

**Debreceni Egyetem
Informatikai Kar**

ALKALMAZÁSFEJLESZTÉS 4GL ESZKÖZÖKKEL

Témavezető:

Márton Ágnes

Készítette:

Jagó Jácint
Programtervező Matematikus

Debrecen
2007

1	BEVEZETÉS	5
2	TÖRTÉNELMI ÁTTEKINTÉS	7
3	A 4GL-ES RENDSZEREK KOMPONENSEI.....	8
3.1	Az emberi komponens	8
3.2	Gépi komponens.....	8
3.3	A kommunikációs felület.....	9
3.3.1	Kezelői felület.....	10
3.3.2	Rendszer kezelői felület.....	10
3.3.3	Vizuális felületek	11
3.3.4	Szemantikus felület.....	11
3.3.5	Passzív objektumok	12
3.3.6	Aktív objektumok	12
3.4	Legáltalánosabb aktív kezelőobjektumok	12
3.4.1	Nyomógombok	12
3.4.2	Kapcsolók	13
3.4.3	Választógombok	13
3.4.4	Szöveges objektumok	13
3.4.5	Listák.....	13
3.4.6	Görgetősávok	14
3.4.7	Egyedi kezelőobjektumok.....	14
3.5	Felhasználó és a felületek	15
3.5.1	A fókusz	15
3.5.2	Az egérmutató	16
3.6	Kommunikációs komponens.....	16
3.7	Alkalmazáslogika	18
3.8	Eseménykezelés	18
3.9	Alkalmazás-specifikus tevékenységek.....	18
4	.NET	20
4.1	A .Net keretrendszer architektúrája	21

4.2	A .NET osztálykönyvtár (Class Library).....	22
4.3	ASP.NET.....	23
4.4	ADO.NET	23
4.5	Osztott alkalmazások készítése.....	24
5	A MEGVALÓSÍTANDÓ FELADAT LEÍRÁSA.....	26
5.1	Felhasználói elvárások, követelmények	26
5.2	Logikai tervezés.....	27
5.3	Az adatbázis ismertetése.....	28
5.3.1	A két alaptábla:	28
5.3.2	Két altábla:	28
5.3.3	A Termékleíró tábla oszlopai:.....	28
5.3.4	A Csoportleíró tábla oszlopai:	29
5.3.5	A Tulajdonság csoport tábla oszlopai:.....	31
5.3.6	A Terméktábla oszlopai:	31
5.3.7	Az Userinfo tábla oszlopai:.....	31
5.3.8	Az adatbázis rajza	31
6	FELHASZNÁLÓI LEÍRÁS	33
6.1	Rendszerkövetelmények	33
6.1.1	Hardver követelmények.....	33
6.1.2	Szoftver követelmények.....	33
6.2	Alkalmazás telepítése:	33
6.2.1	A MySql adatbázis-kezelő rendszer telepítése:	33
6.2.2	MySql Connector telepítése és konfigurálása:.....	34
7	AZ ALKALMAZÁS HASZNÁLATA.....	35
7.1	Az Adminisztrátor lehetőségei.....	35
7.2	A Felhasználó lehetőségei	35
7.3	Bejelentkezés	35
7.4	A Felhasználó lehetőségei:.....	36
7.4.1	Keresés:.....	37
7.4.2	Keresés adott tulajdonságcsoportban	39

7.4.3	Keresés adott tulajdonságcsoporthban, adott tulajdonságon belül	40
7.4.4	Keresés eredménye	41
7.4.5	Keresés eredményének riportolása	42
7.4.6	A szerkesztés:.....	43
7.5	Az adminisztrátor lehetőségei:.....	45
7.5.1	Az adminisztrátor kezdőfelülete:	45
7.5.2	Jelszóváltoztatás	46
7.5.3	Tulajdonságcsoporthok felvitele	48
7.5.4	Terméktípusok definiálása	50
7.5.5	Termékek felvitele	52
7.5.6	Keresés:.....	55
7.5.7	Keresés adott tulajdonságcsoporthban.....	57
7.5.8	Keresés adott tulajdonságcsoporthban, adott tulajdonságon belül	58
7.5.9	Keresés eredménye	59
7.5.10	Keresés eredményének riportolása	60
7.5.11	A szerkesztés:.....	61
8	PROGRAMOZÓI LEÍRÁS	63
8.1.1	A szükséges alkalmazások.....	63
8.2	Alkalmazás telepítése.....	63
8.2.1	A MySql adatbázis-kezelő rendszer telepítése	63
8.2.2	MySql Connector telepítése és konfigurálása:.....	63
8.2.3	Szkriptek futtatása.....	65
8.2.4	Az inicializáló szkriptek	66
8.2.5	A termékleíró tábla inicializáló szkriptje (termekleiro.sql):	67
9	A RENDSZER BIZTONSÁGA.....	68
9.1	Jelszavak kódolása, első bejelentkezés.....	68
9.2	Az adatbázis mentése: backup.....	69
10	ÖSSZEFOGLALÁS	72
11	TOVÁBBFEJLESZTÉSI LEHETŐSÉGEK	73
12	IRODALOMJEGYZÉK.....	74

1 Bevezetés

Napjainkban az informatikai rendszerek a mindennapi élet szerves részét alkotják. A fejlesztőknek az alkalmazásokat felgyorsult ütemben kell fejleszteniük, hogy a piac elvárásainak megfeleljenek. A 4GL-eszközök erre kiválóan alkalmasak. A .NET Framework erre kiválóan alkalmas. Sajátossága, hogy széleskörűen használható, az alkalmazásfejlesztés legtöbb területén. Céлом röviden bemutatni a 4GL és nyelvek eszközeit aztán ugyancsak röviden leírni a .NET keretrendszer architektúráját és lehetőségeit. A fő projekttem pedig egy flexibilis raktárprogram elkészítése.

Ezt az alkalmazást a debreceni National Instrumentsnél dolgozva kezdtem el tervezni és fejleszteni. Eredetileg az alkalmazás egy “Szoftver licenz nyilvántartó alkalmazás”-nak indult. Az ötletet továbbfejlesztve arra jutottam, hogy túl szegényes lenne csupán szoftverlicenzeket nyilvántartani.

Mi lenne, ha a szoftver alkalmas lenne egy akármilyen raktár nyilvántartó feladatait ellátni? Továbbá az egyszerű leltárprogramokon túlmutatva, tudjon szellemi javakat is tárolni nem csak anyagiakat. Valamint a rendszer adminisztrátora tudja saját maga meghatározni, hogy milyen tulajdonságait kívánja felvenni a terméknek. Tehát adjon szabad kezét a termék architektúrájának kialakításában. Legyen rugalmas és normalizált. Tudjon már letárolt táblákat újrafelhasználni, így csökkenteni a redundanciát. Sikert az egész szoftvert “ingyenessé” tenni a felhasznált alkalmazások ingyenessége miatt. Valamint sikerült a szoftvert nagyon egyszerűen felinstallálhatóvá tenni. Akár egyetlen sor begépelése nélkül fel lehet installálni és használni. Természetesen a termékek adatbázisbeli képének a megtervezéséhez szükséges bizonyos fokú absztrahálási képesség. Viszont mivel ez a National Instruments IT részlegének készült így fel sem merült bennem, hogy ez valamilyen problémát is okozhatna az alkalmazás adminisztrátorának. Sőt az informatikus még örömet is lelt abban, hogy végre szabad kezét kaphat a kreativitása kiéléséhez azzal, hogy ő tervezi meg a terméket.

Fontosnak tartottam, hogy a program rendelkezzen bizonyos szintű biztonságtechnikával. Ez jelszóvédelmet és adatbázis-mentést jelent ebben az alkalmazásban. A jelszavak kódolásához az MD5 kriptográfiai technológiát használok. Ennek a

technológiának a lényege, hogy a jelszóból egy hash függvénnyel egy 128 bites kódot generál. Ez a hash függvény adott karaktersorozatra mindig ugyanazt a kódot generálja. Első bejelentkezéskor a megadott jelszó hash kódját az alkalmazás berakja az **userinfo** táblába. Így a tábla lekérdezésekor nem tudódhat ki az eredeti karaktersorozat. Az n . ($n > 1$) bejelentkezéskor a megadott jelszó hash kódját összehasonlítja az első alkalommal az adatbázisba beadott hash kóddal. Ha egyezik, az alkalmazás továbblép.

A programnak tehát egyik célja az volt, hogy egy dinamikus raktárkezelőt nyújtson a felhasználóknak. Egy olyan létesítmény, ami raktározással foglalkozik elvárja, hogy sokféle árut tudjon nyilvántartani a számítógépes rendszerében. Elvárja, hogy biztonságos legyen, hogy meg tudja különböztetni a rendszer adminisztrátorát a raktárostól. Továbbá azt, hogy az adminisztrátori teendők legyenek jól meghatározva és csak a felelős hozzáértő személyzet tudjon hozzáférni az adatbázist változtató funkciókhoz.

Például, ha az aktuális szállítmány éppen egy teherautónyi televízió, akkor az adminisztrátor tudja felvenni azokat a tulajdonságokat, melyek a televíziót írják le. Ha jön egy másik szállítmánnyal bicikli, akkor új tulajdonságcsoporthoz létrehozva tárolni lehessen a biciklit leíró adatokat. Ugyanakkor, ha a cég szoftverfelelős menedzsere tudni szeretné, hogy mikor járnak le a szoftver licenszek, akkor felveszi előtte az adatbázisba, így egy helyen van előtte, hogy melyik szoftvernek kell megújítani a licenszszerződést. Legyen egy bizonyos tulajdonságcsoporthoz, ami minden árura érvényes. Például egységár, mennyiség stb. Ez természetesen egy táblát jelent három oszloppal. Ezt a táblát minden egyes termékhez hozzávesszük és így megoldjuk, hogy termékenként ne egy új tábla jöjjön létre, hanem egy új sor ebben a táblában.

Mit sikerült létrehoznom? Egy olyan alkalmazást amely megállja a helyét minden olyan helyen, ahol nyilvántartani kell. Könnyű installálni és kezelni. Szabadkezet ad a termékek letárolandó adatainak megválasztásában. Ha a nyilvántartott termékek jól vannak megtervezve, akkor a táblarendszer normalizált kevés redundanciával.

2 Történelmi áttekintés

Az 1GL azaz első generációs programozási nyelvek jelentették a gépi szintű utasításokat, amelyeket direkt a processzornak szántak mindenféle átalakítás nélkül.

2GL azaz második generációsnak nevezik azokat az assembly szerű nyelveket melyek a gépi szintű és a magasabb szintű nyelvek között vannak.

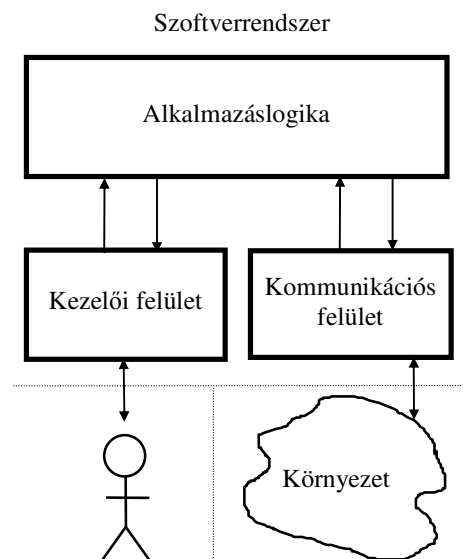
3GL-nek nevezzük a ma legszélesebb körben használt programozási nyelveket, mint a Java, C, C++ stb.

A 4GL azaz a “negyedik generációs programozási nyelv” terminussal látják el azokat a programozási környezeteket (többnyire) és nyelveket melyek speciális rendeltetésre lettek tervezve. Például kereskedelmi szoftver fejlesztés, adatbázis-kezelés, riportkészítés stb. A 3GL –t követve megpróbáltak egy magasabb absztrakciós szintet elérni. A 3GL és nyelvek lehetővé tették a többé-kevésbé természetes nyelvi megfogalmazásokat valamint az utasítások blokkokba szervezését. Ennek ellenére lassúak és hibákat is könnyű véteni a programozás során. Nyilvánvalóvá vált, hogy ezen javítani lehet magasabb szintű programozási eszközökkel és különböző metodológiákkal, amelyek lehetővé teszik a normál 3GL-es instrukciók legenerálását. Így automatizálva, a hibalehetőségek is csökkennek. A 4GL-es nyelvek nagy előnye, hogy lecsökkentik a programozói munkát, ezzel együtt egy projekt kivitelezésének költségeit. A 4GL-es nyelveket sokszor a **Domain Specific Language-ek (DSL)** csoportjába sorolják. A DSL-ek tulajdonsága az, hogy egy adott problématerületet fednek le pl. grafikai programozás, zenekészítés, adatbázis lekérdezés stb.

A 4GL kifejezést először James Martin használta az *“Applications Development Without Programmers”* című könyvében.

Első 4GL-es környezetek és nyelvek az IBM RPG-je (1960), Informatics MARK-IV (1967), MAPPER (1969) stb. A korai kezdés ellenére az első széles körben használt 4GL-es eszközök a nyolcvanas években jelentek meg és napjainkban válnak divattá.

A 4GL szoftver működését a végletekig leegyszerűsítve elmondhatjuk, hogy három fő komponensből állnak. Emberi és gépi komponens, kommunikációs komponens és alkalmazáslogika.



4GL-es rendszerek komponensei

3 A 4GL-es rendszerek komponensei

3.1 Az emberi komponens

Az emberi komponens az a rész (felület), ami a felhasználó igényeinek van kialakítva és ezen keresztül kommunikál a szoftver a felhasználóval.

3.2 Gépi komponens

A gépi komponens témakörébe tartoznak a külső rendszerek, számítógépek melyekkel a szoftverünk kommunikál a kommunikációs felülete segítségével. Az alkalmazáslogika

elnevezés takarja azokat a háttérutrutinokat, algoritmusokat melyek rendelkezésünkre állnak és amelyekkel a különböző feladatokat megoldhatjuk.

3.3 A kommunikációs felület

A kommunikációs felület megkreálása nehéz feladat. Pl. I/O műveletek. Ezt általában magas szintű nyelvek segítségével szokták megoldani, viszont ezáltal költséges és az emberi tényező miatt hibára hajlamos. Ugyancsak nehéz feladat a kezelői felület létrehozása. Ezért a jó 4GL-es rendszerek ennek kiküszöbölésére szofisztikát fejlesztői környezeteket tartalmaznak. Ide tartoznak a kezelői felület dizájnerek, amelyek segítségével a látványt lehet beállítani. Pl. gombok elhelyezkedése, méretek , színek stb.

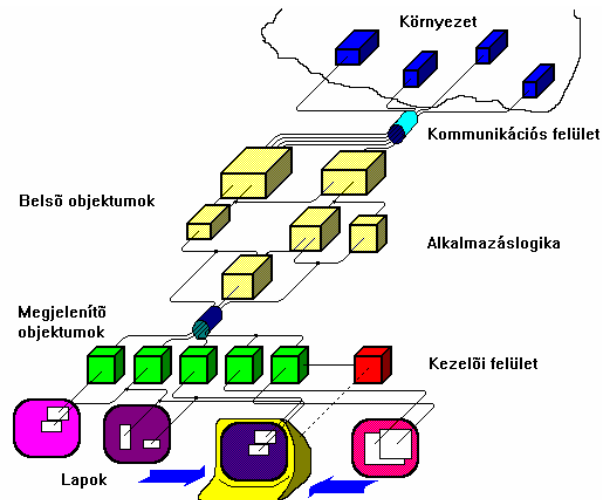
A kommunikációs felület nagyon sokféle lehet. Programunk ugyanis kapcsolatban lehet adatbázissal, vagy mérési eredményeket továbbító ipari szenzorokkal stb.

Egyes 4GL-es rendszerek speciálisan tartalmazhatják a kommunikációt végző speciális eszközöket, mások viszont nyíltak is lehetnek, azaz szabványos illesztő részeken keresztül tartják fent a kapcsolatot. (pl. ODBC, JDBC). Általában ezt egy magas szintű programozási nyelv segítségével intézi el a programozó.

Az alkalmazáslogika implementálását is magasabb szintű objektumorientált programozási nyelveken végzik. Pl. Java, C#, C++ stb. Az Oracle Forms nyelve (PL./SQL) kivétel, nem objektumorientált.

A 4GL-es program tehát a kommunikációs rétegből és a kezelői felületről kapja az információkat, ezt belső objektumai segítségével feldolgozza, majd ugyancsak a kezelői felületre juttatja a feldolgozás eredményét. Ez lehet kép, üzenet, diagram, akció, aminek a hatása látszik (többnyire) a kezelői felületen.

A következő ábrán az adatfolyam útja és az egyes komponensek vannak megrajzolva.



4GL és alkalmazások komponensei részletesebben

3.3.1 Kezelői felület

Mint feljebb leírtam, a kezelői felülettel befolyásoljuk alkalmazásunk futását, viselkedését. A felületek megtervezéséhez szükség van a programozó esztétikai és ergonómiai érzékeire is. Ezért alkalomadtán ezeket dizájnerek, grafikusok tervezik meg. A nagyon sokféle alkalmazás sokféle felületet használ, de ezeket három nagy csoportba lehet sorolni:

3.3.2 Rendszer kezelői felület

A felhasználó ilyenkor parancsokkal kommunikál az alkalmazással. A felületet is így alakították ki. Ezek a karakteres kezelői felületek, menürendszerek. Ezeknek előnye az, hogy a rendszer hatékonyabb működése (gyorsaság, robusztusság) érdekében a lehető legegyszerűbbre tervezték. A képernyőn maximum N darab karakter lehet egy sorban és maximum m sor látható. Ilyen pl. a DOS, UNIX, Oracle Forms (ez mindkettőt használja) stb. Általában nem támogatott közvetlenül 4GL-es eszközökkel a karakteres alkalmazásfejlesztés. Természetesen az alapnyelvükkel lekódolható.

3.3.3 Vizuális felületek

Ezek a grafikus felületek. Céljuk, hogy a felhasználó minél kevesebbet használja a billentyűzetet, egyszerűen, “drag and drop” technikával dolgozzon, egeret használjon. Megpróbálja modellezni a valós világot a gombokkal, görgetősávokkal, képekkel, jelzésekkel. Például egy audio CD lejátszó program. Play gomb, stop gomb, kijelző információkkal a CD tartalmáról stb. Vagy egy kasszaprogram, vagy egy számológép. Erőforrás igényes felületfajta. A képernyőt képpontokra (pixelekre) osztja, melyeknek száma meghaladhatja a milliót.

3.3.4 Szemantikus felület

A jövő zenéje. Természetes nyelvű parancsokkal kommunikálhatunk az alkalmazásunkkal. A parancsok az SQL utasításokhoz hasonlóan épülnének fel, azzal a különbséggel, hogy óriási mennyiségű parancsot kellene elemeznie a szoftvernek. Ez a számítástechnika terén egyelőre nyitott tudomány.

Tehát a karakteres felületek tervezése egy részhalma a grafikus felületeknek, a szemantikus felület meg még várat magára. Ezért elmondhatjuk, hogy a 4GL-es alkalmazásfejlesztés manapság a vizuálisság felé megy el.

A grafikus fejlesztői felület alapja a form (lap-ablak). Mindezeket egy form editorral kezeljük tervezzük dizájnoljuk. Az editorok segítségével átméretezhetjük, törölhetjük, átszínezhethetjük, mozgathatjuk, másolhatjuk, akár többszörösíthetjük formjainkat. És persze a formokon objektumainkat. Elviekben bárhány formunk lehet egy alkalmazáson belül. A formokra felkerülő objektumokat osztálykönyvtárakba sorolják. Ezek vizuálisan is meg vannak jelenítve és a programozó rendelkezésére is állnak az ún. **toolbox**-ban (**toolbar**). A programozó csak ráhúzza egér segítségével a kívánt objektumot a kívánt formra. Ezek az objektumok a megszokott módon rendelkeznek tulajdonságokkal. Az editor ezen tulajdonságok beállítását is megkönnyíti az ún. **property sheet**-ekkel. Ez egy tulajdonság-érték párokból álló lista.

A formon megjelenítendő objektumokat két csoportba sorolhatjuk:

3.3.5 Passzív objektumok

Mindig állandó állapotban vannak. Ilyenek a formon a díszítőelemek, passzívan információt hordozó elemek pl. a szoftverkészítő cég logója stb.

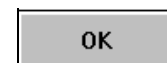
3.3.6 Aktív objektumok

Azok az objektumok melyek a valós transzformációkat végzik a környezetektől beérkező adatokon. Lényegében az alkalmazáslogika dolgozik az adatokkal és adja át a megjelentető objektumoknak. Ezek akár csak a környezet által szolgáltatott adatokat is mutathatják mindenféle átalakítás nélkül.

3.4 Legáltalánosabb aktív kezelőobjektumok

3.4.1 Nyomógombok

Olyan kezelőobjektum, mely rákattintásra kivált egy meghatározott háttérműveletet.



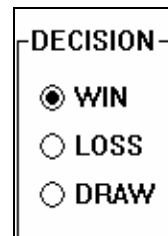
3.4.2 Kapcsolók

Két állapota van: bekapcsolt vagy kikapcsolt. Mindkét állapota egy-egy property értéket jelent, amit rögtön láthatunk a kapcsoló állapotán.



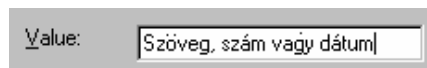
3.4.3 Választógombok

A választógombok lehetőséget nyújtanak egymást kölcsönösen kizáró választási lehetőségekre.



3.4.4 Szöveges objektumok

Adatokat jeleníthetünk meg rajta a környezetből. Ugyanakkor adatbevitelre is szokták használni legyen az karakter vagy dátumformátumban számok vagy bármi.



3.4.5 Listák

Olyan kezelőobjektumok melyek adatok listáját jeleníti meg. A listák különböző értékeit kiválaszthatjuk.

Három fajta listát szoktak használni: állandó lista (T list), beugrólista (popup list) és vegyes lista (combo box).

A listaelemeket statikusan is beállíthatjuk, de programozottan is, azaz a környezet megfelelő adatait listázhatjuk ki. Ebben az esetben természetesen dinamikusan változhat a lista.



A statikus listák-állandó listák- értékeiből választhat a felhasználó, de maga nem adhat meg más listaelemeket.

A beugrólisták akkor használatosak, ha kicsi a hely vagy nem fontos mindig megnézni a listaelemeket. Ilyenkor a felhasználó csak akkor kattint, vagy gördíti le a listát, ha szüksége van rá. Ugyancsak statikus mennyiségű listaelemet mutat.

Vegyes lista- olyan lista, melynek elemszámát a felhasználó is befolyásolhatja új értékek beadásával vagy esetleg törlésével. Pl., ha a tantárgyak listáját soroljuk fel abszolutorium összeállításánál, akkor vannak kötelező tárgyak a listában, ami alapból benne van, viszont a szabadon választható tárgyak mindenkinél különböznek ezért mindenki magának állítja be.

3.4.6 Görgetősávok

Akkor használatosak, amikor valamilyen objektum (sorozat) nem fért bele a formba, vagy kevés a hely és arrébb kell görgetni a képet. Vízszintesen vagy függőlegesen.



3.4.7 Egyedi kezelőobjektumok

A 4GL-es rendszerek nem feltétlenül kell jelentsenek általános célú kezelőszerveket. Nagyon sok kezelőobjektum arra lett kitalálva, hogy speciális elvárásoknak tegyen eleget. Ilyenek általában az iparban használatos szoftverek kezelőobjektumai. Például különböző analóg

húzóák, speciális kapcsolók, oszcilloszkóp szerű az adatáramlást függvényként kirajzoló kijelzők, tartály-ikon, reaktorok, ventilátorok, tekerentyűk.

A 4GL rendszerek általában lehetőséget biztosítanak a beépítetteken kívüli saját kezelőszervek létrehozására. Minden beépített kezelőszerv a háttérben egy objektum. A rendszerek természetesen definiálnak egy objektumok közötti üzenetváltást, és az objektumok viselkedését bizonyos helyzetekben. Az egyedi kezelőszervek megvalósítása ezek után szabványos felületű objektumosztályokon keresztül lehetséges, melyeket megírunk, megfelelő módon lefordítva külső állományokban tárolunk, és a 4GL alkalmazás ezek alapján a szabványos felületen keresztül képes lesz majd egyedi kezelőszerv példányokat generálni és használni.

3.5 Felhasználó és a felületek

Bevitelre és kezelésre a felhasználó a két szabványos eszközt a klaviatúrát és az egeret használja. Amikor e két eszköz egyikével a felhasználó elkezd beavatkozni a program futásába, akkor eseményeket vált ki, amit az alkalmazáslogika kezel le. A kérdés az, hogy a felület melyik objektumára vonatkozik a kezelés. Ugyanis az egérrel egyszerre csak egy objektumot tudunk kezelni, ugyanígy a klaviatúrával is. Ezt a fókusz és az egérmutató bevezetésével oldották meg a 4GL-es rendszerek.

3.5.1 A fókusz

A fókusz határozza meg, hogy a billentyűzet billentyűinek a leütése, mely objektumra vonatkozzanak. Külalakra a fókusz általában egy villogó pálcika, ami függőleges vagy vízszintes, vagy ha gombokra vonatkozik, akkor általában egy vékony keret.

3.5.2 Az egérmutató

Mutatja, hogy ha megnyomjuk az egér valamelyik gombját akkor melyik objektum fog reagálni rá a felületen.

A fókusz és az egérmutatót természetesen egyik objektumról a másikra ugráltathatjuk. Az egérmutatót egyértelműen tudjuk mozgatni az egérrel. A fókusz mozgásának a beállítása már nehezebb. A fókusz mozgását hívjuk navigációnak. A navigációt klaviatúrával, de egérmutatóval is befolyásolhatjuk. Klaviatúrával speciális billentyűzetekkel vagy billentyűzetkombinációkkal tudjuk mozgatni. Egérrel az, hogy melyik objektumot érem el előbb, teljesen a felhasználóra van bízva, viszont ugyanez billentyűzettel nehezebb. Erre van kitalálva a navigálási sorrend, amikor is **tab** billentyűvel tudunk a sorba lefelé navigálni vagy pl. **shift-tab**-al felfelé.

3.6 Kommunikációs komponens

A kommunikációs komponensen keresztül tartja a rendszerünk a kapcsolatot a gépi környezetével. Ez a komponens biztosítja az információáramlást és a szinkronizációt a külső alkalmazások és számítógépek között. Ezek a külső alkalmazások olyan szolgáltatásokat nyújtanak melyek nem a mi szoftverünk dolga megtenni. Pl.:

- Állományok kezelése, hálózati kapcsolat bármi, ami az operációs rendszer feladata.
- Adatbázis hozzáférés, lekérdezés módosítás stb.
- A már a számítógépre felinstallált speciális alkalmazások melyek segítségével szeretnénk pl. adatainkat rendszerezni pl. táblázatkezelők, böngészők stb.
- Méréseket szeretnénk készíteni, mérések eredményeit megjeleníteni, kirajzolni, tárolni stb.

Komplex ipari rendszert szeretnénk alkalmazásunkkal kezelni, irányítani, vagy csak felügyelni.

Természetesen az alkalmazásunkat úgy kell összehozni, hogy ne implementálja ezen külső kapcsolatok által alkalmazott szolgáltatásokat. Azok maradjanak külső szinten. Túl sok hasznos erőforrást és emberi munkát igényelne.

Alkalmazásunk viselkedhet egy kapcsolaton belül aktívan és passzívan. Passzívan az alkalmazásunk nem válaszol eseményekkel a külső alkalmazás reakcióira. Adatokat beolvas tőle, de nem reagál rá semmivel. Aktív esetben viszont a külső alkalmazás eseményeket válthat ki az alkalmazásunkból.

Lényeges a kommunikációs csatorna, mely lehet hálózati kommunikációs felület, vagy egygépes objektumorientált felület.

Az operációs rendszerek alkalmazásait rendszerhívásokon keresztül éri el a 4GL-es program. Természetesen az alkalmazáslogika a háttérben olyan objektumokat generál melyek metódusai rendszerhívásokat generálnak.

Az adatbázis-kezelőkkel való kommunikáció már sokkal bonyolultabb és szerteágzóbb. Először meg kell jelölni azt az adatbázist, amelyikkel szeretnénk kommunikálni. Ezzel kell kiépítenünk a kapcsolatot. Ha nem a gépünkön van, akkor hálózati kapcsolattal kell számolnunk. Az adatok feldolgozása után ezt a kapcsolatot le kell bontani. SQL DML parancsokat küldhet az alkalmazásunk az adatbázisnak, majd megjeleníti a szelektált sorokat vagy hibaüzeneteket.

Ha a kezelő egy adatelemet megváltoztat, töröl, vagy éppen új sort visz be, a módosításokat előbb utóbb be kell vinni az adatbázisba is. Mindez DML utasítások automatikus megfogalmazását és végrehajtását teszi szükségessé. Mivel a DML utasításokból már tranzakciók keletkeznek, a 4GL-eszköz feladata a tranzakciók lezárása, esetleges visszagörgetése, mentési pontok elhelyezése, egyszóval maga a tranzakció-kezelés is.

Más külső alkalmazás esetén több különböző kommunikációs szabvány létezik. Windows-os rendszeren az OLE2 vagy OCX pl. Ilyen esetekben a 4GL-es nyelv biztosítja számunkra a megfelelő objektumok létrehozásának lehetőségét amelyeken keresztül képesek vagyunk a külső alkalmazás kezelésére. Pl. Word vagy Excelben képesek leszünk pl. szerkesztés, módosítás végrehajtására.

Eddig csak tipikusan olyan kapcsolatokról esett szó, amikor a külső alkalmazás passzív félként vesz részt. Azonban vannak különböző ipari rendszerek melyek szolgáltatott adatai eseményt kell kiváltanak a 4GL-es alkalmazásból. Például a mérési adatok feldolgozását alkalmazásunk végzi és bizonyos esetekben reagálnia is kell. Az ilyen környezetben működő eszközök mindig párhuzamos működésűek, és bennük egy (vagy több) külön folyamat gondoskodik a kommunikációs felületen megjelenő mérési adatok fogadásáról, és a feldolgozó és megjelenítő folyamatok számára való továbbításáról.

3.7 Alkalmazáslogika

Az alkalmazáslogika valósítja meg az alkalmazásunk procedurális belső működéseit. A végrehajtandó tevékenységeket két általános csoportba sorolhatjuk:

3.8 Eseménykezelés

Ezek a kezelői felületről és a kommunikációs felületről érkező események kiszolgálásával kapcsolatos tevékenységek. A felületen megjelenő előredefiniált elemek száma bármely 4GL környezetben adott, és így definiálható azon események köre, melyek ezekhez a felületi elemekhez kapcsolódnak.

3.9 Alkalmazás-specifikus tevékenységek

Olyan egyedi, általunk megírt kódrészletek ezek, melyek az alkalmazás igényeinknek megfelelő működését eredményezik. A kétfajta tevékenység természetesen szorosan összefügg. A 4GL-eszközök az általuk ismert események körét előre definiálják, és kezelésükre egy kész alkalmazásvázlat kínálnak. Az események kezeléséhez a 4GL-es rendszer számos objektum számos metódusa tartozik, ami majd az esemény bekövetkeztekor lefut. Ennek következményeként egyszerűbb, tipikus feladatokat megoldó alkalmazások akár egyetlen programsor megírása nélkül, pusztán a

megfelelő objektumok létrehozásával, és a hozzájuk kapcsolódó beépített tevékenységek kihasználásával megvalósíthatók. Saját kódot olyan esetekben helyezünk el a “vázban”, amikor a feladatunk specialitása miatt szükségünk van rá. Ez nem hagyományos módon történik, hogy egybefüggő kódrészletet írunk, hanem meg kell keresni azokat az **entry point**-okat ahova kódrészletünket be tudjuk illeszteni. Ezek különböző háttérobjektumok metódusai, eseménykezelők stb. Ez nem egy-egyértelmű feladat. Az egyedi tevékenységek megvalósítása során a 4GL-eszközök lehetővé teszik a programozó számára, hogy a rendszer beépített működéseit kiegészítse vagy egyszerűen felüldefiniálja. Az utóbbi esetben a 4GL-eszköz által nyújtott alapviselkedés teljes egészében felüldefiniálható, és igényeinkhez igazítható.

4 .NET

Mi a .Net? A legegyszerűbb válasz erre a kérdésre az, hogy egy keretrendszer, amiben Windows alkalmazások készíthetők és futtathatóak. Miből áll a keretrendszer? A .NET keretrendszer elemei négy fő csoportba sorolhatók:

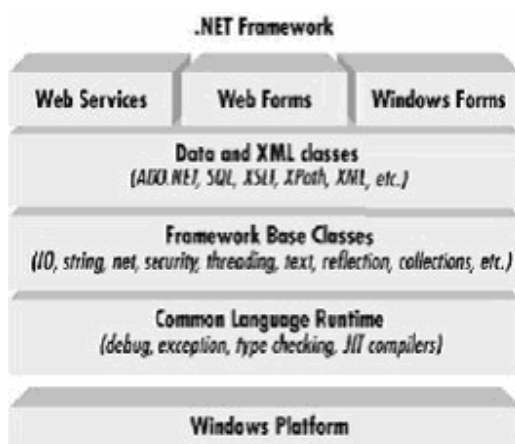
- Programozási nyelvek (bővülő) halmaza pl. C#, J#, Visual Basic.NET stb. , fejlesztőkörnyezetek halmaza: Pl. Visual Studio.NET, ami több, mint 5000 osztályból álló könyvtárrendszert foglal magába, valamint a CLR (common language runtime), mely a háttérfolyamatokért felelős.
- .Net Enterprise szerverek halmaza pl. : SQL Server 2000, Exchange 2000, BizTalk 2000, email stb.
- Kereskedelmi webszolgáltatások halmaza névszerint „Project Hailstorm”, mely nem ingyenes, de nagyon hasznos webszolgáltatások vásárolhatók s illeszthetők be saját szoftverbe.
- Új .Net képes nem PC-s eszközök pl. mobiltelefonok, game box-ok stb.

A nyelvek között áthivatkozásokra is van lehetőség egy programon belül, például egyik nyelvből lehet örököltetni más nyelv osztályából. Ezt a **Common Type System** és a **Common Language Specification** teszi lehetővé. A közös típus platformot (CTS) a .NET úgy oldja meg, hogy a .NET-ben minden egy bizonyos osztály objektuma. A CLS pedig közös nyelvi szabályokat határoz meg. Minden .NET-es nyelvnek ennek megfelelőnek kell lennie. A .Net programokat először egy köztes nyelvre fordítja le a rendszer (MSIL), amit később futtat a CLR. Különböző nyelvekből fordított köztes nyelvű fájlok identikusak.

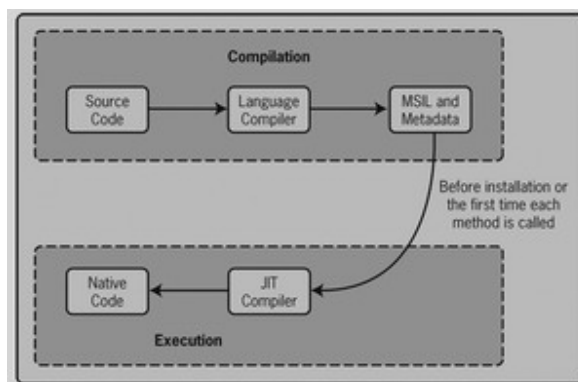
4.1 A .Net keretrendszer architektúrája

Architektúrális szempontból a .Net keretrendszer három fő komponensből áll:

- Négy hivatalos nyelv: C#, VB.NET, Managed C++, Jscript.NET
- Common Language Runtime (CLR), ami a fenti nyelvek által használt objektum orientált platform.
- Framework Class Library (FCL) vagyis egy hatalmas osztálykönyvtár.



A legfontosabb komponens ebben a CLR, mely egy működési környezetet teremt a programnak. Nagyon hasonlít a Java Virtual Machine-hoz. Debuggol, kivételeket kezel, típusellenőrzést végez, Just In Time Compilert tartalmaz.



4.2 A .NET osztálykönyvtár (Class Library)

Mint ahogy a név sugallja ez nem más, mint egy osztálykönyvtár, ami megkönnyíti a programozók életét. Ez a könyvtár egy faszerkezetet alkot. A lenti ábrán egy apró részlete van ennek a fának.

System: A gyökere a fának, ez tartalmazza az összes névteret. Megtalálhatóak benne továbbá bizonyos alaptípusok, amiket a CLR használ.

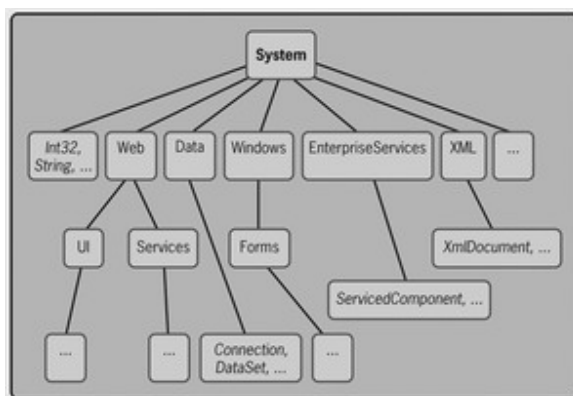
System.Web: Ez a névtér webes alkalmazások készítéséhez szükséges osztályokat tartalmazza.

System.Data: Ez a névtér határozza meg az ADO.NET-et. Pl.: a Connection osztály arra való, hogy kapcsolatot teremtsen egy adatbázis-kezelő rendszerrel. Egy példánya a DataSet-nek pedig letárolja egy lekérdezés eredményét.

System.Windows.Forms: Az ebben a névtérben evő típusok segítségével rakunk össze Windows-os Formokat, amik felhasználói felületek.

System.EnterpriseServices: Ezen névtér osztályai szolgáltatásokat nyújtanak bizonyos vállalati szoftverfelesztéhez.

System.XML: Ezen névtér típusai elősegítik az XML-es adatfeldolgozást.

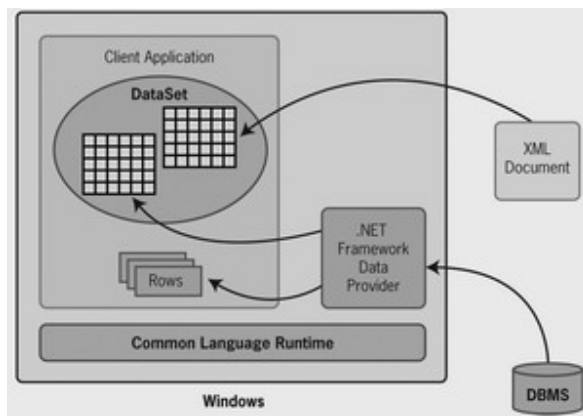


4.3 ASP.NET

Webes alkalmazásfejlesztésre való. Active Server Pages-t rövidít. A System.Web névtérbe van implementálva. Egy ASPs alkalmazás egy vagy több lapból áll, HTML és/vagy végrehajtható kódot tartalmaz, és aspx kiterjesztése van. Megkönnyíti a felhasználói felület készítését. Webes eszközöket lehet „drag and drop” technikával ráhúzni a felületre. Tartalmaz eszközöket egy alkalmazás állapot-információinak meghatározására. Beépített eszközöket tartalmaz egy alkalmazás felhasználóinak a menedzselésére. Támogatja a **data binding**-et, ami megkönnyíti a kapcsolódást egy relációs adatbázis-kezelő rendszerhez. A legtöbb .Net-es fejlesztő az ASP területén dolgozik.

4.4 ADO.NET

Ez a csoport teszi lehetővé, hogy letárolt adatokkal dolgozzunk. Ahogy a lenti ábra mutatja, az adatbázis-hozzáférés a **Data Provider**-en (adatszolgáltató) alapul, ami managed kódként van megírva. Ez azt jelenti, hogy a CLI generálja és nem a programozó.



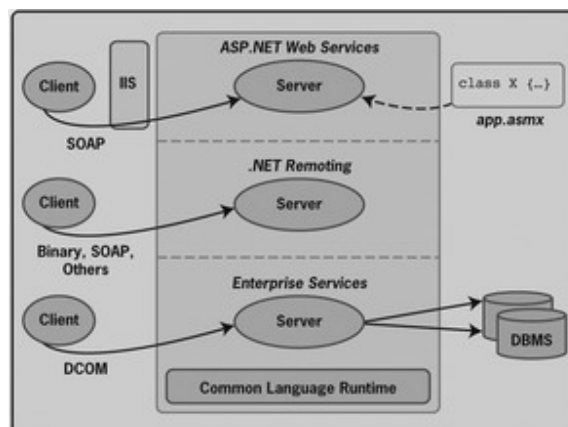
Ezekon a szolgáltatókon keresztül csatlakozik SQL Szerverre vagy más relációs adatbázis-kezelő rendszerre, ami a keretrendszerbe beletartozik. Ennek lényege az, hogy kliens alkalmazások parancsokat küldenek az adatbázis-kezelőnek azok adatokkal válaszolnak, és a kliens alkalmazás feldolgozza az adatot. Például egy SQL lekérdezés eredménye kétféleképpen

dolgozható fel. Olyan alkalmazások melyeknek elegendő egyszerre egy sor, mint lekérdezés eredmény azok használják a **DataReader** osztályt. Azok melyek komplexebb tevékenységet végeznek, pl. egy böngészőben frissítenek adatokat, azok inkább használják a Data Set objektumot, mely beemeli a lekérdezés eredményét és a metaadatokat egy cache-be, ahol könnyen el lehet érni.

A fenti ábra azt mutatja, hogy a Data Set-ek egyszerre több táblát is tartalmazhatnak. Minden tábla külön eredménye lehet egy-egy lekérdezésnek, tehát egy Data Set több lekérdezés eredményét is tartalmazhatja. Akár külön adatbázisokból is. Az is lehetséges, hogy XML dokumentumot olvassunk be direkt egy DataSet-be anélkül, hogy az adatszolgáltatót használtuk volna. Pontosan azért mert az XML technológia az utóbbi időben nagyon fontossá vált.

4.5 Osztott alkalmazások készítése

Nagyon fontos része a technológiának, hisz manapság már alig van olyan alkalmazás, ami más alkalmazással ne kommunikálna. A problémát az okozza, hogy különböző alkalmazások különböző képpen kommunikálnak. A .Net keretrendszer három technológiát különböztet meg osztott alkalmazások készítésére. A következő ábrán a 3 technológia látható.



- ASP.Net Web szolgáltatások SOAP “Simple Object Access Protocol” szabvánnyal kommunikálnak. Ezek olyan alkalmazások melyek Web szolgáltatásokként tesznek közzé metódusokat. Asmx kiterjesztésű fájlokból épülnek fel. A kliens program kéréseket közöl használva a standard Web szolgáltatás protokollt (SOAP) és a megfelelő lap betöltődik. A kérések és válaszok a standard Windows Web Serveren keresztül (IIS) történnek. Ez a megoldás inkább különböző platformokon létrehozott alkalmazásokra jellemző.
- Kommunikáció .Net keretrendszer alapú alkalmazásokkal. Mint a fenti ábra mutatja itt egyszerű kliens-szerver szerepet vesz fel az alkalmazás és tetszés szerinti módszert használva kommunikál. Az ebben a keretrendszerben készült alkalmazások az MSIL miatt elég hordozhatóak. MSIL Microsoft Intermediate Language olyan, mint a Java byte kódja. Viszont sok része a osztály könyvtárnak (Class Library) nem támogatott más nem Windows-os rendszeren. Pl. ilyen az Enterprise Services, mely segítségével tranzakció orientált alkalmazásokat építhetünk.
- Enterprise Services támogatja az elosztott tranzakciókat. Ahogy az ábra 3. sora mutatja, egy szerver alkalmazás kettő darab adatbázishoz kapcsolódik. Egy alkalmazás tartalmazhat tranzakciókat melyek kezelik mindkét adatbázis adatait. Távoli kliensek DCOM (Distributed COM)-on keresztül kezelhetik az adatbázisokat. Az Enterprise Services lehetőséget nyújt ASP.Net-es alkalmazásnak is az adatelérésre és kezelésre.

5 A megvalósítandó feladat leírása

A diplomamunkám témájául választott alkalmazás egy raktár nyilvántartási és keresési feladatait igyekszik megvalósítani. A lényeg abban rejlik, hogy ez a raktárnyilvántartó rendszer nagyon elasztikus legyen, és mindent nyilván tudjon tartani. Teszi ezt úgy hogy a felhasznált szoftverek teljesen ingyen letölthetők.

5.1 Felhasználói elvárások, követelmények

- Bejelentkezéskor autentikáció
- A program tegye lehetővé adatok, termékek, árucikkek tárolását
- A program tegye lehetővé az árubejegyzések szerkezetének a felhasználó általi összeállítását, így növelve a rugalmasságot.
- A program nyújtson valamilyen egyszerű riportgenerálási lehetőséget
- Legyen megoldott az adatmentés, ki lehessen menteni az adatbázis tartalmát

A program abból a célból készült, hogy egy dinamikus raktárkezelőt nyújtson a felhasználóknak. Egy olyan létesítmény, ami raktározással foglalkozik elvárja, hogy sokféle árut tudjon nyilvántartani a számítógépes rendszerében. Elvárja, hogy biztonságos legyen, hogy meg tudja különböztetni a rendszer adminisztrátorát a raktárostól. Továbbá azt, hogy az adminisztrátori teendők legyenek jól meghatározva és csak a felelős hozzáértő személyzet tudjon hozzáférni az adatbázist változtató funkciókhoz. Kiemelten fontos az, hogy a rendszer bármilyen árufajtát vagy szellemi javakat rendszerezni, tárolni tudjon. Például, ha az aktuális szállítmány éppen egy teherautónyi televízió, akkor az adminisztrátor tudja felvenni azokat a tulajdonságokat, melyek a televíziót írják le. Ha jön egy másik szállítmánnyal bicikli, akkor új tulajdonságcsoporthoz létrehozva tárolni lehessen a biciklit leíró adatokat. Ugyanakkor, ha a cég szoftverfelelős menedzsere tudni szeretné, hogy mikor járnak le a szoftver licenszek, akkor felveszi előtte az adatbázisba, így egy helyen van előtte, hogy melyik szoftvernek kell

megújítani a licenz szerződését. Legyen egy bizonyos tulajdonságcsoporthoz, ami minden árura érvényes. Például egységár, mennyiség.

5.2 Logikai tervezés

A programot két felhasználó tudja használni: **adminisztrátor** és **felhasználó**. Az **adminisztrátor** tud új tulajdonságcsoporthoz létrehozni. Ezáltal új tulajdonságokat tud csoportokba szervezni. Tud új termék típust létrehozni, és persze ezáltal az új terméktípushoz tud hozzáadni tulajdonságcsoporthoz. Szám szerint maximum 10-et. Ezután jön értelemszerűen az új termék felvitele. A termék adott típusú termék lesz és így előre meg van határozva milyen tulajdonságcsoporthoz fogják jellemezni és persze ezen belül milyen tulajdonságok. Konkrét termék megadásánál az **adminisztrátor** konkretizálni tudja a termék adatait. Pl. megadja a raktárszámot, darabszámot stb. Alaptevékenysége az **adminisztrátor**-nak természetesen a keresés. Kiválasztja a termék típusát kiválasztja a kívánt tulajdonságcsoporthoz, azon belül kiválasztja a kívánt tulajdonságot és megadja a keresett szöveget. Természetesen kereshet az összes tulajdonság között adott szöveget, viszont mennyivel előnyösebb ha adott tulajdonság között tudunk kutakodni. Teszem azt jön egy ügyfél a raktárba és repülőgépet szeretne venni, aminek a motorteljesítménye „3000”, akkor a raktáros motorteljesítményre keressen rá. És nem egyszerűen beírni hogy „3000” a keresőbe, mert akkor kiadhat rengeteg repülőt, aminek pl. a hatótávolsága „3000” vagy más tulajdonsága. Az adminisztrátornak legyen joga módosítani a termékek adatait egyénileg, vagy akár módosítsa az összes ezzel a tulajdonsággal rendelkező termék ezen tulajdonságát. Pl. ha van egy vagy több termék ugyanattól a beszállítótól. Ha annak megváltozik a címe vagy telefonszáma, akkor jó hogy nem kell minden termékénél megváltoztatni külön-külön. Továbbá az **adminisztrátor** tud törölni terméket. Ez például jelentheti a raktárból elszállítást. A **felhasználó** sokkal kevesebb lehetőséggel rendelkezik. Keres, és módosít olyan terméktulajdonságokat melyeknek az **adminisztrátor** a **public** tulajdonságot adta.

5.3 Az adatbázis ismertetése

Az adatbázis magját két tábla képezi. Az adatbázis viszont nagyon gyorsan bővül. Az első tulajdonságcsoporthoz bevitelénél és az első termék bevitelénél már 4 tábla van. Termékenként és tulajdonságcsoporthoz egy-egy új tábla jön létre.

5.3.1 A két alaptábla:

- Termékleiro
- Csoportleiro

5.3.2 Két altábla:

- Terméktábla
- Tulajdonságcsoporthoz tábla

5.3.3 A Termékleíró tábla oszlopai:

- ``TermekId` int(11) NOT NULL auto_increment`: A termék egyedi azonosítója.
- ``Termeknev` char(64) default NULL`, : A termék neve
- ``TermekIDX` char(4) default NULL`, Ez azonosítja a termékhez tartozó terméktáblát
- ``TulajdonsagCsoportSzam` smallint(6) default NULL`, Hány tulajdonság csoportja van egy terméknek
- ``Csoport01` int(11) default '0'`,
- ``Csoport02` int(11) default '0'`,
- ``Csoport03` int(11) default '0'`,
- ``Csoport04` int(11) default '0'`,
- ``Csoport05` int(11) default '0'`,
- ``Csoport06` int(11) default '0'`,
- ``Csoport07` int(11) default '0'`,
- ``Csoport08` int(11) default '0'`,

- ``Csoport09` int(11) default '0',`
- ``Csoport10` int(11) default '0',`
- ``CsoportNev01` char(32) default NULL,`
- ``CsoportNev02` char(32) default NULL,`
- ``CsoportNev03` char(32) default NULL,`
- ``CsoportNev04` char(32) default NULL,`
- ``CsoportNev05` char(32) default NULL,`
- ``CsoportNev06` char(32) default NULL,`
- ``CsoportNev07` char(32) default NULL,`
- ``CsoportNev08` char(32) default NULL,`
- ``CsoportNev09` char(32) default NULL,`
- ``CsoportNev10` char(32) default NULL,`

Egy terméknek 10 tulajdonságcsoportha van és ennek megfelelően 10 csoportneve. A tábla elsődleges kulcsa **TermekId**.

5.3.4 A Csoportleíró tábla oszlopai:

- ``CsoportLeiroID` int(11) NOT NULL auto_increment, :` Tulajdonságcsoportha azonosítója.
- ``CsoportNev` char(32) default "", :` Tulajdonságcsoportha neve
- ``TablaIDX` char(4) default '0000', :` Tulajdonságcsoportha értéktáblájára mutató mező.
- ``MezoSzam` smallint(6) default NULL, :` Ahány tulajdonság lesz
- ``MezoNev01` char(32) default "",`
- ``MezoNev02` char(32) default "",`
- ``MezoNev03` char(32) default "",`
- ``MezoNev04` char(32) default "",`
- ``MezoNev05` char(32) default "",`

- ``MezoNev06` char(32) default "`,
- ``MezoNev07` char(32) default "`,
- ``MezoNev08` char(32) default "`,
- ``MezoNev09` char(32) default "`,
- ``MezoNev10` char(32) default "`,
- ``MezoTulajdonsag01` char(8) default "`,
- ``MezoTulajdonsag02` char(8) default "`,
- ``MezoTulajdonsag03` char(8) default "`,
- ``MezoTulajdonsag04` char(8) default "`,
- ``MezoTulajdonsag05` char(8) default "`,
- ``MezoTulajdonsag06` char(8) default "`,
- ``MezoTulajdonsag07` char(8) default "`,
- ``MezoTulajdonsag08` char(8) default "`,
- ``MezoTulajdonsag09` char(8) default "`,
- ``MezoTulajdonsag10` char(8) default "`,
- ``MezoTipus01` char(18) default "`,
- ``MezoTipus02` char(18) default "`,
- ``MezoTipus03` char(18) default "`,
- ``MezoTipus04` char(18) default "`,
- ``MezoTipus05` char(18) default "`,
- ``MezoTipus06` char(18) default "`,
- ``MezoTipus07` char(18) default "`,
- ``MezoTipus08` char(18) default "`,
- ``MezoTipus09` char(18) default "`,
- ``MezoTipus10` char(18) default "`,

Értelemszerűen **Mezőnév** lesz a tulajdonság neve, **Mezőtulajdonság** állítja be a láthatóságot, **Mezőtípus** lesz a tulajdonság típusa. **CsoportleiroID** kulcs. **TablaIDX** külső kulcs.

5.3.5 A Tulajdonság csoport tábla oszlopai:

- ``Idx``: int(11) NOT NULL auto_increment : elsődleges kulcs.
- ``tul0`` : „mezőtípus 0” típusú tulajdonság.
- ``tul1`` : „mezőtípus 1” típusú tulajdonság.
- ``tul2`` : „mezőtípus 2” típusú tulajdonság.
- ...

Ebbe a táblába lesznek letárolva a mezőtípus típusú tulajdonságértékek és ezen tábla rekordjaira fog mutatni egy-egy bejegyzés a **Terméktáblából**.

5.3.6 A Terméktábla oszlopai:

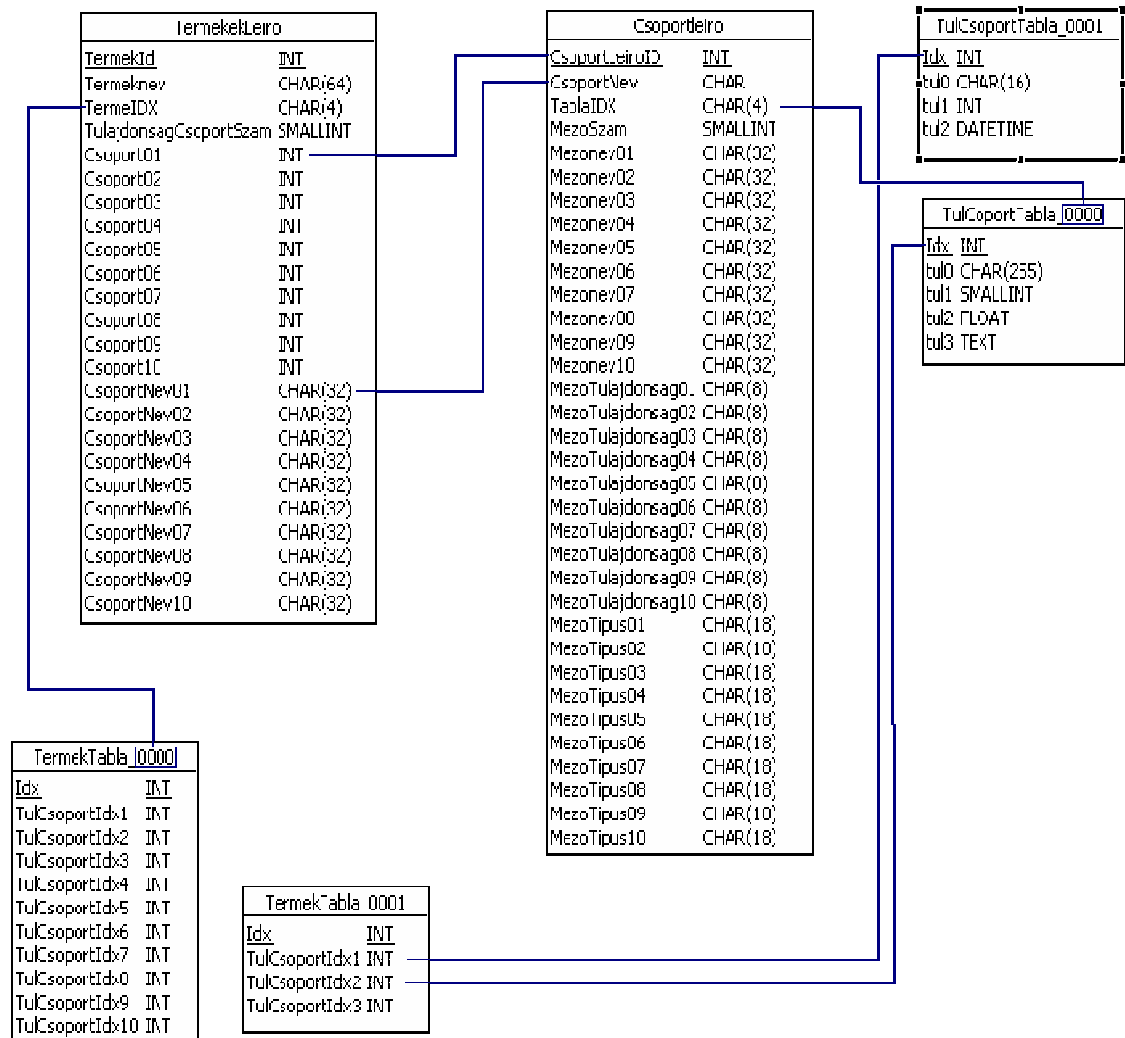
- ``Idx``: int(11) NOT NULL auto_increment : elsődleges kulcs.
- ``TulCsoportIdx``: int(11) : külső kulcs
- ...

Valójában a ``TulCsoportIdx`` csak úgy viselkedik, mint egy külső kulcs. De nem az. Ez a bejegyzés megmutatja, hogy a **Tulajdonság Csoport Tábla** hanyadik sorában van a termékre jellemző rekord.

5.3.7 Az Userinfo tábla oszlopai:

- ``user``: char(11) : felhasználó
- ``passw``: char(32): jelszó

5.3.8 Az adatbázis rajza



Az alkalmazás táblái két termék és két tulajdonságcsoporthoz tartoznak

6 Felhasználói leírás

6.1 Rendszerkövetelmények

6.1.1 Hardver követelmények

A program futtatásának hardveres feltételei hozzávetőlegesen egy 300 MHz-s Intel asztali processzor vagy egy vele ekvivalens teljesítményű más processzorok. A program futásához elegendő 256 Mb. RAM memória.

6.1.2 Szoftver követelmények

(Mind ingyenesen letölthető !)

- Windows operációs rendszer, de valójában bármelyik amelyre .NET Framework telepíthető fel.
- Microsoft .NET Framework: <http://www.microsoft.com/downloads>
- MySql 5.0 : <http://dev.mysql.com/downloads/mysql/5.0.html>
- MySql Query Browser: <http://www.mysql.com/products/tools/query-browser>
- Mysql ODBC Connector 3.51: <http://mysql.sote.hu/Downloads/MyODBC3/mysql-connector-odbc-3.51.14-win32.msi>

6.2 Alkalmazás telepítése:

Magát az alkalmazást nem kell telepíteni, de a MySql adatbázis-kezelő rendszert igen és a hozzávaló MySql Connectort is.

6.2.1 A MySql adatbázis-kezelő rendszer telepítése:

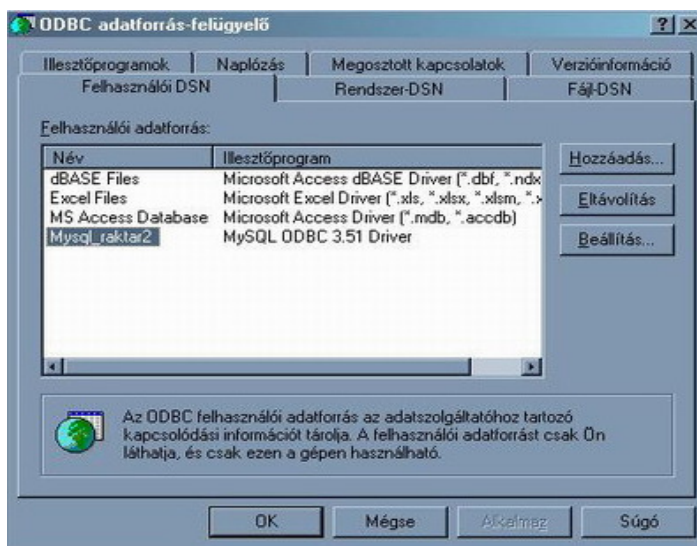
A MySql adatbázis-kezelő rendszer telepítése a szokványos módon történik. Létre kell hozni egy új adatbázist **raktar** néven a **Catalogs** menürészen. Valamint létre kell hozni egy "raktaros" usert az "User administration" menürészen. Aztán a **Schema Privileges**-nél meg kell adni az összes látható jogosultságot.

Ezután a **MySql Query Browser** ben le kell futtatni a programhoz adott **csoportleiro.sql** és **termekleiro.sql**-t. Ez létrehozza a két legfontosabb táblát.

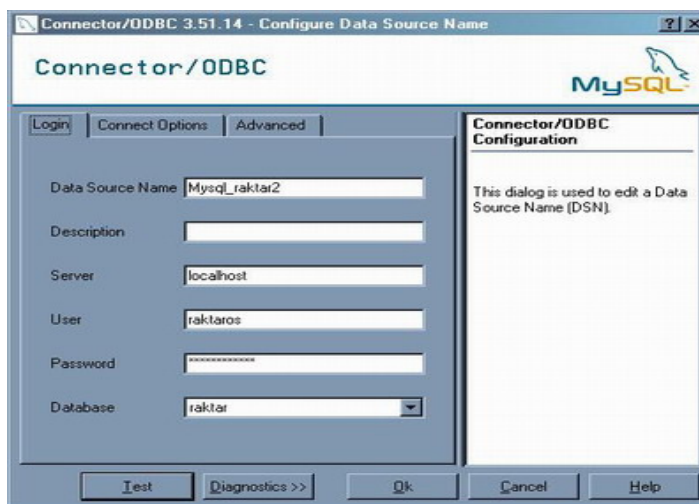
6.2.2 MySql Connector telepítése és konfigurálása:

A telepítés egyszerűen történik a **next** gombok nyomogatásával. A konfigurálása komplikáltabb feladat. El kell indítani az ODBC Adatforrás felügyelőt.

A legegyszerűbb elindítani a parancssorból a következő parancs begépelésével:
„%SystemRoot%\system32\odbcad32.exe”.



ODBC Adatforrás felügyelő (utolsó sorában a MySql Connector)



Ezeket a beállításokat kell megadni a Mysql connectornak.

A .NET Framework nem szorul konfigurálásra. Ezek után az alkalmazást elindíthatjuk.

7 Az alkalmazás használata

7.1 Az Adminisztrátor lehetőségei

- Új tulajdonságcsoporthoz létrehozása
- Új tulajdonságok megadása
- Új terméktípus létrehozása
- Új terméktípus felvitele
- Összetett keresés
- Termék adatainak módosítása
- Egy tulajdonság módosítása az összes termékben

7.2 A Felhasználó lehetőségei

- Összetett keresés
- Engedélyezett mezők módosítása

7.3 Bejelentkezés

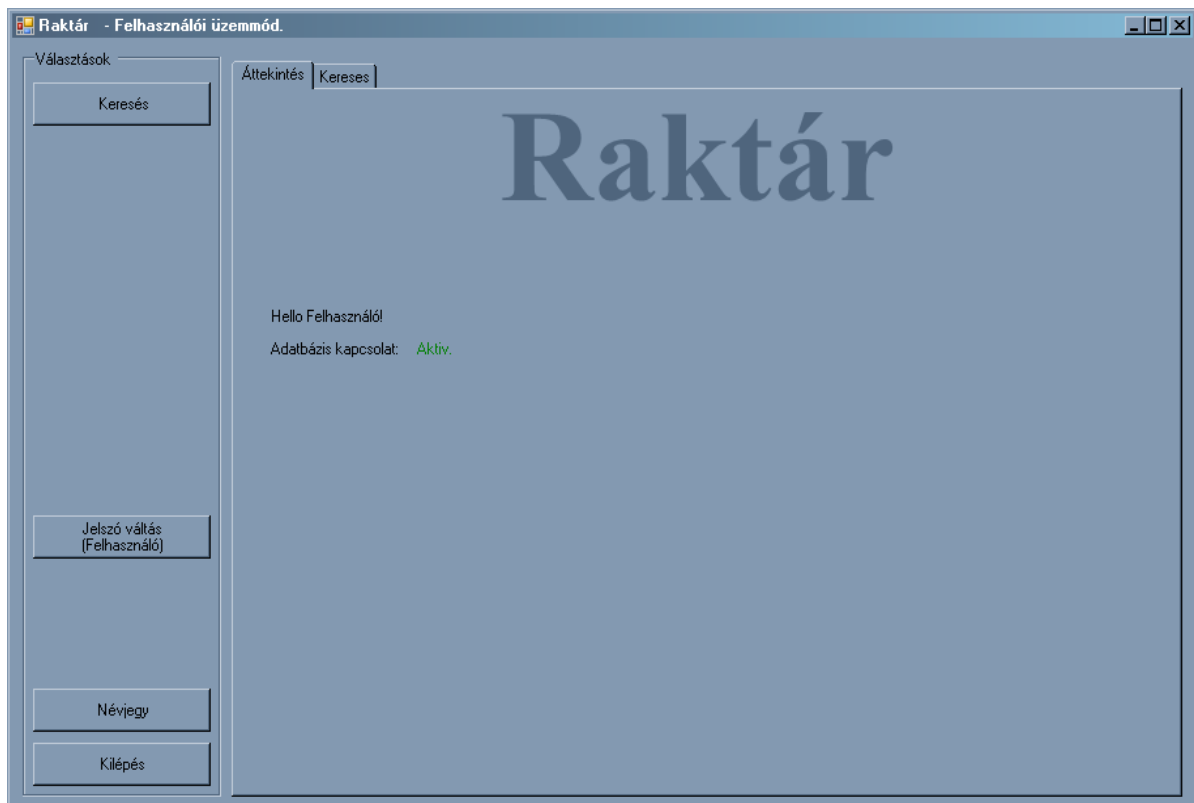
A felhasználónak be kell jelentkeznie mielőtt bármit csinálna. A bejelentkezés mindkét felhasználótípus esetében megegyezik.



Nem szorul magyarázatra az első ablak, a belépés nagyon egyszerűen meg van oldva. Megadjuk a jelszót és **Belépés** gomb. A belépés után adminisztrátor szerepkörrel a következő formot látjuk:

7.4 A Felhasználó lehetőségei:

A **Felhasználó** jelenti itt az egyszerű raktárost akinek lekérdezési és csökkentett módosítási jogai vannak.



7.4.1 Keresés:

Első opció, hogy a keresést lehet végezni az összes tulajdonság csoportban. Ez egyes esetekben nagyon sok rekordot eredményezhet. Ezért további szűkítési lehetőségként ki lehet választani adott tulajdonságcsoporthoz, és akár adott tulajdonságot. Aztán a keresett szövegre rá lehet keresni. A **Törlés** gombbal eltüntetjük az ablakból a kiszelektált rekordokat.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The interface is divided into several sections:

- Választások (Left Sidebar):** Contains buttons for 'Keresés', 'Új termék', 'Termék típusok', 'Tulajdonság csoportok', 'Jelszó váltás (Felhasználó)', 'Jelszó váltás (Adminisztrátor)', 'Névjegy', and 'Kilépés'.
- Navigation Tabs (Top):** 'Áttekintés', 'Keresés' (selected), 'Új termék', 'Termék típusok', and 'Tulajdonság csoportok'.
- A keresés feltételei (Search Conditions):**
 - 'A termék típusa:' dropdown menu with a list showing 'repulogep', 'repulogep', 'helikopter', 'repulogep', and 'harci repülő'.
 - Radio buttons for search scope: 'Keresés az összes tulajdonság csoportban' (selected) and 'Keresés az összes tulajdonság között'.
 - Radio button for 'Keresés adott tulajdonságon belül'.
 - 'Feltétel:' dropdown menu with 'Tartalmaz' selected.
 - 'Keresett szöveg:' text input field.
 - Buttons: 'Keresés', 'Törlés', and 'Eredmény mentése'.
- Keresés (Search Results):** A section titled 'A keresés eredménye' with a large empty area for displaying results.

Megnyomva a **Keresés** gombot kiszelektálódnak a kívánt rekordok. Minden oszlop szerint lehet rendezni növekvő illetve csökkenő sorrendben, törölhető illetve szerkeszthetőek külön-külön a rekordok. A lenti képen éppen egy **repülőgép** termékfajta között próbálunk keresni.

The screenshot shows a software window titled "Raktár - Adminisztrátor üzemmód". It features a sidebar on the left with buttons for "Keresés", "Új termék", "Termék típusok", "Tulajdonság csoportok", "Jelszó váltás (Felhasználó)", "Jelszó váltás (Adminisztrátor)", "Névjegy", and "Kilépés". The main area has a tabbed interface with "Áttekintés", "Keresés", "Új termék", "Termék típusok", and "Tulajdonság csoportok". The "Keresés" tab is active, displaying search criteria: "A termék típusa:" with a dropdown menu showing "repulogep", "repulogep", "repulogep", "helikopter", "repulogep", and "harci repülő"; "Keresés az összes tulajdonság csoportban" (selected) and "Keresés az összes tulajdonság között"; "Keresés adott tulajdonságon belül" with a dropdown for "motorok_szama"; "Feltétel:" with a dropdown for "Tartalmaz"; and "Keresett szöveg:" with a text input field. Below these are buttons for "Keresés", "Törölés", and "Eredmény mentése". At the bottom, there is a section labeled "Keresés" with a sub-header "A keresés eredménye" and a large empty area for results.

7.4.2 Keresés adott tulajdonságcsoporthban

A keresési feltételek szűkítése lehetséges. Megadható, hogy milyen tulajdonságcsoporthban keressünk. A lenti esetben a **Harci_repülő** termékek között akarunk keresni. Kiválasztjuk a **repülőgépteljesítmény** tulajdonságcsoporthot, és azon belül keresünk.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The left sidebar contains buttons for 'Választások', 'Keresés', 'Új termék', 'Termék típusok', 'Tulajdonság csoportok', 'Jelszó váltás (Felhasználó)', 'Jelszó váltás (Adminisztrátor)', 'Névjegy', and 'Kilépés'. The main area has tabs for 'Áttekintés', 'Keresés', 'Új termék', 'Termék típusok', and 'Tulajdonság csoportok'. The 'Keresés' tab is active, showing search criteria: 'A termék típusa' is set to 'harci_repülő'. Under 'A keresés feltételei', there are two radio buttons: 'Keresés az összes tulajdonság csoportban' (selected) and 'Keresés adott tulajdonság csoportban:'. A dropdown menu next to the second radio button shows a list of property groups: 'alapadatok', 'alapadatok', 'repülőgépteljesítmény', and 'beszállító'. Below this, there are fields for 'Feltétel' (set to 'Tartalmaz') and 'Keresett szöveg'. At the bottom, there are buttons for 'Keresés', 'Törlés', and 'Eredmény mentése'. The search results area is currently empty.

7.4.3 Keresés adott tulajdonságcsoporthban, adott tulajdonságon belül

Kiválasztható adott tulajdonságcsoporthban belül milyen tulajdonságoszlopban keresgéljünk értékeket. A képen éppen **harci_repülő**-t keresünk **alapadatok** tulajdonság csoporton belül, és a tulajdonság csoporton belül pedig kiválasztjuk a **raktárkód** tulajdonságot. Mindez szűkíti, pontosítja a találatokat.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The 'Keresés' (Search) tab is active. The search criteria are as follows:

- A termék típusa:** harci_repülő
- Keresés az összes tulajdonság csoportban:** ☐ (unselected)
- Keresés adott tulajdonság csoportban:** ☒ (selected), alapadatok
- Keresés az összes tulajdonság között:** ☐ (unselected)
- Keresés adott tulajdonságon belül:** ☒ (selected), típus
- Feltétel:** Tartalmaz
- Keresett szöveg:** raktárkód

The search results section, titled 'A keresés eredménye', is currently empty. The left sidebar contains buttons for 'Keresés', 'Új termék', 'Termék típusok', 'Tulajdonság csoportok', 'Jelszó váltás (Felhasználó)', 'Jelszó váltás (Adminisztrátor)', 'Névjegyzék', and 'Kilépés'.

7.4.4 Keresés eredménye

A keresés eredménye függ attól, hogy a programot most épp a **felhasználó** használja vagy éppen az **adminisztrátor**. A **felhasználó** csak azokat a tulajdonságokat láthatja amelyeket a termék létrehozója **Publikus** tulajdonsággal látott el.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The 'Keresés' (Search) tab is active. The search criteria are set to 'A termék típusa: harci_repülő'. The search scope is 'Keresés az összes tulajdonság csoportban' and 'Keresés az összes tulajdonság között'. The search condition is 'Tartalmaz' and the search text is empty. The 'Eredmény mentése' (Save results) button is circled in red. The search results table is titled 'A keresés eredménye' and contains the following data:

	Törölés	Szerkesztés	(T1)típus	(T1)raktárkód	(T1)darabsz	(T2)motorok	(T2)motortelj	(T2)hatótáv	(T3)beszállító	(T3)b
▶	Töröl	Szerkeszt	Tomcat	AA00004	14	2	40000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	F16	AA00005	1	1	50000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	Mirage	AB00006	2	2	40000	60000	TWA	1-800
	Töröl	Szerkeszt	Gripen	AB00007	36	1	50000	60000	S.A.A.B	1-800
	Töröl	Szerkeszt	MIG 29	AB00026	21	2	30000	40000	CCCP	1-800

7.4.5 Keresés eredményének riportolása

A program lehetőséget biztosít arra, hogy a felhasználó HTML fájlba kiírja a kiszelektált sorokat. Csak azok a tulajdonságok fognak látszani amelyeket a termék létrehozója **Publikus** módosítóval látott el.

Raktár - Adminisztrátor üzemmód

Választások: Keresés, Új termék, Termék típusok, Tulajdonság csoportok

Áttekintés | Keresés | Új termék | Termék típusok | Tulajdonság csoportok

A keresés feltételei

A termék típusa:

☒ Keresés az összes tulajdonság csoportban ☐ Keresés adott tulajdonság csoportban:

☒ Keresés az összes tulajdonság között ☐ Keresés adott tulajdonságon belül:

Feltétel: Keresett szöveg:

Keresés Törölés **Eredmény mentése**

Keresés

A keresés eredménye

Törölés	Szerkesztés	(T1)típus	(T1)raktárkód	(T1)darabszám	(T2)motorok	(T2)motortípus	(T2)hatótávolság	(T3)beszállító	(T3)beszállító telefonszáma
Töröl	Szerkeszt	Tomcat	AA00004	14	2	40000	60000	Lockheed M	1-800
Töröl	Szerkeszt	F16	AA00005	1	1	50000	60000	Lockheed M	1-800
Töröl	Szerkeszt	Mirage	AB00006	2	2	40000	60000	TWA	1-800
Töröl	Szerkeszt	Gripen	AB00007	36	1	50000	60000	S.A.A.B	1-800
Töröl	Szerkeszt	MIG 29	AB00026	21	2	30000	40000	CCCP	1-800

Keresés eredmény - Mozilla Firefox

File Edit View History Bookmarks Yahoo! Tools Help

file:///C:/Diplomamunka/versionuto/Raktar/bin/Debug/20070420_125107.html

Search Web Mail My Yahoo! Basketball Games Music Answers Personals

(T1)típus	(T1)raktárkód	(T1)darabszám	(T2)motorok_száma	(T2)motorteljesítmény	(T2)hatótávolság	(T3)beszállító_neve	(T3)beszállító_telefonszáma	(T3)beszállító
Tomcat	AA00004	14	2	40000	60000	Lockheed Martin	1-800-654654-65465	Arizona 1121 ajsakjd
F16	AA00005	1	1	50000	60000	Lockheed Martin	1-800-654654-65465	Arizona 1121 ajsakjd
Mirage	AB00006	2	2	40000	60000	TWA	1-800-654654-65465	Arizona 1121 ajsakjd
Gripen	AB00007	36	1	50000	60000	S.A.A.B	1-800-654654-65465	Stockholm
MIG 29	AB00026	21	2	30000	40000	CCCP	1-800-654654-65465	MOCKBA

7.4.6 A szerkesztés:

A szerkesztésben felugró ablakban láthatjuk a régi értékeket, és az új oszlopban megadhatunk újakat. Az ismerős **H** betűt megnyomva előugrik a már meglévő értékek halmaza. Ha kipipáljuk a bal alsó sarokban levő négyzetet, akkor lehetőségünk nyílik arra, hogy az adott értékek minden értéktáblában átváltozzanak erre az új megadott értékre. Ez akkor jó ha egy terméknek van egy beszállítója, annak van egy megadott telefonszáma, és az megváltozik.

Választások

Keresés

Új termék

Termék típusok

Tulajdonság csoportok

Jelszó váltás (Felhasználó)

Jelszó váltás (Adminisztrátor)

Névjegy

Kilépés

Áttekintés Keresés Új termék Termék típusok Tulajdonság csoportok

A keresés feltételei

A termék típusa: harci_repülő

☒ Keresés az összes tulajdonság csoportban ☐ Keresés adott tulajdonság csoportban: alapadatok

☒ Keresés az összes tulajdonság között ☐ Keresés adott tulajdonságon belül: típus

Feltétel: Tartalmaz Keresett szöveg:

Keresés Törlés Eredmény mentése

Keresés

A keresés eredménye

	Töröl	Szerkesztés	(T1)típus	(T1)raktárkó	(T1)darabsz	(T2)motorok	(T2)motortelj	(T2)hatotavo	(T3)beszállít	(T3)bs
►	Töröl	Szerkeszt	Tomcat	AA00004	14	2	40000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	F16	AA00005	1	1	50000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	Mirage	AB00006	2	2	40000	60000	TWA	1-800
	Töröl	Szerkeszt	Gripen	AB00007	36	1	50000	60000	S.A.A.B	1-800
	Töröl	Szerkeszt	MIG 29	AB00026	21	2	30000	40000	CCCP	1-800

Raktár - Adminisztrátor üzemmód

Választások:

Szerkesztés

Termék tulajdonságainak szerkesztése

Tulajdonságcsoport: alapadatok

H típus	(Szöveg (255)/Publikus)	Tomcat	Tomcat
raktárkód	(Szöveg (255)/Publikus)	AA00004	AA00004
darabszám	(Egész szám (nagy)/Publikus)	14	14

Tulajdonságcsoport: repulogepteljesítmény

H motorok_száma	(Egész szám (kicsi)/Publikus)	2	2
motor teljesítmény	(Egész szám (nagy)/Publikus)	40000	40000
hatótávolság	(Egész szám (nagy)/Publikus)	60000	60000

Tulajdonságcsoport: beszállító

H beszállító_neve	(Szöveg (64)/Privát)	Lockheed Martin	Lockheed Martin
beszállító_telefonszáma	(Szöveg (64)/Privát)	1-800-654654-65465	1-800-654654-65465
beszállító_címe	(Szöveg (255)/Védet)	Arizona 1121 ajsakjd	Arizona 1121 ajsakjd

☐ Változtatás a tulajdonság táblákban is

Ez a kicsi ablak ugrik fel, amikor a **H** betűt megnyomjuk. A már létező értékekkel itt szerkeszthetjük át a kisselektált sort.

Tulajdonságcsoport: alapadatok

Létező tulajdonság kiválasztása

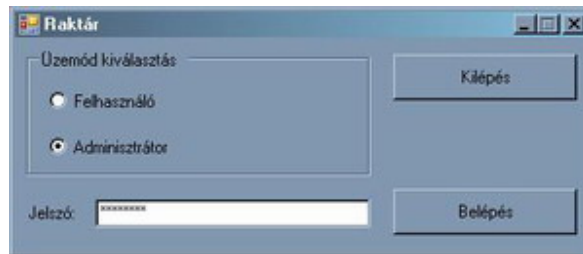
Tulajdonság: (Tomcat AA00003 13)
Tulajdonság: (Tomcat AA00004 14)
Tulajdonság: (F16 AA00005 1)
Tulajdonság: (Mirage AB00006 2)
Tulajdonság: (Grippen AB00007 36)
Tulajdonság: (MIG 29 AB00026 21)

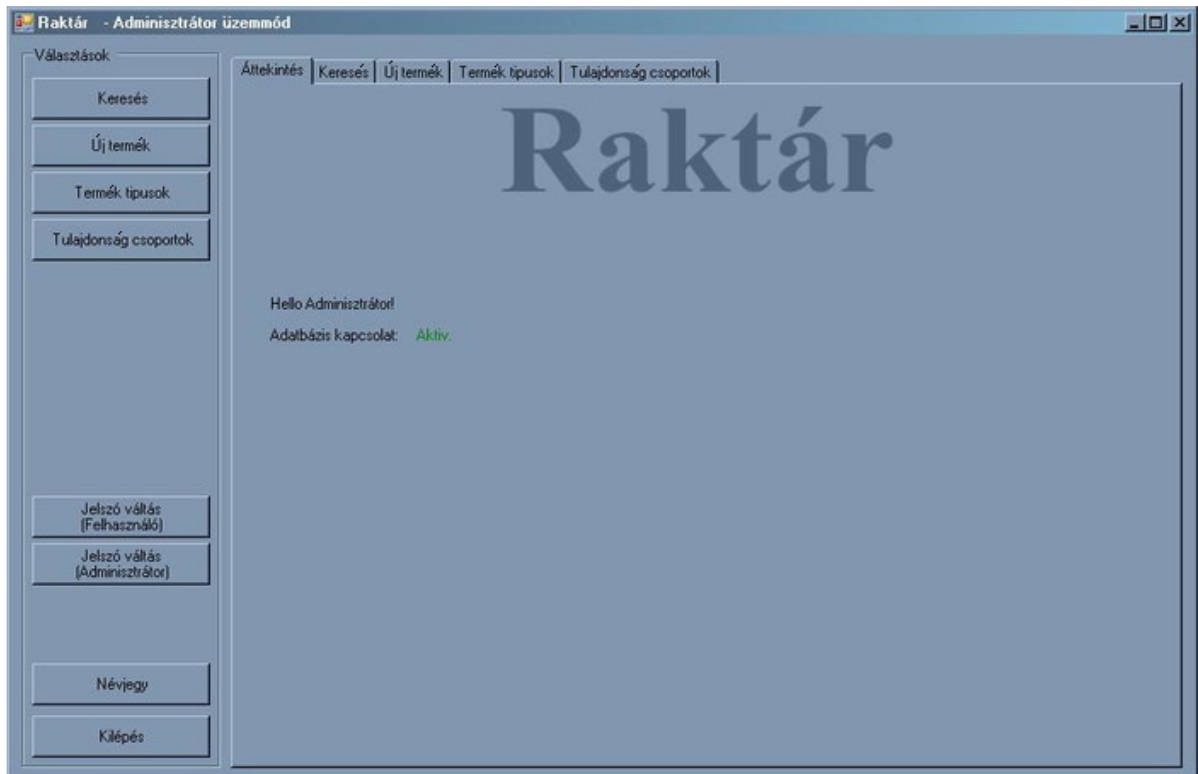
7.5 Az adminisztrátor lehetőségei:

7.5.1 Az adminisztrátor kezdőfelülete:

Látható a kép balközép részén zöld színnel, hogy az adatbázis kapcsolat **Aktív**. Ha bármi probléma adódik a kapcsolattal a felírás átvált piros színre és **Hibás** felírra.

Bejelentkezési ablak:





7.5.2 Jelszóváltoztatás

A jelszóváltoztatás a megszokott módon történik. Megadva a régi jelszót és kétszer az újat.

Raktár - Adminisztrátor üzemmód

Választások:

Keresés

Új termék

Termék típusok

Tulajdonság csoportok

Áttekintés | Keresés | Új termék | Termék típusok | Tulajdonság csoportok

Raktár

Hello Adminisztrátor!

Adatbázis kapcsolat: **Aktiv**

Jelszavak

Adminisztrátor jelszavának megváltoztatása...

Régi jelszó:

Új jelszó:

Új jelszó megegyezzer:

Rendben Mégsem

Jelszó váltás (Felhasználó)

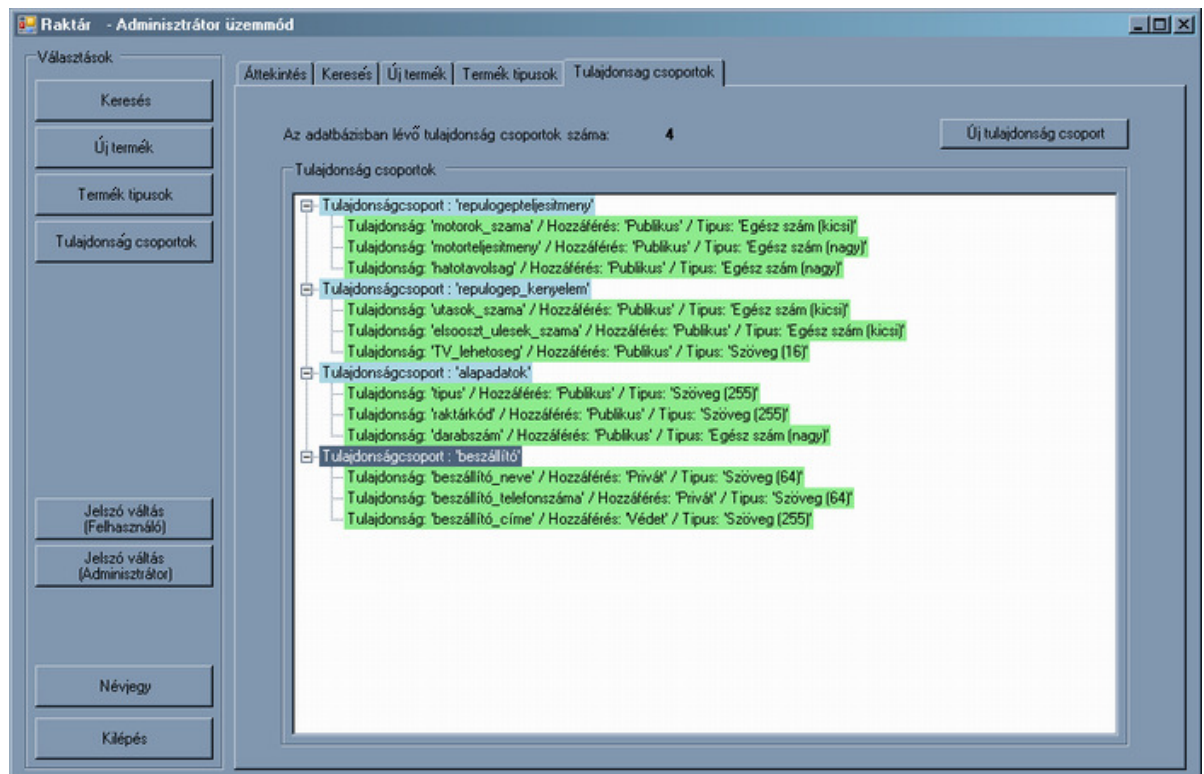
Jelszó váltás (Adminisztrátor)

Névjegy

Kilépés

7.5.3 Tulajdonságcsoportok felvitele

Kezdjük a leghátsó fültől, az adatbázis feltöltésének irányával vezérelve. A **Tulajdonság csoportok** fülre kattintva látjuk ezt. Itt vihetünk be új tulajdonságcsoportokat. Az aktuális ablakon a kék színnel beszínezett sorok a tulajdonságcsoportok neveit jelzik, a világoszöldek pedig az egyes csoportokba szervezett tulajdonságokat.



A következő beugróablak segítségével új tulajdonságcsoporthoz adunk az adatbázishoz. Megadjuk a tulajdonságcsoporthoz a tulajdonságok nevét, láthatóságát, és típusát.

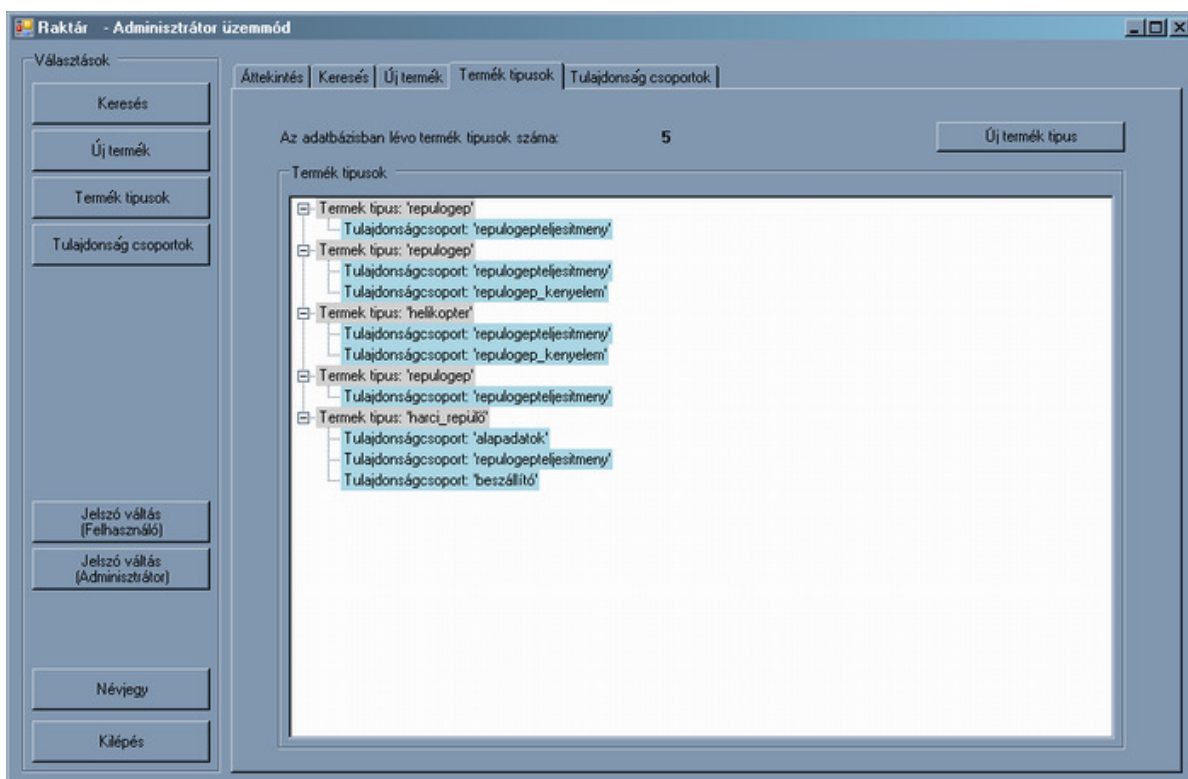
The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The 'Tulajdonság csoportok' tab is selected. The 'Új termék tulajdonság' dialog box is open, showing the 'Termék tulajdonság csoport jellemzői' section. The 'Tulajdonság csoport neve' field contains 'karbantartás'. Below this is a table titled 'Tulajdonság mezők jellemzői'.

Tulajdonság neve	Hozzáférés	Típus
felújítás dátuma	Publikus	Dátum
használt alkatrészek	Publikus	Szöveg (2048)
▶ felújítás összköltsége	Publikus	Valós szám
*		

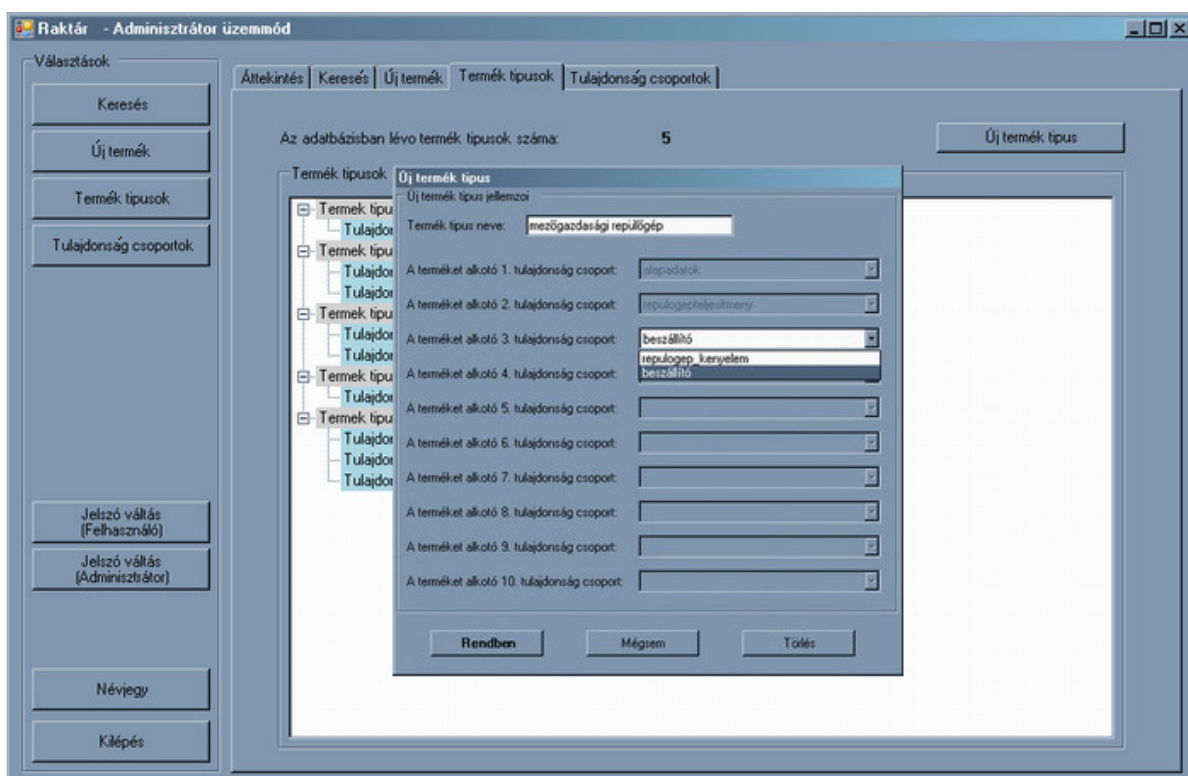
At the bottom of the dialog are three buttons: 'Rendben', 'Mégsem', and 'Törölés'. On the right side of the main window, there is a section for 'Új tulajdonság csoport'.

7.5.4 Terméktípusok definiálása

A következő ablak kilistázza a már létező terméktípusokat. Szürke színnel a terméktípusokat, világoskékkel a terméktípusok felépítésében résztvevő tulajdonságcsoportokat. Ebből az ablakból juthatunk el az új terméktípus felvitelére szolgáló ablakhoz.

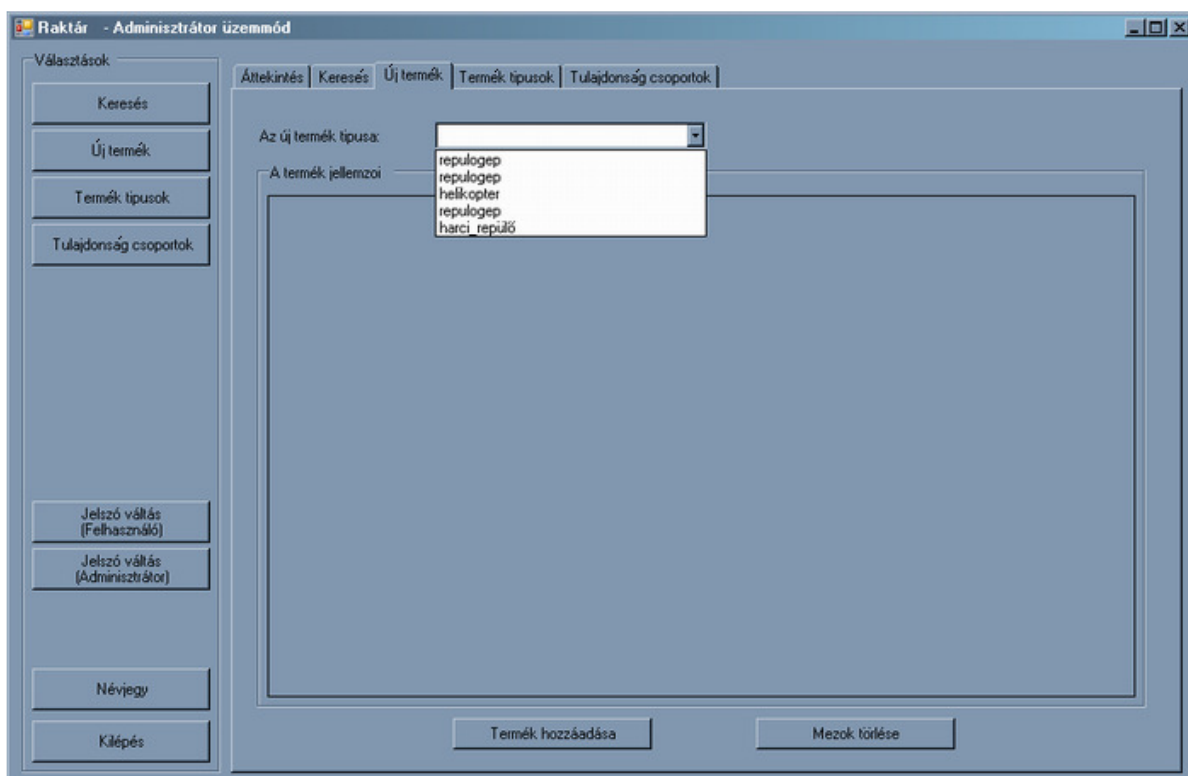


Ebben a felugró ablakban lehet új terméktípust létrehozni. Értelmszerűen meg kell adni a terméktípus nevét, aztán kiválogatni a már felvitt tulajdonságcsoporthok közül maximum 10-et, amelyek megadják a terméktípus vázát. A program legördülő menüvel biztosítja a könnyebb tulajdonságcsoporth kiválasztást.



7.5.5 Termékek felvitele

Ebben az ablakban nyílik lehetőség az előre definiált terméktípusokból válogatni, és konkrét termék „példányokat” létrehozni. Felvihetünk többször is ugyanabból a termékből, de egyszerűen csinálhatunk is egy **darabszám** oszlopot, amit majd változtatunk termék raktárból kikerülésekor.



A terméktípusok konkretizálása csak a terméktípus kiválasztása után történik. Kiválasztva a terméktípust, egyből bejönnek a terméktípusok kitöltendő tulajdonságai csoportokba szervezve. Kitöltjük a megfelelő adatokkal, majd **Termék hozzáadása** gombbal bevisszük az adatbázisba. Ha nem tetszik, akkor **Mezők törlése** gombbal minden mezőt törölünk.

A termékek adatait már létező értékhalmozatokkal is feltölthetjük. Például, ha már ilyen típusú termék létezik az adatbázisban, akkor ennek az adataival feltölthetünk egy új terméket. A következő ablakban például egy „helikopter”-t akarunk felvinni. De a “teljesítmény” nevű tulajdonságcsoporthoz már létező adatokkal kívánjuk feltölteni. Ezért az oda tartozó **H** betűre kattintunk, és a felugró ablakból kiválasztjuk a kívánt adathalmazt. Jelen esetben az {1,2,3} – at.

Raktár - Adminisztrátor üzemmód

Választások:

Keresés

Új termék

Termék típusok

Tulajdonság csoportok

Áttekintés | Keresés | Új termék | Termék típusok | Tulajdonság csoportok

Az új termék típusa:

A termék jellemzői

Tulajdonságcsoport: alapadatok

H típus (Szöveg (255)/Publikus)

raktárkód

darabszám

Tulajdonságcsop **H** motorok_sz

motor teljes hatótávolsága

Tulajdonságcsop **H** beszállító_neve

beszállító telefonszáma (Szöveg (64)/Privát)

beszállító címe (Szöveg (255)/Védet)

Tulajdonságcsoport: repulogepteljesitmeny

Létező tulajdonság kiválasztása

Tulajdonság:	(2 3 4)
Tulajdonság:	(7 8 9)
Tulajdonság:	(6 6 5)
Tulajdonság:	(1 2 3)
Tulajdonság:	(4 5 6)
Tulajdonság:	(1 2 3)

Rendben Mégsem

Termék hozzáadása Mezők törlése

Jelszó váltás (Felhasználó)

Jelszó váltás (Adminisztrátor)

Névjegy

Kilépés

7.5.6 Keresés:

Első opció, hogy a keresést lehet végezni az összes tulajdonság csoportban. Ez egyes esetekben nagyon sok rekordot eredményezhet. Ezért további szűkítési lehetőségként ki lehet választani adott tulajdonságcsoporthoz, és akár adott tulajdonságot. Aztán a keresett szövegre rá lehet keresni. További opciók, hogy a vizsgálhatunk tartalmazást, kisebb e a keresett szám vagy nagyobb, vagy egyenlőséget. A **Törlés** gombbal eltüntetjük az ablakból a kisselektált rekordokat.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The interface is divided into several sections:

- Választások (Left Sidebar):** Contains buttons for 'Keresés', 'Új termék', 'Termék típusok', 'Tulajdonság csoportok', 'Jelszó váltás (Felhasználó)', 'Jelszó váltás (Adminisztrátor)', 'Névjegy', and 'Kilépés'.
- Navigation Tabs (Top):** 'Áttekintés', 'Keresés' (selected), 'Új termék', 'Termék típusok', and 'Tulajdonság csoportok'.
- A keresés feltételei (Search Conditions):**
 - 'A termék típusa:' dropdown menu with 'repülőgép' selected. A list of options is visible: repülőgép, repülőgép, helikopter, repülőgép, harci repülő.
 - Radio buttons for search scope: 'Keresés az összes tulajdonság csoportban' (selected) and 'Keresés az összes tulajdonság között'.
 - Radio button for search type: 'Keresés adott tulajdonságon belül'.
 - 'Feltétel' dropdown menu with 'Tartalmaz' selected.
 - 'Keresett szöveg:' text input field.
 - Buttons: 'Keresés', 'Törlés', and 'Eredmény mentése'.
- Keresés (Search Results):** A section titled 'A keresés eredménye' with a large empty area for displaying results.

Megnyomva a **Keresés** gombot kiszelektálódnak a kívánt rekordok. Minden oszlop szerint lehet rendezni növekvő illetve csökkenő sorrendben, törölhető illetve szerkeszthetőek külön-külön a rekordok. A lenti képen éppen egy „repülőgép” termékfajta között próbálunk keresni.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The interface is divided into several sections:

- Választások (Left Sidebar):** Contains buttons for 'Keresés', 'Új termék', 'Termék típusok', 'Tulajdonság csoportok', 'Jelszó váltás (Felhasználó)', 'Jelszó váltás (Adminisztrátor)', 'Névjegy', and 'Kilépés'.
- Navigation Tabs (Top):** 'Áttekintés', 'Keresés' (selected), 'Új termék', 'Termék típusok', and 'Tulajdonság csoportok'.
- A keresés feltételei (Search Conditions):**
 - 'A termék típusa:' dropdown menu is open, showing options: 'repulogep', 'repulogep', 'repulogep', 'helikopter', 'repulogep', 'harci repulo'.
 - Radio buttons for search scope: 'Keresés az összes tulajdonság csoportban' (selected) and 'Keresés az összes tulajdonság között'.
 - Radio button for search type: 'Keresés adott tulajdonságon belül'.
 - 'Feltétel' dropdown menu is set to 'Tartalmaz'.
 - 'Keresett szöveg:' text input field.
 - Buttons: 'Keresés', 'Törölés', and 'Eredmény mentése'.
- Keresés (Results Section):** A large empty area with the header 'A keresés eredménye'.

7.5.7 Keresés adott tulajdonságcsoporthban

A keresési feltételek szűkítése lehetséges. Megadható, hogy milyen tulajdonságcsoporthban keressünk. A lenti esetben a **Harci_repülő** termékek között akarunk keresni. Kiválasztjuk a **repülőgépteljesítmény** tulajdonságcsoporthot, és azon belül keresünk.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The 'Keresés' (Search) tab is active. The search criteria are set to 'A termék típusa: harci_repülő'. The search scope is 'Keresés adott tulajdonság csoportban:' (Search in specific property group), with the selected group being 'repülőgépteljesítmény'. The search condition is 'Tartalmaz' (Contains). The search text field is empty. The search results area is currently empty.

Választások

- Keresés
- Új termék
- Termék típusok
- Tulajdonság csoportok

Áttekintés | **Keresés** | **Új termék** | **Termék típusok** | **Tulajdonság csoportok**

A keresés feltételei

A termék típusa:

☐ Keresés az összes tulajdonság csoportban ☒ Keresés adott tulajdonság csoportban:

☒ Keresés az összes tulajdonság között ☐ Keresés adott tulajdonságon belül:

Feltétel: Keresett szöveg:

Keresés

A keresés eredménye

7.5.8 Keresés adott tulajdonságcsoporthban, adott tulajdonságon belül

Kiválasztható adott tulajdonságcsoporthban belül milyen tulajdonságoszlopban keresgéljünk értékeket. A képen éppen **harci_repülő**-t keresünk **alapadatok** tulajdonság csoporton belül, és a tulajdonság csoporton belül pedig kiválasztjuk a **raktárkód** tulajdonságot. Mindez szűkíti, pontosítja a találatokat.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The 'Keresés' (Search) tab is active. The search criteria are as follows:

- A termék típusa:** harci_repülő
- Keresés az összes tulajdonság csoportban:** ☐ (unselected)
- Keresés adott tulajdonság csoportban:** ☒ (selected), alapadatok
- Keresés az összes tulajdonság között:** ☐ (unselected)
- Keresés adott tulajdonságon belül:** ☒ (selected), típus
- Feltétek:** Tartalmaz
- Keresett szöveg:** raktárkód

The search results area, titled 'A keresés eredménye', is currently empty. The left sidebar contains buttons for 'Keresés', 'Új termék', 'Termék típusok', 'Tulajdonság csoportok', 'Jelszó váltás (Felhasználó)', 'Jelszó váltás (Adminisztrátor)', 'Névjegy', and 'Kilépés'.

7.5.9 Keresés eredménye

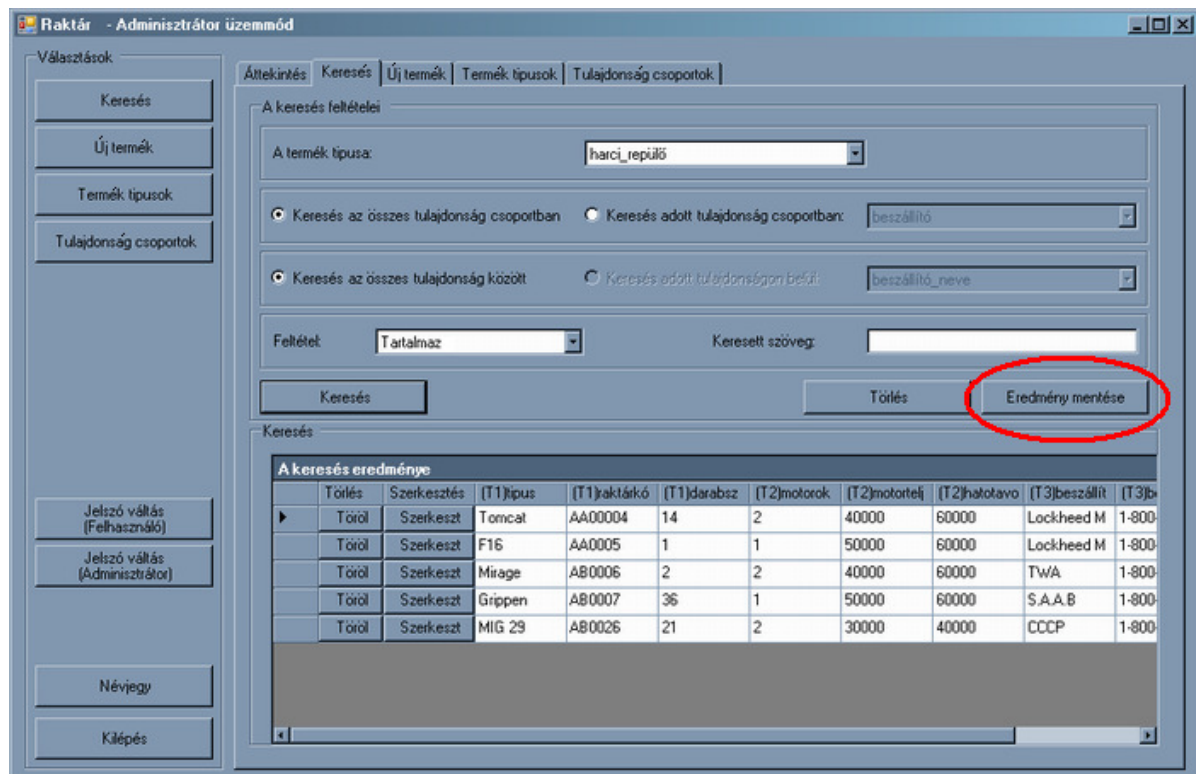
A keresés eredménye függ attól, hogy a programot most épp a **felhasználó** használja vagy éppen az **adminisztrátor**. A **felhasználó** csak azokat a tulajdonságokat láthatja amelyeket a termék létrehozója **Publikus** tulajdonsággal látott el.

The screenshot shows the 'Raktár - Adminisztrátor üzemmód' application window. The 'Keresés' (Search) tab is active. The search criteria are set to 'A termék típusa: harci_reptülő'. The search scope is 'Keresés az összes tulajdonság csoportban' and 'Keresés az összes tulajdonság között'. The search filter is 'Tartalmaz' and the search text is empty. The 'Eredmény mentése' (Save results) button is circled in red. The search results table is displayed below.

	Törölés	Szerkesztés	(T1)típus	(T1)raktárkód	(T1)darabsz	(T2)motorok	(T2)motortelj	(T2)hatótáv	(T3)beszállít	(T3)bs
▶	Töröl	Szerkeszt	Tomcat	AA00004	14	2	40000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	F16	AA00005	1	1	50000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	Mirage	AB00006	2	2	40000	60000	TWA	1-800
	Töröl	Szerkeszt	Gripen	AB00007	36	1	50000	60000	S.A.A.B	1-800
	Töröl	Szerkeszt	MIG 29	AB00026	21	2	30000	40000	CCCP	1-800

7.5.10 Keresés eredményének riportolása

A program lehetőséget biztosít arra, hogy a felhasználó HTML fájlba kiírja a kiszelektált sorokat. Csak azok a tulajdonságok fognak látszani amelyeket a termék létrehozója **Publikus** módosítóval látott el.



Keresés eredmény - Mozilla Firefox

File Edit View History Bookmarks Yahoo! Tools Help

file:///C:/Diplomamunka/versionuto/Raktar/bin/Debug/20070420_125107.html

Search Web Mail My Yahoo! Basketball Games Music Answers Personals

(T1)típus	(T1)raktárkód	(T1)darabszám	(T2)motorok_száma	(T2)motorteljesítmény	(T2)hatótávolság	(T3)beszállító_neve	(T3)beszállító_telefonszáma	(T3)beszállító
Tomcat	AA00004	14	2	40000	60000	Lockheed Martin	1-800-654654-65465	Arizona 1121 ajsakjd
F16	AA00005	1	1	50000	60000	Lockheed Martin	1-800-654654-65465	Arizona 1121 ajsakjd
Mirage	AB00006	2	2	40000	60000	TWA	1-800-654654-65465	Arizona 1121 ajsakjd
Grippen	AB00007	36	1	50000	60000	S.A.A.B	1-800-654654-65465	Stockholm
MIG 29	AB00026	21	2	30000	40000	CCCP	1-800-654654-65465	MOCKBA

7.5.11 A szerkesztés:

A szerkesztésben felugró ablakban láthatjuk a régi értékeket, és az új oszlopban megadhatunk újakat. Az ismerős **H** betűt megnyomva előugrik a már meglévő értékek halmaza. Ha kipipáljuk a bal alsó sarokban levő négyzetet, akkor lehetőségünk nyílik arra, hogy az adott értékek minden értéktáblában átváltozzanak erre az új megadott értékre. Ez akkor jó ha egy terméknek van egy beszállítója, annak van egy megadott telefonszáma, és az megváltozik.

Választások:

- Keresés
- Új termék
- Termék típusok
- Tulajdonság csoportok

Áttekintés Keresés Új termék Termék típusok Tulajdonság csoportok

A keresés feltételei

A termék típusa: harci_repülő

☒ Keresés az összes tulajdonság csoportban ☐ Keresés adott tulajdonság csoportban: alapadatok

☒ Keresés az összes tulajdonság között ☐ Keresés adott tulajdonságon belül: típus

Feltétet: Tartalmaz Keresett szöveg:

Keresés Törlés Eredmény mentése

Keresés

A keresés eredménye

	Törölés	Szerkesztés	(T1)típus	(T1)raktárkó	(T1)darabsz	(T2)motorok	(T2)motortelj	(T2)hatotavo	(T3)beszállít	(T3)br
▶	Töröl	Szerkeszt	Tomcat	AA00004	14	2	40000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	F16	AA00005	1	1	50000	60000	Lockheed M	1-800
	Töröl	Szerkeszt	Mirage	AB00006	2	2	40000	60000	TWA	1-800
	Töröl	Szerkeszt	Gripen	AB00007	36	1	50000	60000	S.A.A.B	1-800
	Töröl	Szerkeszt	MIG 29	AB00026	21	2	30000	40000	CCCP	1-800

Névjegy

Kilépés

Raktár - Adminisztrátor üzemmód

Választások:

Szerkesztés

Termék tulajdonságainak szerkesztése

Tulajdonságcsoport: alapadatok

H típus	(Szöveg (255)/Publikus)	Tomcat	Tomcat
raktárkód	(Szöveg (255)/Publikus)	AA00004	AA00004
darabszám	(Egész szám (nagy)/Publikus)	14	14

Tulajdonságcsoport: repulogepteljesítmény

H motorok_száma	(Egész szám (kicsi)/Publikus)	2	2
motor teljesítmény	(Egész szám (nagy)/Publikus)	40000	40000
hatótávolság	(Egész szám (nagy)/Publikus)	60000	60000

Tulajdonságcsoport: beszállító

H beszállító neve	(Szöveg (64)/Privát)	Lockheed Martin	Lockheed Martin
beszállító telefonszáma	(Szöveg (64)/Privát)	1-800-654654-65465	1-800-654654-65465
beszállító címe	(Szöveg (255)/Védet)	Arizona 1121 ajsakjd	Arizona 1121 ajsakjd

☐ Változtatás a tulajdonság táblákban is

Ez a kicsi ablak ugrik fel, amikor a **H** betűt megnyomjuk. A már létező értékekkel itt szerkeszthetjük át a kiszelektált sort.

Tulajdonságcsoport: alapadatok

Létező tulajdonság kiválasztása

Tulajdonság: (Tomcat AA00003 13)
Tulajdonság: (Tomcat AA00004 14)
Tulajdonság: (F16 AA00005 1)
Tulajdonság: (Mirage AB00006 2)
Tulajdonság: (Grippen AB00007 36)
Tulajdonság: (MIG 29 AB00026 21)

8 Programozói leírás

8.1.1 A szükséges alkalmazások

- Windows operációs rendszer, de valójában bármelyik amelyre .NET Framework telepíthető fel.
- Microsoft .NET Framework: <http://www.microsoft.com/downloads>
- MySql 5.0 : <http://dev.mysql.com/downloads/mysql/5.0.html>
- MySql Buery Browser: <http://www.mysql.com/products/tools/query-browser>
- Mysql ODBC Connector 3.51: <http://mysql.sote.hu/Downloads/MyODBC3/mysql-connector-odbc-3.51.14-win32.msi>

8.2 Alkalmazás telepítése

Magát az alkalmazást nem kell telepíteni, de a MySql adatbázis-kezelő rendszert igen és a hozzávaló MySql Connectort is.

8.2.1 A MySql adatbázis-kezelő rendszer telepítése

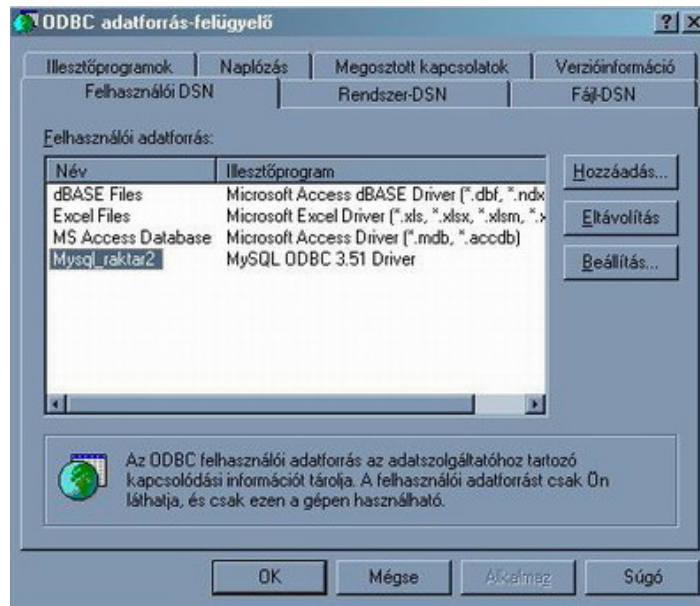
A MySql adatbázis-kezelő rendszer telepítése a szokványos módon történik. Létre kell hozni egy új adatbázist benne **raktar** néven a **Catalogs** menürészen. Valamint létre kell hozni egy **raktaros** usert az **User administration** menürészen. Aztán a **Schema Privileges**-nél meg kell adni az összes látható jogosultságot.

Ezután a **MySql Query Browser** ben le kell futtatni a programhoz adott **csoportleiro.sql** és **termekleiro.sql**-t. Ez létrehozza a két legfontosabb táblát.

8.2.2 MySql Connector telepítése és konfigurálása:

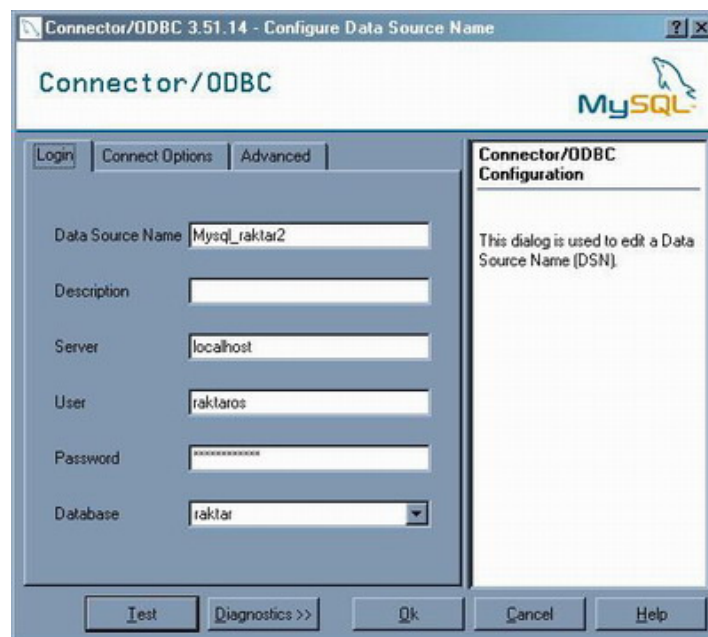
A telepítés egyszerűen történik a **next** gombok nyomogatásával. A konfigurálása komplikáltabb feladat. El kell indítani az ODBC Adatforrás felügyelőt.

A legegyszerűbb elindítani a parancssorból a következő parancs begépelésével:
„%SystemRoot%\system32\odbcad32.exe”.



ODBC Adatforrás felügyelő (utolsó sorában a MySql Connector)

A következő beállításokat kell megadni:



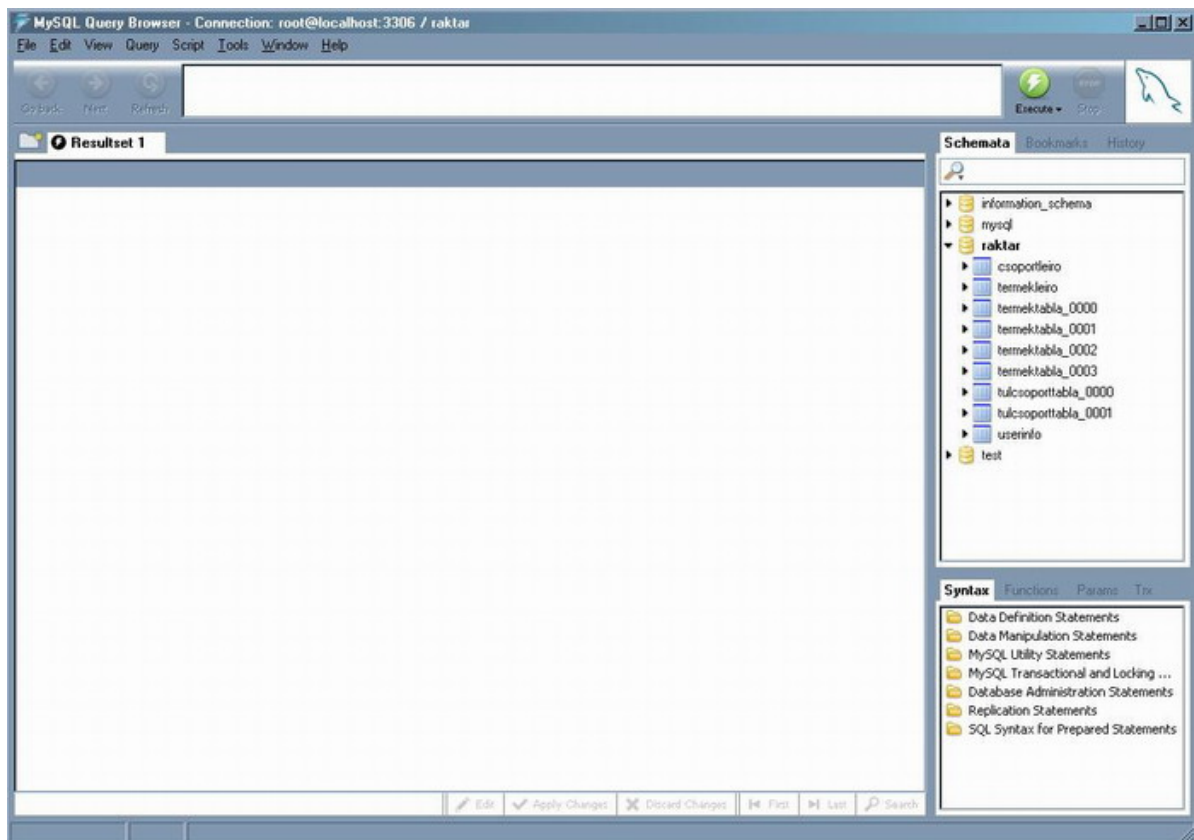
A .NET Framework nem szorul konfigurálásra. Ezek után az alkalmazást elindíthatjuk.

8.2.3 Szkriptek futtatása

Ahhoz, hogy az adatbázisunk inicializálva legyen, szükséges lefuttatni a következő szkripteket:

- csoportleiro.sql
- termekleiro.sql
- userinfo.sql

A szkripteket a MySql Query Browser-ben futtatjuk le. Segítségként lássuk a munkafelületet:



Jobbklikk az **Execute** melletti keskenyebb sávba, válasszuk a **open query** opciót majd kikereshetjük a megfelelő Sql szkriptet. A létrejött táblák megjeleníthetők a jobboldali függőleges oszlopban egy **refresh** gombot nyomva.

8.2.4 Az inicializáló szkriptek

A csoportleíró tábla inicializáló szkriptje (csoportleiro.sql):

```
CREATE TABLE `raktar`.`csoportleiro` (  
  `CsoportLeiroID` int(11) NOT NULL auto_increment,  
  `CsoportNev` char(32) default "",  
  `TablaIDX` char(4) default '0000',  
  `MezoSzam` smallint(6) default NULL,  
  `MezoNev01` char(32) default "",  
  `MezoNev02` char(32) default "",  
  `MezoNev03` char(32) default "",  
  `MezoNev04` char(32) default "",  
  `MezoNev05` char(32) default "",  
  `MezoNev06` char(32) default "",  
  `MezoNev07` char(32) default "",  
  `MezoNev08` char(32) default "",  
  `MezoNev09` char(32) default "",  
  `MezoNev10` char(32) default "",  
  `MezoTulajdonsag01` char(8) default "",  
  `MezoTulajdonsag02` char(8) default "",  
  `MezoTulajdonsag03` char(8) default "",  
  `MezoTulajdonsag04` char(8) default "",  
  `MezoTulajdonsag05` char(8) default "",  
  `MezoTulajdonsag06` char(8) default "",  
  `MezoTulajdonsag07` char(8) default "",  
  `MezoTulajdonsag08` char(8) default "",  
  `MezoTulajdonsag09` char(8) default "",  
  `MezoTulajdonsag10` char(8) default "",  
  `MezoTipus01` char(18) default "",  
  `MezoTipus02` char(18) default "",  
  `MezoTipus03` char(18) default "",  
  `MezoTipus04` char(18) default "",  
  `MezoTipus05` char(18) default "",  
  `MezoTipus06` char(18) default "",  
  `MezoTipus07` char(18) default "",  
  `MezoTipus08` char(18) default "",  
  `MezoTipus09` char(18) default "",  
  `MezoTipus10` char(18) default "",  
  PRIMARY KEY (`CsoportLeiroID`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

8.2.5 A termékleíró tábla inicializáló szkriptje (termekleiro.sql):

```
CREATE TABLE `raktar`.`termekleiro` (  
  `TermekId` int(11) NOT NULL auto_increment,  
  `Termeknev` char(64) default NULL,  
  `TermekIDX` char(4) default NULL,  
  `TulajdonsagCsoportSzam` smallint(6) default NULL,  
  `Csoport01` int(11) default '0',  
  `Csoport02` int(11) default '0',  
  `Csoport03` int(11) default '0',  
  `Csoport04` int(11) default '0',  
  `Csoport05` int(11) default '0',  
  `Csoport06` int(11) default '0',  
  `Csoport07` int(11) default '0',  
  `Csoport08` int(11) default '0',  
  `Csoport09` int(11) default '0',  
  `Csoport10` int(11) default '0',  
  `CsoportNev01` char(32) default NULL,  
  `CsoportNev02` char(32) default NULL,  
  `CsoportNev03` char(32) default NULL,  
  `CsoportNev04` char(32) default NULL,  
  `CsoportNev05` char(32) default NULL,  
  `CsoportNev06` char(32) default NULL,  
  `CsoportNev07` char(32) default NULL,  
  `CsoportNev08` char(32) default NULL,  
  `CsoportNev09` char(32) default NULL,  
  `CsoportNev10` char(32) default NULL,  
  PRIMARY KEY (`TermekId`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

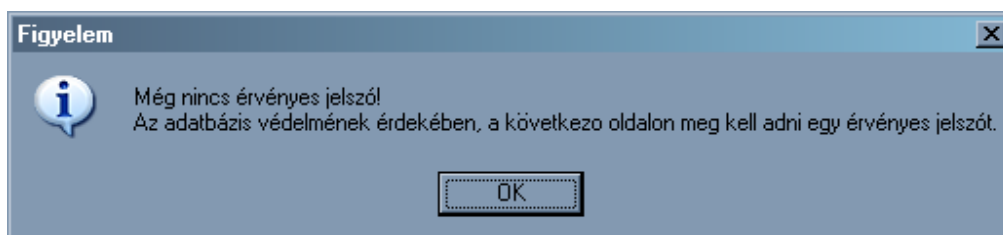
Az Userinfo tábla inicializáló scriptje (userinfo.sql) :

```
CREATE TABLE raktar.userinfo (user char(16) default NULL, passw char(32) default  
NULL);  
INSERT INTO raktar.userinfo SET user='admin';  
INSERT INTO raktar.userinfo SET user='user';
```

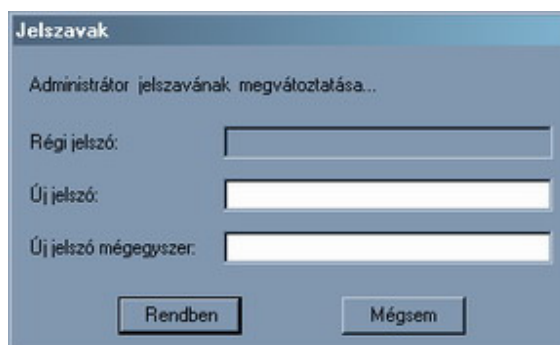
9 A rendszer biztonsága

9.1 Jelszavak kódolása, első bejelentkezés

A jelszavak kódolásához az MD5 kriptográfiai technológiát használom. Ennek a technológiának a lényege, hogy a jelszóból egy hash függvénnyel egy 128 bites kódot generál. Ez a hash függvény adott karaktersorozatra mindig ugyanazt a kódot generálja. Első bejelentkezéskor a megadott jelszó hash kódját az alkalmazás berakja a **userinfo** táblába. Így a tábla lekérdezésekor nem tudódhat ki az eredeti karaktersorozat. Az n -ik ($n > 1$) bejelentkezéskor a megadott jelszó hash kódját összehasonlítja az első alkalommal az adatbázisba beadott hash kóddal. Ha egyezik, az alkalmazás továbblép. Első futtatáskor (tehát még a **userinfo** tábla nem tartalmaz jelszavakat) ez az ablak jön be.

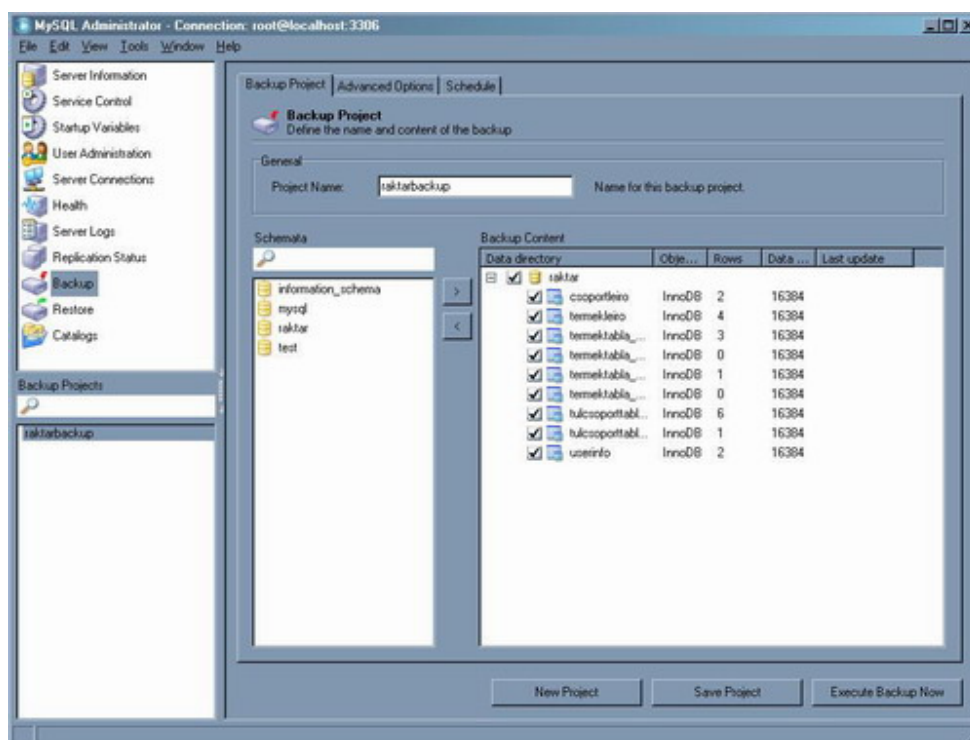


Rögtön az **OK** gomb lenyomása után jön be a jelszóbekérő ablak.

A dialog box titled "Jelszavak" (Passwords). It has a blue header bar. Below the title, the text "Administrátor jelszavának megváltoztatása..." (Change administrator password...) is displayed. There are three input fields: "Régi jelszó:" (Old password:), "Új jelszó:" (New password:), and "Új jelszó megegyszer:" (New password again:). Each field is followed by a text input box. At the bottom, there are two buttons: "Rendben" (OK) and "Mégsem" (Cancel).

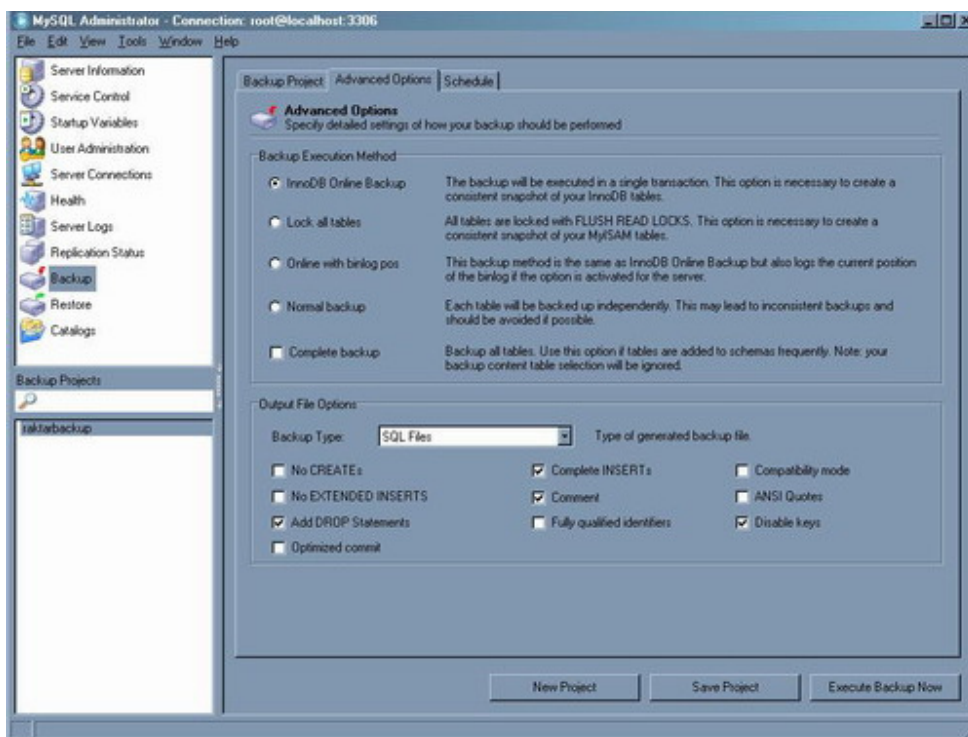
9.2 Az adatbázis mentése: backup

Az adatbázisban tárolható adatokról érdemes bizonyos időközönként másolatot készíteni, hogy a rendszer sérülésekor ne vesszen el minden információ. Az adatvesztéssel járó sérülés után a másolatokból visszaállítható az adatbázis egy korábbi állapota. A másolatok készítésének gyakoriságáról a rendszert be lehet állítani. A rendszer kezelője maga dönthet a gyakoriság beállításáról, de ajánlatos a heti legalább egy adatmentés.

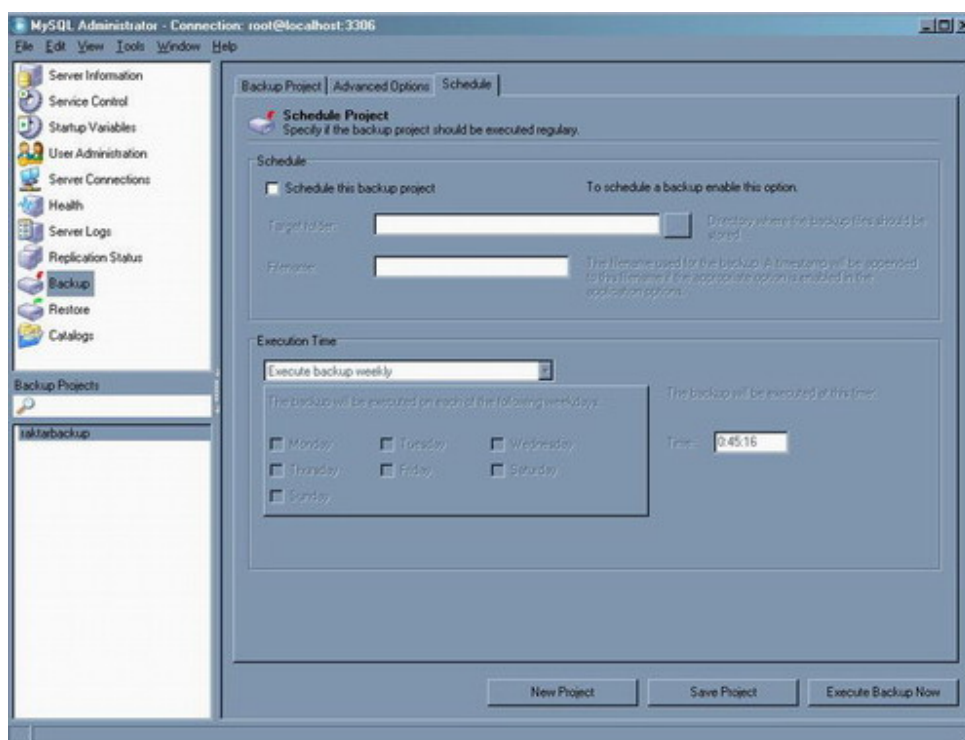


Ebben az ablakban kell elkezdni a biztonsági mentés beállítását. **New Project** gombra kattintva megadjuk a projekt nevét, kiválasztjuk az adatbázist és azokat a táblákat amelyekről biztonsági mentést akarunk csinálni.

Az **Advanced Options** fülre kattintva további pontosításokat végezhetünk. Itt a biztonsági mentés beállításain lehet finomítani. Javaslom a fenti részben kiválasztani a **complete backup**-ot mivel ennek az alkalmazásnak az a lényege, hogy a táblák száma dinamikusan nő.



A **Schedule** fülre kattintva lehet beállítani, hogy a MySQL milyen időközönként végezze automatikusan a biztonsági mentést.



10 Összefoglalás

A szoftver alkalmas lett egy akármilyen raktár nyilvántartó feladatait ellátni. Legyen az egy kicsi horgászbolt árukészlet nyilvántartása vagy egy autógyár alkatrész leltára. Továbbá az egyszerű leltárprogramokon túlmutatva, tud szellemi javakat is tárolni, nem csak anyagiakat. A rendszer adminisztrátora meg tudja határozni, hogy milyen tulajdonságait kívánja felvenni a terméknek. Szabad kezét ad a termék architektúrájának kialakításában. Rugalmas és normalizált. Tud már letárolt táblákat újrafelhasználni, így csökkenteni tudja a redundanciát. Sikerült az egész szoftvert ingyenessé tenni a felhasznált alkalmazások ingyenessége miatt. Sikerült a szoftvert nagyon egyszerűen felinstallálhatóvá tenni. Akár egyetlen sor begépelése nélkül fel lehet installálni és használni. Természetesen a termékek adatbázisbeli képének a megtervezéséhez szükséges bizonyos fokú absztrahálási képesség. Viszont mivel ez a National Instruments IT részlegének készült így fel sem merült bennem, hogy ez valamilyen problémát is okozhatna az alkalmazás adminisztrátorának. Sőt az informatikus még örömet is lelt abban, hogy végre szabad kezét kaphat a kreativitása kiéléséhez azzal, hogy ő tervezi meg a terméket.

A program rendelkezik bizonyos szintű biztonságtechnikával. Ez jelszóvédelmet és adatbázis-mentést jelent ebben az alkalmazásban. A jelszavak kódolásához az MD5 kriptográfiai technológiát használom. Az adatbázis mentést a MySQL biztosítja. Egyszerűen bekattintható, hogy milyen időközönként végezze az adatbázis mentést. Kiválasztható, hogy milyen táblákat tartunk fontosnak, hogy melyeket mentse le egy biztonsági mentésben.

Természetesen az alkalmazás közel sem teljes. Nagyon sok fejleszténivaló lenne rajta, hogy megállja a helyét egy komoly telephelyen. Egy raktárban több **felhasználó** dolgozhat, valamint a főnöknek is kell legyen adminisztrátori jelszava. Le kellene tárolnia az alkalmazásnak, hogy mikor ki lépett be, a nagyobb biztonság érdekében. A nagyobb eladhatóság érdekében jó lenne egy ízléses dizájn. Valamint a felhasználóknak egy aprólékos Help. Az alkalmazás ezek nélkül is használható.

Úgy gondolom, hogy ezek nélkül is sikerült bemutatnom a 4GL-es nyelvek előnyeit, és azt a képességét, hogy egy igazán flexibilis, alkalmazás is készíthető **drag and drop** technikával.

11 Továbbfejlesztési lehetőségek

- Egyszerű továbbfejlesztési lehetőség annak a megoldása, hogy több **Felhasználó** és esetleg több **Adminisztrátor** legyen.
- Különböző statisztikák készítése.
- Design javítása
- Egy aprólékos Help elkészítése
- Új keresési algoritmusok készítése
- Connection Management, gyorsabb adatáramlás

12 Irodalomjegyzék

- **Gábor András, Juhász István:** PL.\SQL-programozás
- **Dr. Tilly Károly:** 4GL-eszközök: <http://home.mit.bme.hu/~tilly/pages/4gl.html>
- **Novák István, Borbély András, Holpár Péter:** Fejlesztői Infrastruktúra - .NET
- **Bill Wagner:** Hatékony C#
- **MySql 5.0 Reference Manual :** <http://dev.mysql.com/doc/refman/5.0/en/>
- **Jeffrey D. Ullman-Jennifer Widom:** Adatbázisrendszerek. Alapvetés
- **Hector Garcia-Molina-Jeffrey D. Ullman-Jennifer Widom:** Adatbázisrendszerek megvalósítása