**Fontosabb MS-ACCESS 97 SQL parancsok**

A példákhoz kapcsolódó adatbázis megtalálható az F:\OKTINF\ADATB\NWIND alkönyvtárban.   
1.1 Tábla létrehozása: CREATE TABLE   
Szintaxis:   
CREATE TABLE tábla\_neve   
(mező1 típus [(méret)] [NOT NULL] [ CONSTRAINT…],   
mező2 típus [(méret)] [NOT NULL] [ CONSTRAINT…],…);

A CREATE TABLE utasítás részei:   
tábla - A létrehozandó tábla neve   
mező1, mező2 - Az új tábla mezőjének vagy mezőinek neve. Legalább egy mezőt létre kell hoznunk.   
típus - Az új tábla mezőjének adattípusa   
méret - A mező mérete karakterben (csak Szöveg és Bináris mezőnél)   
NOT NULL - Kötelező adatbevitel   
CONSTRAINT … - Megkötések egy mezőre

Példák:   
A következő példa létrehoz egy új táblát (PróbaTábla1 névvel) két Szöveg típusú mezővel:   
CREATE TABLE [Próba Tábla1]    
(Vezetéknév TEXT, Keresztnév TEXT);

(Ha az azonosítóban szóköz van, akkor azt szögletes zárójelbe kell tenni!!!)

A következő példa létrehoz egy új táblát két Szöveg típusú és egy Dátum/Idő típusú mezővel, továbbá egy egyedi indexszel, amely mindhárom mezőt tartalmazza:   
CREATE TABLE [Próba Tábla2]    
(Vezetéknév TEXT,Keresztnév TEXT,[Születési dátum] DATETIME,   
CONSTRAINT TöbbmezősIndexNeve UNIQUE (Vezetéknév, Keresztnév, [Születési dátum]));

A következő példa létrehoz egy új táblát két Szöveg típusú és egy Egész típusú mezővel. Az SSN mező az elsődleges kulcs.   
CREATE TABLE [Próba Tábla3]   
(SSN INTEGER CONSTRAINT ElsődlegesKulcsNeve PRIMARY KEY,   
Vezetéknév TEXT, Keresztnév TEXT);

A következő példa létrehoz egy új táblát elsődleges- és idegenkulccsal, és mindenféle csingilingivel:   
CREATE TABLE [Próba Tábla4]   
(Rendeléskód INTEGER CONSTRAINT ElsődlegesKulcsNeve PRIMARY KEY,   
Címzett TEXT NOT NULL,    
Irányítószám TEXT CONSTRAINT EgyediIndexNeve UNIQUE,   
Vevőkód TEXT (5) CONSTRAINT KülsőKulcsNeve REFERENCES Vevők);

1.2 Indexek létrehozása – CREATE INDEX   
Új index létrehozása a már létező táblán   
Szintaxis:   
CREATE [ UNIQUE ] INDEX index\_neve   
ON tábla    
(mező1 [ASC|DESC],   
mező2 [ASC|DESC],…[WITH…]);

A CREATE INDEX utasítás részei:   
UNIQUE – Egyedi index (ismétlődések nem fordulhatnak elő a mezőben)   
index - A létrehozandó index neve   
tábla - A tábla neve, amelyet indexelni akarunk.   
mező - A indexelendő mező(k) neve(i). Egymezős index esetén írjuk a mező nevét zárójelben a tábla neve után. Többmezős index esetén soroljuk fel azokat a mezőket, amelyekre az index vonatkozik. Csökkenő sorrend létrehozásához használjuk a DESC foglalt szót; egyébként a sorrend növekvő (ASC).   
WITH – Ezzel a záradékkal az adatok érvényességi szabályait határozhatjuk meg (megadása nem kötelező):   
A DISALLOW NULL beállítással tilthatjuk a Null értékű bejegyzéseket az új rekordok mezőibe.   
Az IGNORE NULL beállítás esetén a Null értékű mezővel rendelkező rekordok nem fognak az indexben szerepelni.   
A PRIMARY beállítással az indexelt mező(ke)t elsődleges kulcsként jelöljük meg. Ez magába foglalja a kulcs egyediségét is, ezért nem kell a UNIQUE foglalt szót használnunk.

Példák:   
A következő példa létrehoz az Alkalmazottak táblához egy új indexet, mely az Otthoni telefon és a Mellék mezőből áll.   
CREATE INDEX ÚjIndexNeve ON Alkalmazottak (Otthoni telefon, Mellék);

A következő példa létrehoz a Vevők táblához egy új indexet a Vevőkód mező alapján. Nem létezhet két rekord ugyanazzal a Vevőkód értékkel, és nincs engedélyezve a Null érték.   
CREATE UNIQUE INDEX CustID ON Vevők (Vevőkód) WITH DISALLOW NULL;

1.3 Indexek törlése – DROP INDEX

Szintaxis:   
DROP INDEX index\_név ON táblanév;

Példa:   
Az előző példában létrehozott indexet a DROP utasítás segítségével lehet törölni. A tábla Tervező nézetében az index kizárólag akkor törölhető, ha a Vevők táblához tartozó kapcsolatot is töröljük. A következő kódrészlet a DROP utasítás segítségével törli az új indexet.

DROP INDEX CustID ON Vevők;

1.4 Adatbevitel táblákba – INSERT INTO   
Egy vagy több rekordot ad hozzá a táblához. Ezt hozzáfűző lekérdezésnek hívják.

Szintaxis:   
Több rekordot létrehozó hozzáfűző lekérdezés:   
INSERT INTO cél\_tábla [(mező1, mező2, ...)]   
SELECT [mező1, mező2, ...]   
FROM forrás\_tábla;

Egyetlen rekordot létesítő hozzáfűző lekérdezés:   
INSERT INTO cél\_tábla [(mező1, mező2, ...)]   
VALUES (érték1, érték2, ...);

Az INSERT INTO utasítás a következő részekből áll:   
Cél\_tábla - A tábla neve, melyhez hozzá szeretnénk fűzni az új rekordokat.   
Forrás\_tábla - A tábla vagy lekérdezés neve, melyből a rekordokat át szeretnénk másolni.   
mező1, mező2 - Ha cél argumentum után szerepelnek, az új rekordban kitöltendő mezők nevei, ha forrás argumentum után szerepelnek, akkor ezekből a mezőkből veendők az adatok.   
érték1, érték2 - Az új rekord egyes mezőibe írandó értékek. Minden érték a vele a listában azonos pozíción szereplő mezőbe kerül: az érték1 a mező1-be, az érték2 a mező2-be, és így tovább. Az értékeket vesszővel választjuk el és a szöveges értékeket idézőjelbe (" ") tesszük.

Példák:   
Ha ki szeretnénk próbálni a következő példákat, akkor hozzunk létre egy másolatot a Vevők tábláról Vevők2 névvel (csak a struktúrát másolja le).    
A példa kijelöli a Vevők tábla valamennyi rekordját, és hozzáadja a Vevők2 táblához. Ha nincsenek külön megjelölve az oszlopnevek, akkor a SELECT és az INSERT INTO utasításban szereplő oszlopneveknek meg kell egyezniük.   
INSERT INTO Vevők2 SELECT \* FROM [Vevők];

A következő példa létrehoz egy új rekordot az Alkalmazottak táblában:   
INSERT INTO Alkalmazottak (Keresztnév, Vezetéknév, Beosztás)    
VALUES ('Pista', 'Budai', 'Gyakornok');

A következő példa egy feltételezett Tanulók táblából kiválaszt minden olyan gyakornokot, aki több, mint 30 napja van a vállalatnál, és az őket tartalmazó rekordokat hozzáadja az Alkalmazottak táblához.   
INSERT INTO Alkalmazottak SELECT Gyakornok.\* FROM Tanulók    
WHERE [Belépés dátuma] < Now() - 30;

1.5 Adatok módosítása - UPDATE   
Frissítő lekérdezést hoz létre, amely a megadott tábla mezőit módosítja a megadott feltétel alapján.

Szintaxis:   
UPDATE tábla   
SET új érték   
WHERE feltétel;

Az UPDATE utasítás a következő részekből áll:   
Tábla - Azon tábla neve, amelynek adatait módosítani szeretnénk.   
új érték - Az a kifejezés, amely meghatározza, milyen érték kerül be a módosítandó rekordok egyes mezőibe.   
Feltétel - Az a kifejezés, amely meghatározza, mely rekordok módosulnak. Csak a kifejezésnek eleget tevő rekordok változnak.   
Megjegyzés:   
Az UPDATE különösen akkor hasznos, amikor sok rekordot szeretnénk módosítani, vagy amikor a módosítandó rekordok több különböző táblában vannak.   
Egyszerre több mezőt is módosíthatunk.    
Az UPDATE nem hoz létre eredményhalmazt. Ha a rekordokat frissítő lekérdezéssel módosítjuk, a műveletet nem lehet visszavonni. Ha látni szeretnénk, mely rekordok módosulnának, először futtassunk le egy választó lekérdezést ugyanolyan feltétellel, s ennek eredményét vizsgáljuk meg. Csak ezután futtassuk le a módosító lekérdezést.   
Az adatokról mindig legyen másolatunk. Ha rossz rekordokat módosítunk, a másolatból helyreállíthatjuk az eredeti adatokat.

Példák:   
A következő példa minden olyan rekord Felettes mezőjét 5-re állítja, amelynek jelenleg 2 az értéke.   
UPDATE Alkalmazottak SET Felettes = 5 WHERE Felettes = 2;

A következő példa minden olyan termék egységárát megnöveli 10 százalékkal, amely a 8. számú szállítótól származik, és amelyből van raktáron.   
UPDATE Termékek SET Egységár = Egységár \* 1.1   
WHERE Szállítókód = 8 AND Raktáron>0;

A következő példa minden olyan termék egységárát csökkenti 5 százalékkal, amely a Tokyo Traders nevű szállítótól származik, és amelyből van raktáron. A Termékek és a Szállítók tábla között több-az-egyhez kapcsolat van.   
UPDATE Szállítók INNER JOIN Termékek   
ON Szállítók.Szállítókód = Termékek.Szállítókód SET Egységár = Egységár \* .95   
WHERE Cégnév = 'Tokyo Traders' AND Raktáron>0;

1.6 Adatok törlése - DELETE   
Létrehoz egy törlő lekérdezést, amely a FROM záradékban felsorolt táblákból törli azokat a rekordokat, amelyek eleget tesznek a WHERE záradék feltételének.

Szintaxis:   
DELETE \* FROM tábla   
WHERE feltétel

A DELETE utasítás két részből áll:   
Tábla - A tábla neve, amelyből a rekordokat törölni kell   
Feltétel - A törlendő rekordokat kijelölő kifejezés

Megjegyzés   
A DELETE különösen hasznos, ha sok rekordot szeretnénk törölni   
Egész táblát a DROP TABLE utasítással töröljünk. Ha az egész táblát töröljük, akkor annak szerkezete is elvész. Ezzel szemben a DELETE használatakor csak az adatok törlődnek, a tábla szerkezete és minden jellemzője (például a mezőnevek, az indexek) érintetlen marad.   
A DELETE segítségével olyan táblákból is törölhetünk rekordokat, amelyek más táblákkal egy-a-többhöz kapcsolatban állnak. Kaszkádolt törlésnél a kapcsolat "több" oldalán levő rekordok is törlődnek, ha a kapcsolat "egy" oldalán levő rekordot törli a lekérdezés. Például a Vevők és a Rendelések tábla esetében a Vevők vannak az "egy" oldalon, a Rendelések a "több" oldalon. Kaszkádolt törlés használatakor a Vevők táblából törölt rekordokkal együtt a hozzájuk tartozó Rendelések rekordok is törlődnek.   
A törlő lekérdezés nem egyes mezők adatait törli, hanem a teljes rekordokat. Ha egyes mezők adatait szeretnénk törölni, használjunk frissítő lekérdezést, ami a mezők értékét Null értékre változtatja.   
A törlő lekérdezés nem vonható vissza, a törölt rekordok elvesznek. Ha látni szeretnénk a törlendő rekordokat, akkor a törlés előtt először vizsgáljuk meg egy ugyanolyan feltételt használó választó lekérdezés eredményét.   
Mindig legyen biztonsági mentés az adatokról. Így ha rossz rekordokat töröltünk, ezek visszanyerhetők.

Példa:   
Ha a következő példát ki szeretnénk próbálni, akkor adjunk néhány rekordot az Alkalmazottak táblához, és a Beosztás mezőbe írjuk be a “Gyakornok” értéket.   
A következő példa töröl az Alkalmazottak táblából minden olyan rekordot, amelyben a Beosztás mező értéke “Gyakornok”.    
DELETE \* FROM Alkalmazottak WHERE Beosztás = 'Gyakornok';

1.7 Adatok lekérdezése - SELECT   
A SELECT utasítás hatására a Microsoft adatbázis-motor rekordhalmaz formátumú adatokat ad az adatbázisról.

Szintaxis:   
SELECT [tábla.]mező1 [AS alias1], [tábla.]mező2 [AS alias2] …   
FROM tábla1, tábla2, ...   
[WHERE... ]   
[GROUP BY... ]   
[HAVING... ]   
[ORDER BY... ]

A SELECT utasítás részei:

Tábla - A tábla neve, amely a kiválasztandó rekordok mezőit tartalmazza.   
mező1, mező2 - A mező neve, amelyben az adatokat keressük. Ha több mezőt jelölünk meg, azok a megjelölt sorrendben lesznek visszakeresve.   
alias1, alias2 - A tábla eredeti oszlopnevei helyett használandó oszlop fejlécek.   
tábla1, tábla2 - A tábla vagy táblák neve, amelyekből az adatokat vissza szeretnénk keresni.

Megjegyzés:   
Az utasítás végrehajtásakor a Microsoft Jet adatbázis-motor megkeresi a meghatározott tábla vagy táblák megadott oszlopait, kiválasztja a feltételeknek megfelelő sorokat, majd sorba rakja és csoportosítja azokat a megadott sorrendnek megfelelően.   
A SELECT utasítás nem változtatja meg az adatbázis adatait.

1.7.1 Legegyszerűbb alak:   
A csillag (\*) használatával a tábla minden mezőjét kiválasztjuk. A következő utasítás az Alkalmazottak tábla minden mezőjét kijelöli:

SELECT \* FROM Alkalmazottak;   
1.7.2 Az ismétlődő sorok elnyomása (DISTINCT)   
Ha azt szeretnénk megtudni, hogy milyen országból származnak vevőink, akkor az ismétlődő sorokat el kell nyomni, hiszen egy adott országból több vevő is származhat.

SELECT DISTINCT Ország FROM Vevők;

1.7.3 A WHERE záradék használata:   
Meghatározza, hogy a FROM záradékban felsorolt táblák mely rekordjait érintik a SELECT, UPDATE vagy DELETE utasítások.

Példák:   
A következő példa minden olyan rekord Vezetéknév és Keresztnév mezőjét kijelöli, melyben a vezetéknév “King”.   
SELECT Vezetéknév, Keresztnév FROM Alkalmazottak   
WHERE Vezetéknév = 'King';

A következő példa az Alkalmazottak táblában minden olyan rekord Vezetéknév és Keresztnév mezőjét kijelöli, amelyben a vezetéknév S betűvel kezdődik.   
SELECT Vezetéknév, Keresztnév FROM Alkalmazottak   
WHERE Vezetéknév Like 'S\*';

A következő példa kijelöli mindazokat a rekordokat, melyeknek egységára 2000 és 5000 forint között van.   
SELECT Terméknév, Egységár FROM Termékek   
WHERE (Egységár >=2000.00 And Egységár <= 5000.00);

A következő példa minden olyan terméket kijelöl, amelynek neve betűrend szerint “Cha” és “Out” közé esik (vagy ezekkel megegyezik). Az “Outback Lager” név nem esik ebbe a tartományba, mert az “Out” után van.   
SELECT Terméknév, Egységár FROM Termékek   
WHERE Terméknév Between 'Cha' And 'Out';

A következő példa kijelöli azokat a rendeléseket, melyek 1995 első felében születtek.   
SELECT Rendeléskód, [Rendelés dátuma] FROM Rendelések   
WHERE [Rendelés dátuma] Between #1-1-95# And #6-30-95#;

A következő példa kijelöli azokat a rendeléseket, melyek az Egyesült Államok Idaho, Oregon vagy Washington államába szóltak.   
SELECT Rendeléskód, Körzet FROM Rendelések   
WHERE Körzet In ('ID', 'OR', 'WA');

1.7.4 Táblák összekapcsolása – INNER JOIN   
Egyesíti két tábla rekordjait, ha azok közös mezőjében azonos érték szerepel.

Szintaxis:   
FROM tábla1 INNER JOIN tábla2 ON tábla1.mező1 hasonlító\_művelet tábla2.mező2

Az INNER JOIN művelet a következő részekből áll:   
tábla1, tábla2 - A két tábla neve, melyek rekordjaiból az egyesített rekordok készülnek.   
mező1, mező2 - Az egyesítendő mezők neve. Ha a mezők nem szám típusúak, ugyanolyan adattípushoz kell tartozniuk, ugyanolyan jellegű adatokat kell tartalmazniuk, de a nevük lehet különböző.   
Hasonlító\_művelet - Bármely összehasonlító művelet: "=", "<", ">", "<=", ">=" vagy "<>".

Megjegyzés:   
Az INNER JOIN művelet bármely FROM záradékban használható. Ez az illesztés leggyakoribb fajtája. Egyesíti két tábla rekordjait, ha azok közös mezőjében azonos érték szerepel.   
Az INNER JOIN művelet segítségével kiválogathatjuk a Részlegek és az Alkalmazottak táblából minden részleg összes alkalmazottját. Ezzel szemben, ha az összes részleget szeretnénk kiválogatni (akkor is, ha némelyikhez nincs alkalmazott rendelve), vagy az összes alkalmazottat (akkor is, ha egyesek nem tartoznak egyik részleghez sem), a LEFT JOIN vagy RIGHT JOIN művelettel létesíthetünk laza illesztést.

Példa:   
A következő példa azt mutatja, hogyan illeszthető a Kategóriák és a Termékek tábla a Kategóriakód mező segítségével:   
SELECT Kategórianév, Terméknév   
FROM Kategóriák INNER JOIN Termékek   
ON Kategóriák.Kategóriakód = Termékek.Kategóriakód;

A példában a Kategóriakód az illesztett mező, de a lekérdezés eredményében nem jelenik meg, hiszen nem szerepel a SELECT utasításrészben. Ha az illesztett mezőt is meg szeretnénk jeleníteni, be kell azt (a fenti példában a Kategóriák.Kategóriakód mezőt) írni a SELECT után.

1.7.5 Eredménysorok csoportosítása (GROUP BY záradék)   
A megadott mezőben az azonos értékű rekordokat egy rekorddá alakítja át. Összesítő érték hozható létre minden rekordról, ha SQL összesítő függvényt, mint például a Sum vagy a Count függvényt, írunk be a SELECT utasításba.

Példák:   
A következő példa szállítónként kiszámítja a termékek átlagos egységárát, és az árlistát szállítónként csoportosítja.   
SELECT Szállítókód, Avg(Egységár) AS [Átlagos egységár]   
FROM Termékek GROUP BY Szállítókód;

A következő példa kategóriánként meghatározza a legmagasabb egységárú terméket.   
SELECT Kategóriakód, Max(Egységár) AS [Legnagyobb egységár]   
FROM Termékek GROUP BY Kategóriakód;

A következő példa megszámolja, hány rendelés tartozik az egyes alkalmazottakhoz a Rendelések táblában.   
SELECT Alkalmazottkód, Count(Rendeléskód)AS [Rendelések száma]    
FROM Rendelések GROUP BY Alkalmazottkód;

1.7.6 Csoportosított eredmények szűrése (HAVING záradék)   
A záradékban megadhatjuk, hogy a SELECT utasítás GROUP BY záradékával csoportosított rekordok közül melyek legyenek megjelenítve. Miután a GROUP BY csoportosítottuk a rekordokat, a HAVING záradékban megadott feltételeknek megfelelő összes rekord megjelenik a képernyőn.

Példák:   
A következő példa kikeresi azokat a szállítókat, akik által szállított termékek átlagos egységára 2500 forint fölött van.   
SELECT Szállítókód, Avg(Egységár) AS [Átlagos egységár]   
FROM Termékek GROUP BY Szállítókód   
HAVING (Avg(Egységár)>2500);

A következő példa kijelöli azokat az alkalmazottakat, akik több, mint 100 rendelést bonyolítottak le.   
SELECT Alkalmazottkód, Count(Rendeléskód) AS [Rendelések száma]   
FROM Rendelések GROUP BY Alkalmazottkód   
HAVING Count(Rendeléskód) > 100;

1.7.7 Eredménysorok rendezése (ORDER BY záradék)   
A lekérdezés eredményeként kapott rekordok megadott mezőit növekvő vagy csökkenő sorrendbe rendezi.

Példák:   
Az alábbi SQL utasítások az ORDER BY záradék segítségével rendezik a rekordokat betűrendbe, illetve kategória szerint.

A következő példa a rekordokat vezetéknév szerint rendezi visszafelé betűrendbe    
(Z-től A-ig).   
SELECT Vezetéknév, Keresztnév FROM Alkalmazottak ORDER BY Vezetéknév DESC;

A következő példa elsősorban a kategóriakód, másodsorban a terméknév szerint rendezi a rekordokat.   
SELECT Kategóriakód, Terméknév, Egységár FROM Termékek   
ORDER BY Kategóriakód, Terméknév