, ezek a metódusok ebben a sorrendben hivják egymást. A model.search(...) felel azért hogy adatbázisból kinyerje az adatokat ezeket vissza adja a controllnak, ha szükséges a ? helyére az keresési évet helyettesíti be. A controll osztályban végzek bemenet validációt hogy a megadott keyword csak 4 jegyű szám lehessen és itt végzem a result set átalakítását String listává, így a viewnak nem kell tudnia semmit a visszakapot értékekről a kilistázáshoz. A viewnak visszadott sorokat kirakom a search fülön lévő táblázatba.

#### 2. feladat

### A megoldáshoz használt SQL utasítás

```
Egyediség vizsgálathoz:
```

```
SELECT rtid
FROM RESZVENYTIPUS rt
WHERE
```

rtid=?

#### Beillesztéshez:

```
INSERT INTO RESZVENYTIPUS
VALUES(? ,?, ?, ? ,?)
```

#### Updatehez:

WHERE rtid=?

# Magyarázat

A feladat megoldásához felvettem új mezőket (6 mezőt) az fxml fájlba és ezeket a megfelelő annotációval változóként a view osztályba, ezek a részvénytípus adatainak beviteléhez szükségesek.

A viewban lévő editeventhandler()-t kellett implementálni, ez meghívta a controll osztályt modifyData fügvényét átadva a textfieldekben lévő adatokból képzett mapet. A controll validálta az texfieldbe beírt adatokat, majd ha ez sikerült, meghívta a model osztály modifyData függvényét. A model osztály megnézte, hogy létezik e már ilyen részvénytípus egy lekérdezés segítségével, amennyiben a lekérdezés nem adott

vissza adatot, akkor beszurást hajtott végre a program, ha talált ilyen sort, akkor pedig frissítést hajtott végre, ezekhez a mapként megkapott adatokat parseolta és a ?-hez rendelte. A visszatérési értékből a controll tudta hogy milyen esemény történt és ez megfelelően logolta.

#### 3. feladat

### A megoldáshoz használt SQL utasítás

Nem kellett

### Magyarázat

A feladat elvégzéséhez a controller.verifyData(...) metódust kellet megvalósítani reguláris kifejezések segítségével.

3 tipusu adatot kellett vizsgálnom,

Szám: \d+ regexp-el a nevertek,rtid,arfolyam, (később a befektetőid) mezőkhöz

Dátum:  $d{4}-[0-1]d-[0-3]d$  regexp-el a kibocsátás mezőhöz (Formátumát a guin jeleztem label-en)

Szöveg: ...toString().equals("") // ....toString().trim().isEmpty() itt csak azt kellett ellenőriznem, hogy van megadva szöveg és nem csupa whitespace, ez a megjegyzés és a cégnév mezőhöz kellett, ehhez nem kellett reguláris kifejezés.

Legördülő panelt nem indokolt egyik adattipusom se, de később az auto commithoz vettem fel.

### 4. feladat

### A megoldáshoz használt SQL utasítás

#### ID generáláshoz:

```
select max(trid) from tranzakcio
```

#### Beszuráshoz:

```
insert into TRANZAKCIO (RTID, BEFEKTETOID, MENNYISEG, EGYSEGAR, DATUM, TRID)
values(? ,?, ?, ?, SYSDATE,? )
```

### Magyarázat

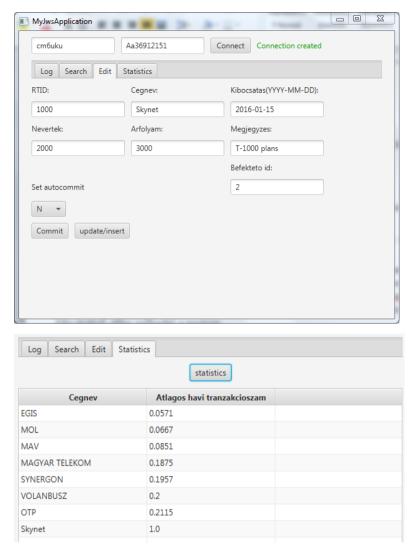
A megoldáshoz a guira fel kellett venni még egy textfieldet a befektető id nak, továbbá egy commit gombot és hozzá egy comboboxot.

A feladat megoldásához, a 2. feladatban megírt függvényeket kellett kiegészíteni, az modelben ki kellett kapcsolni az autocommitot, ha az volt megadva a legördülö fülön.

Ha meg van adva befektetőid akkor validálom, ha üresen van hagyva akkor az eredeti insert/update funkciót nyújtja a program. Ha megvan adva jó formátumban, továbbá az autocommit ki van kapcsolva és beszúrás történt akkor beszúr egy új tranzakciót is az eddigi bekért adatokkal továbbá a sysdate felhasználásával, az újonnan generált trid-vel ami az eddigi max trid+1 értéket képvisel és mennyiségnek égetve 1 el. Ha valami hiba történik akkor rollbackel a program.

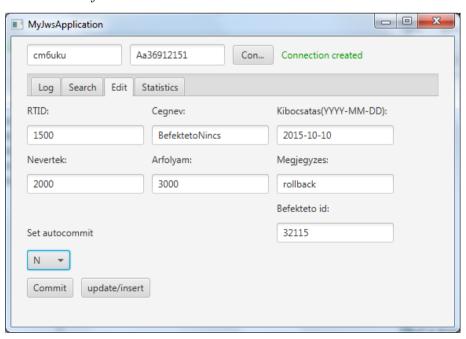
A commit gomb megnyomására commitál a program, amennyiben már előzőleg commitált, úgy nincs hatása a gombnak.

Tesztelés:



Látható hogy a tesztelés eredménye sikeres jól beszurta a mai dátummal a tranzakciót és természetessen így a havi átlag tranzakciója 1.

Teszt rossz befektetőid vel:



logba kiirta hogy rollback occured, és sql developerben megnézve nem vette fel a részvénytípus sem.

Ha megszakitjuk commit nélkül a program és kilépünk akkor elvesznek az előző commit ota beszúrt adatok/módosítások.

Ennek orvoslására a view osztályban az init fügvényben onclose requestek közé felvettem a commit végrehajtását. Így már nem tünnek el az adatok ha bezárjuk az alkalmazást, természetessen ez hirtelen áramszünet esetén/hálozati hibától nem véd meg minket.

#### 5. feladat

### A megoldáshoz használt SQL utasítás

### Magyarázat

Az előre gyártott statictis fül használható volt változtatás nélkül, a view kódjában a map kulcsait és oszlopneveit kellett csak megváltoztatni.

Kód szinten ez a viewban egy újabb event handler megvalósítását jelentette továbbá, a controller és a model getStatistics() függvényének implementálását

A feladat egy kicsit komplexebb lekérdezés megírását igényelte ami a gazdag részvényesek általi tranzakciókat figyelembe véve adott részvények átlagos havi tranzakciószámát listázta ki azokat is kilistázva akiknél ez 0.

A decodera azért volt szükség mert az eltelt hónapok töredékre pontossan vannak megadva a months\_between által, de tervezői döntésként meghoztam hogy az egy perce beszúrt tranzakciónak ne legyen átlagos havi forgalma felszorozva addig amíg l nél kisebb az eltelt hónapok száma így nem ad a frissen beszúrt tranzakció se több ezres fals havi forgalmat.

Where ben az nvl re pedig azért hogy megmaradjon a 0 db tranzakciós rt is.

ID megjelenítését is fontosnak találtam hiszen a név nem feltétlen egyedi.

# Vélemény(ek) a mérésről