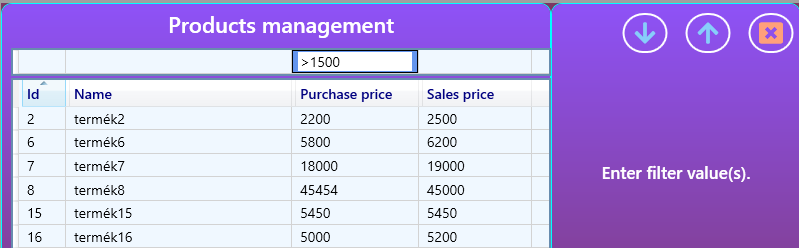
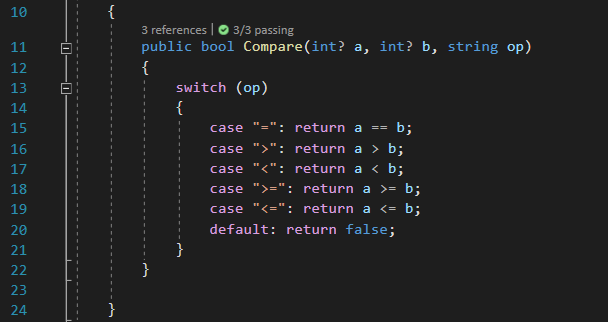
### Unit tesztek - WPF

Kirsch Ádám Péter – Sztrelcsik Zoltán

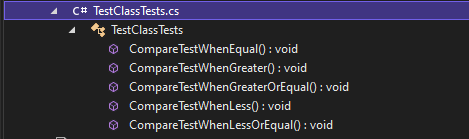
Unit tesztelés céljára **a relációs szűrést végző függvényt választottuk**. Az asztali frontend alkalmazásban bármelyik adattábla esetén lehetőség van nemcsak egyetlen, hanem egy meghatározott érték feletti, vagy alatti rekordokra is szűrést végezni. Az alábbi képen például az 1500 Ft feletti vételárú termékek láthatóak.



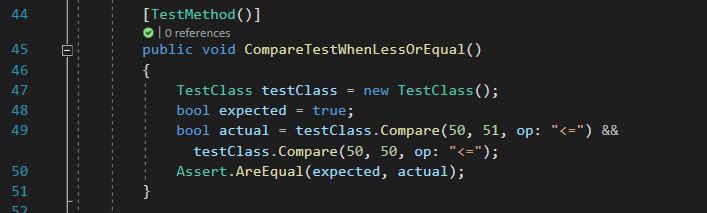
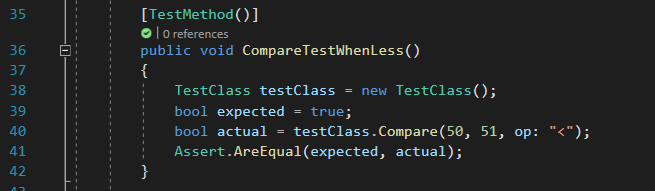
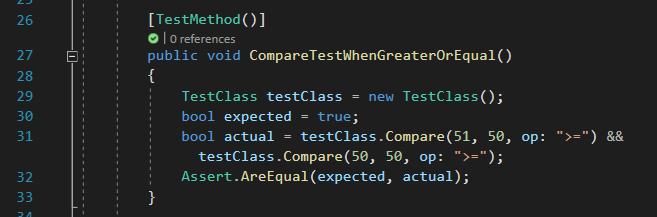
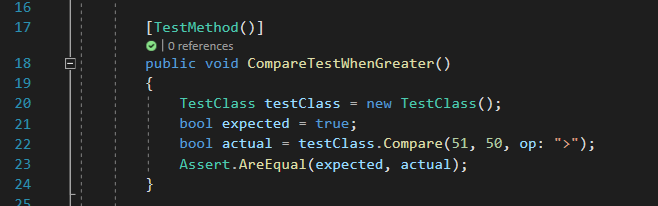
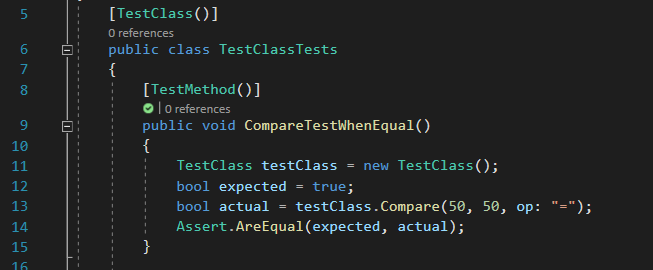
A szűrést végző függvény beazonosítja, hogy milyen relációs jel szerepel a keresőcellában, és annak függvényében ad vissza egy igaz-hamis értéket:



Mivel 5 eset szerepel a függvényben, ezért külön tesztmetódusokat hoztunk létre az "=", ">", ">=", "<" és "<=" műveleti jelekre. A tesztekhez egy projektet hoztunk létre, ebben kapott helyet a teszt osztály és annak metódusai:

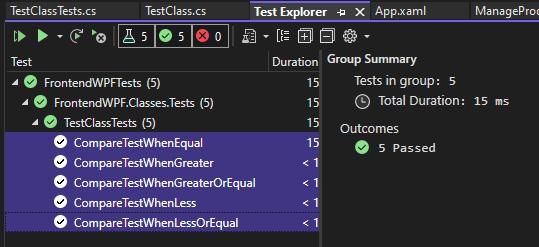


A **tesztmetódusok** a következőképpen kerültek kialakításra:

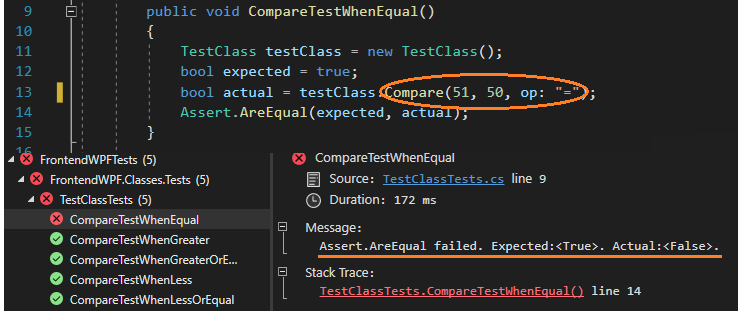


A teljesség kedvéért a ">=" és "<=" relációs jelek esetén mindkét reláció teljesülését vizsgáltuk, tehát például az utolsó tesztben szereplő "<=" relációs jelek esetén a sikeres teszt feltétele az volt, hogy egy kisebb és nagyobb szám esetén a "<" viszony, két egyenlő érték esetén pedig a "=" teljesüljön.

Az elvárt értékeket az összes teszt estében az igaz értéknek megfelelően adtuk meg, így az összes teszt sikeresnek bizonyult:

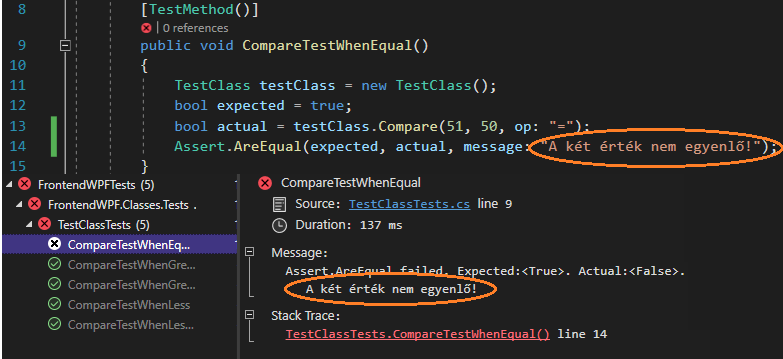


Amennyiben bármelyik esetben olyan paramétert adunk meg, amelynél hamis értéket kell adnia a tesztelt függvénynek – például két eltérő szám megadása az "=" viszony esetén, akkor a tesztnek szükségszerűen sikertelen eredményt kell adni:

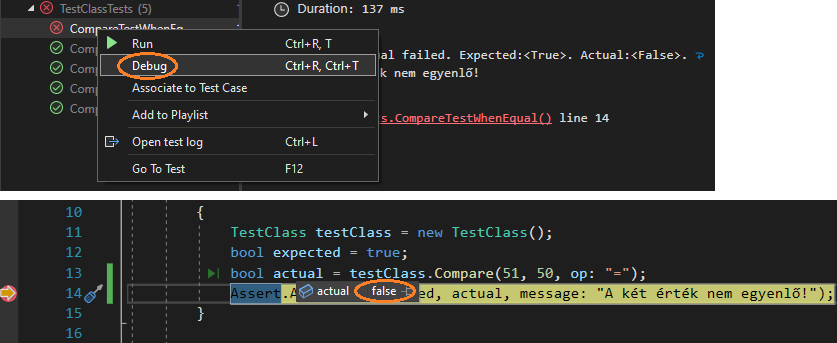


A jobb alsó sarokban lévő "Stack trace" alatt konkrétan az is megjelenik, hogy melyik teszteset melyik sorban hiúsult meg, ami a hibakeresést jelentősen megkönnyíti.

Arra is lehetőség van, hogy az egyes teszteseteknél saját egyéni hibaüzenetet adjunk meg, így a futtatáskor kiírt mezőben ez is megjelenik:



A teszteket is futtathatjuk hibakeresési módban, így azokat tetszés szerinti helyen megállíthatjuk, léptethetjük, és még a teszt meghiúsulása előtt megvizsgálhatjuk, hogy konkrétan hol és milyen feltételek mellett adódott a hamis érték:

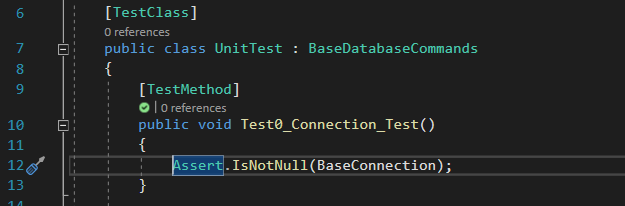


A fentiek alapján bármely más függvényre, komponensre teszteset írható.

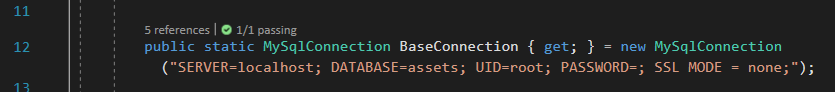
### Unit tesztek – WCF

Unit tesztelés céljára **az adatbázis műveleteket végző (CRUD) függvényeket választottuk**. Ezek bármelyik tábla vonatkozásában működnek, a tesztelés céljára a függvényeket a felhasználók (users) tábla vonatkozásában hívtuk meg.

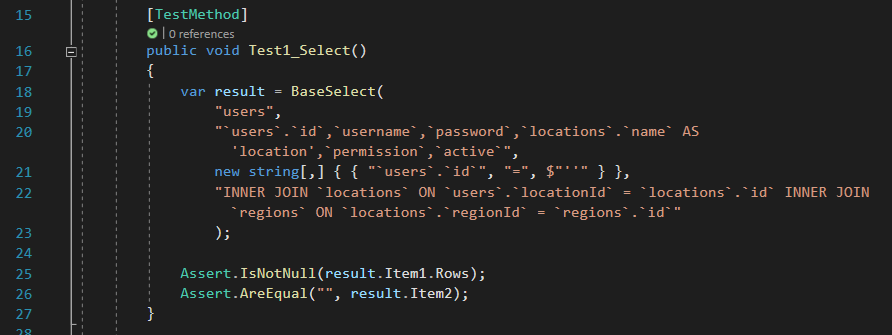
Az első unitteszt a MySQL adatbázishoz történő kapcsolódást teszteli:



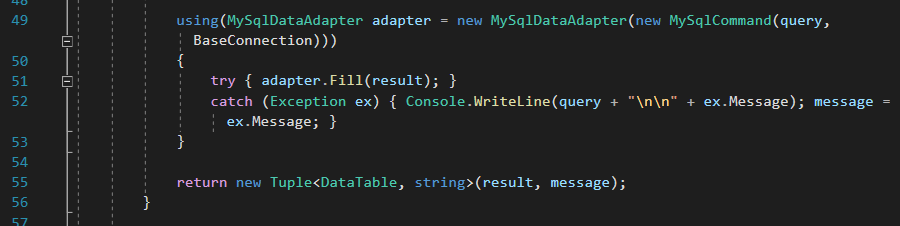
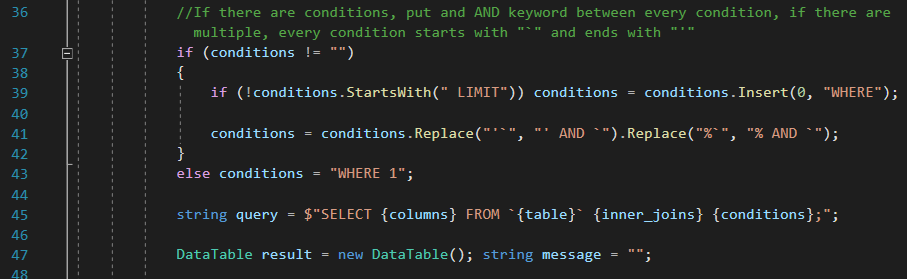
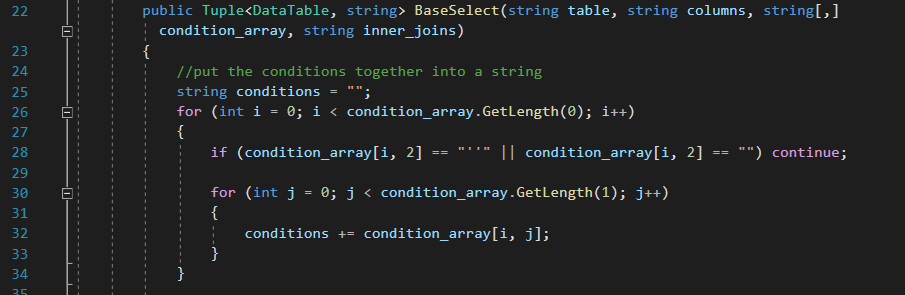
A függvény a meghívásakor megkapja a kapcsolat létrehozásához szükséges adatokat:



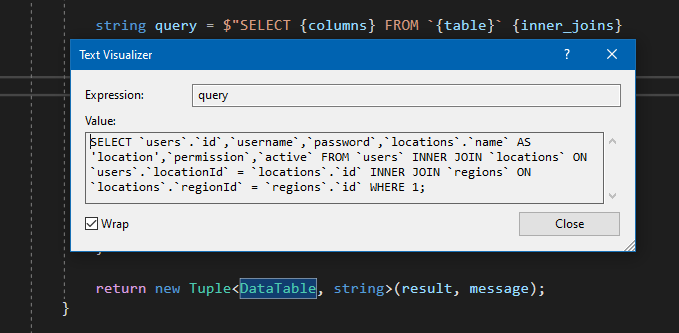
A második teszt az adatbázis-lekérés (GET) funkciót megvalósító BaseSelect() függvényt teszteli:



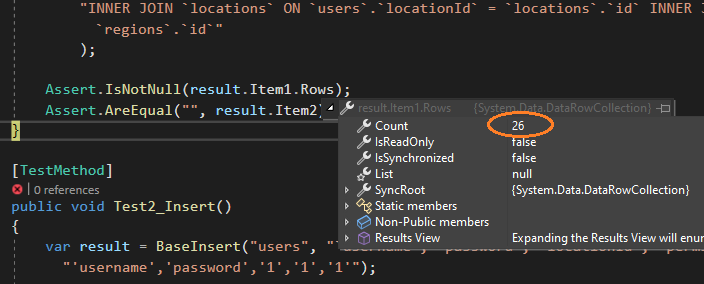
Látható, hogy a függvény megkapja a lekérdezéshez szükséges összes információt: az adattábla nevét, a mezők neveit; a tábla idegen kulcsot is tartalmaz a telephely és a régió vonatkozásában, így a lekérdezéshez megadásra kerül a többi tábla összekapcsoláshoz szükséges INNER JOIN parancs is. Erre nem lenne szükség, ha a felhasználó az idegen kulcsok értékeit adná meg a lekérdezéskor. Ez azonban nem lenne felhasználóbarát megoldás, hiszen a felhasználó könnyebben jegyzi meg, hogy a telephely neve például Miskolc, és a régió neve Borsod-Abaúj-Zemplén, és kevésbé tudja észben tartani ezek azonosító számait. Ezért a backend oldal a megadott telephely és régió neve alapján kikeresi a hozzátartozó azonosítókat a hozzácsatolt többi adattáblából, és a lekérdezéseket ezen értékek alapján végre tudja hajtani. A lekérdezett adatok visszaadásakor hasonlóképpen jár el: a felhasználó adataiként nem a táblában található azonosítószámokat adja át, hanem az azokhoz tartozó telephely és régió neveket. Amennyiben a felhasználó nem ad meg mezőneveket, akkor azokra nem történik szűrés, így ilyen esetben az adattábla teljes tartalmát fogja visszakapni. Ez a logika kiolvasható a BaseSelect() függvényből:



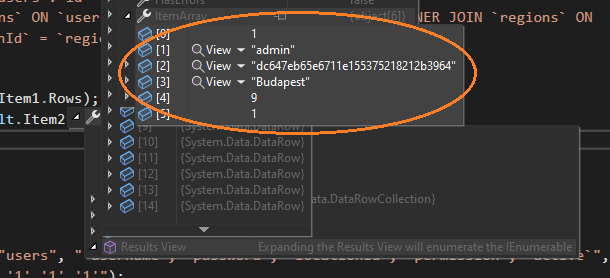
Amennyiben hibakeresési módban futtatjuk a lekérést, akkor látható, hogy a függvény milyen adatbázis lekérést hozott létre:



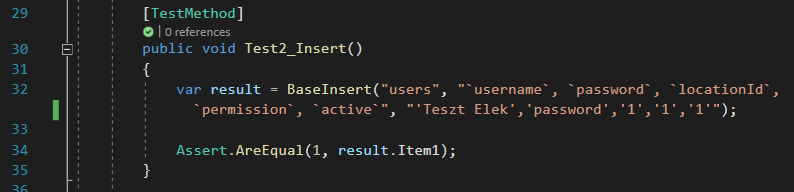
Valamint az is kiolvasható, hogy ebben a konkrét esetben 26 sor lekérése történt meg:



Igény szerint még az is megnézhető, hogy az egyes lekért sorok milyen adatokat tartalmaznak:

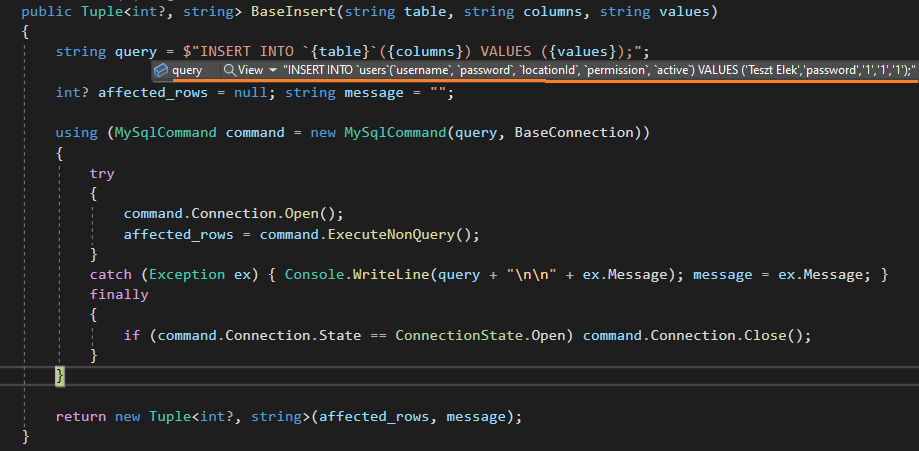


A harmadik teszt az új rekord beszúrását (POST) végző BaseInsert() függvényt ellenőrzi:

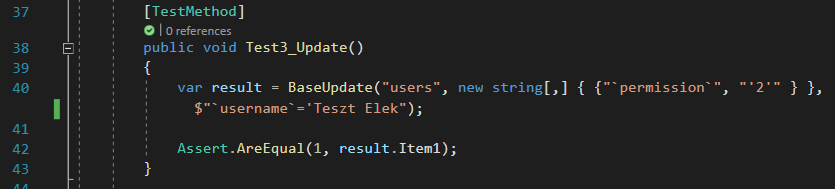


A hibakereső módban történő teszteléskor láthatjuk, hogy a függvény a fentiekben megadott bemeneti paraméterek alapján milyen INSERT parancsot állított elő:

"INSERT INTO `users`(`username`, `password`, `locationId`, `permission`, `active`) VALUES ('Teszt Elek','password','1','1','1');"

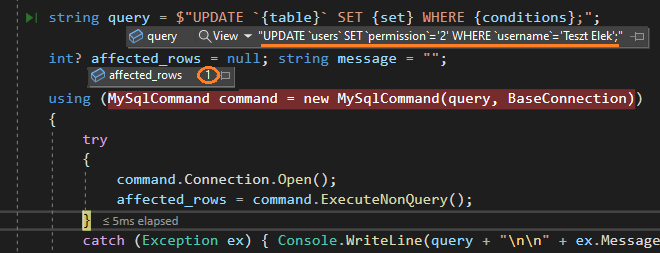


A negyedik teszt a kiválasztott rekord frissítését (PUT) végző BaseUpdate() függvényt futtatja:

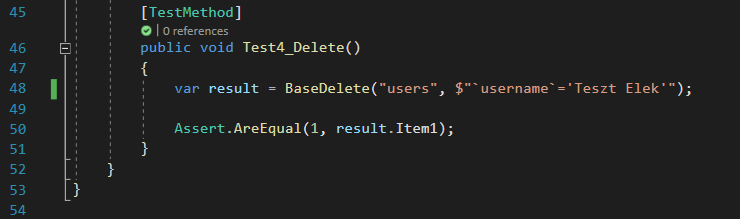


A paraméterek alapján megállapítható, hogy a "Teszt Elek" nevű felhasználó jogosultság szintjét 2-es értékre állítja. A teszt akkor sikeres, ha az adatbázis művelettel egy sor érintett, azaz ha az adatbázisban megtalálta az adott felhasználót, és a sikeres módosítás eredményeként az adatbázis lekérdezés egy érintett sor értékkel tér vissza.

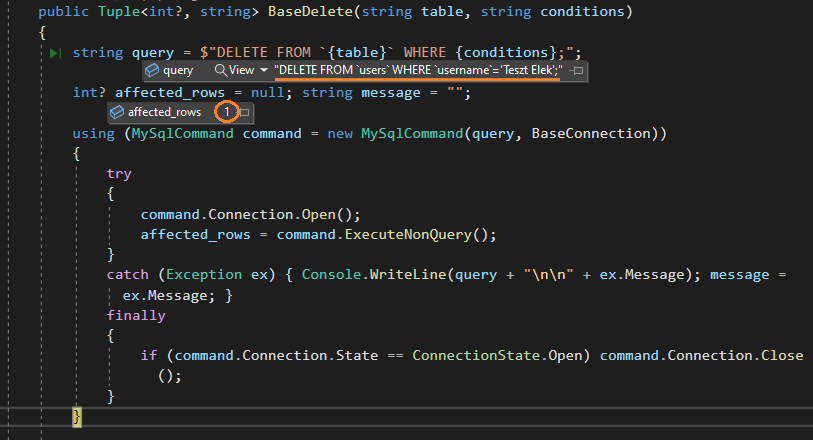
Hibakeresési módban történő futtatáskor itt is megnézhetjük, hogy az SQL lekérdezés hogyan került összeállításra, és hogy az érintett sorok száma valóban egy, vagyis hogy a teszt sikeresen fog lefutni:



Az ötödik unitteszt a törlést (DELETE) végrehajtó BaseDelete() függvény ellenőrzését végzi:



Egy breakpoint segítségével itt is megállíthatjuk a tesztet, és megnézhetjük az SQL lekérés tartalmát, és az érintett sorok számát, ami 1-es érték esetén sikeres tesztet vetít előre:



A tesztek végrehajtási sorrendje a fenti függvények esetében kiemelkedően fontos: hiszen például az adott felhasználót először fel kell venni az adatbázisba, és csak utána lehet törölni. Ezért az ABC sorrendben végrehajtásra kerülő tesztek ennek megfelelően lettek elnevezve. A tesztek kivétel nélkül sikeresnek bizonyultak:

