

3.előadás: Adatbázisok-I.

dr. Hajas Csilla (ELTE IK)

<https://sila.hajas.elte.hu>

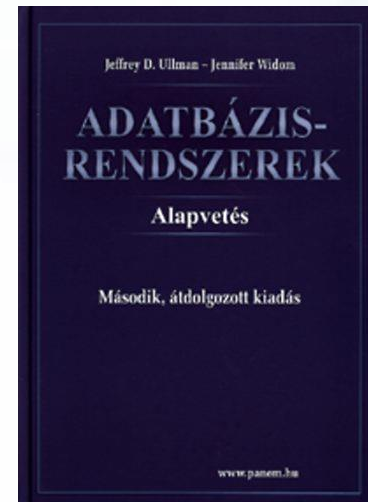
Adatbázis-kezelő rendszerek felépítése SQL gyakorlat (a táblák előkészítése)

1.fejezet: Az adatbázisrendszerek világa

2.3. Relációsémák definiálása SQL-ben

(Táblák létrehozása és előkészítése,

folyt.később 6.ea részletek: 7.fejezet)



Mit várunk egy ABKR-től?

- Adatbázis: Adatok együttese, amelyet az adatbázis-kezelő rendszer kezel. **Mit várunk az ABKR-től? (DBMS-től?)**
 - Támogassa nagy méretű (több terabyte mennyiségű) adat hosszú időn keresztül való tárolását, és tegye lehetővé a hatékony hozzáférést a lekérdezések és adatbázis-módosítások számára.
 - Tegye lehetővé a felhasználók számára, hogy új adatbázisokat hozhassanak létre, és azok sémáját, vagyis az adatok logikai struktúráját egy speciális nyelven adhassák meg: **Adatdefiníciós nyelv (DDL)**
 - Tegye lehetővé a felhasználóknak, hogy az adatokat egy megfelelő nyelv segítségével lekérdezhessék vagy módosíthassák: **Adatkezelő nyelv (DML)**

(1) Adatmodell támogatása

- Az adatmodell a valóság fogalmainak, kapcsolatainak, tevékenységeinek magasabb szintű ábrázolása
 - File-kezelés indexekkel együtt, ezt váltotta fel a
 - CODASYL szabvány, hálós adatmodell (hatékony keresés)
 - Hierarchikus adatmodell (apa-fiú kapcsolatok gráfja)
 - Ted Codd - Relációs adatmodell (táblák rendszere, könnyen megfogalmazható műveletek)
 - Objektum-orientált adatmodell (az adatbázis-kezelés funkcionalitásainak biztosítása érdekében gyakran relációs adatmodellre épül), + Objektum-relációs adatmodell
 - Logikai adatmodell (szakértői rendszerek, tények és következtetési szabályok rendszere)
 - Dokumentumok - Félig strukturált adatmodell, az XML (szabvány adatcsereformaként jelent meg), XML, JSON
 - Gráf adatbázisok, NoSQL

(2) Adatbázis-kezelő nyelvek támogatása

- **SQL** – relációs (és objektum-relációs) adatbázis-kezelő szabvány nyelve, és erről lesz a mai előadáson áttekintés.

Főbb részei:

- **DDL** (Data Definition Language) adatdefiniáló (sémaleíró) nyelv: sémák, adatstruktúrák megadása, objektumok létrehozása, módosítása, törlése: CREATE, ALTER, DROP
- **DML** (Data Manipulation Lang.) adatkezelő és lekérdező nyelv: INSERT, DELETE, UPDATE és SELECT
- **DCL** (Data Control Lang.) adatvezérlő nyelv, jogosultságok kiosztása és visszavonása: GRANT, REVOKE
- **Tranzakció-kezelés**: COMMIT, ROLLBACK
- **Procedurális kiterjesztések**: SQL/PSM, Oracle PL/SQL

(3) Több felhasználó támogatása

- Felhasználói csoportok. Kulcsemberek:
 - DBA adatbázis-rendszergazda
 - felügyeli az adatbázis-példányokat és adatbázis-szervereket
 - felépíti a rendszert, implementálja és optimális adatbázis-megoldást biztosít
 - Adatbázis-tervező (sématervezés)
 - Alkalmazás-fejlesztő, programozó (kódolás)
 - Felhasználók (akik használják a rendszert)

(4) Adatvédelem, adatbiztonság

- **Jogosultságok** (objektumok olvasása, írása, módosítása, készítése, törlése, jogok továbbadása, visszavonása) GRANT és REVOKE
- Jogosultságok tárolása rendszertáblákban történik
- **Jogosultságok kezelése**, felhasználók, jelszavak, hozzáférési jogok
- Adatbázissémák korlátozása (virtuális) nézettáblák segítségével
- Tárolt adatok, hálózati adatforgalmak titkosítása

Adatbázis-kezelők részei

➤ Lekérdezés-feldolgozó

- Lekérdezés szintaktikai ellenőrzése
- Adatbázis-objektumok létezésének, és a hozzáférési jogoknak az ellenőrzése (metaadatbázis, rendszertáblák)
- Lekérdezés optimális átfogalmazása
- Végrehajtási tervek készítése
- Az adatstruktúrák, méretek statisztikái alapján várhatóan minimális költségű végrehajtási terv kiválasztása
- Az optimális végrehajtási terv lefuttatása

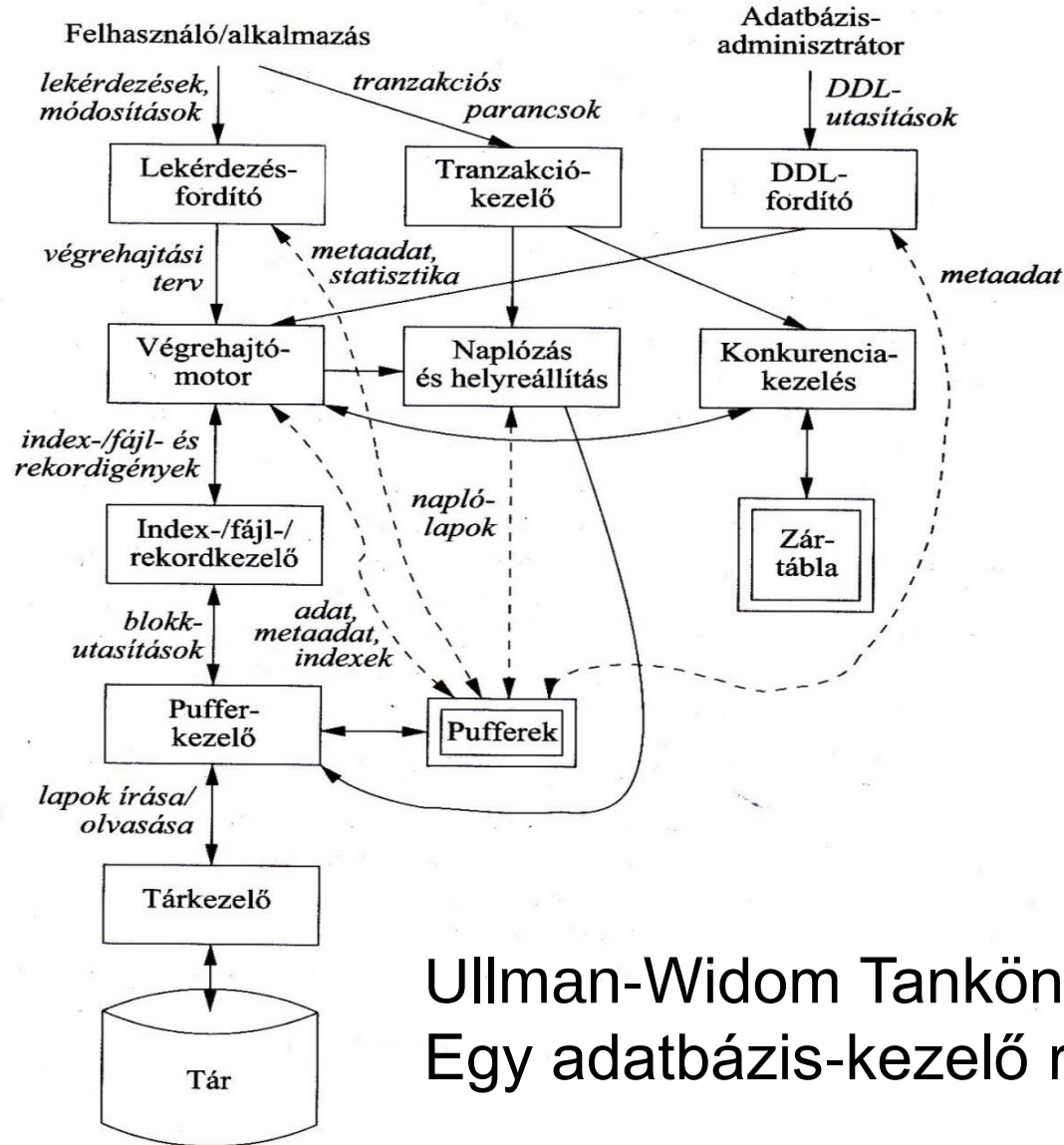
➤ Tranzakció-kezelő:

- Tranzakciók párhuzamos és biztonságos végrehajtásának a tranzakciók ACID tulajdonságainak biztosítása

➤ Tárkezelő és pufferkezelő

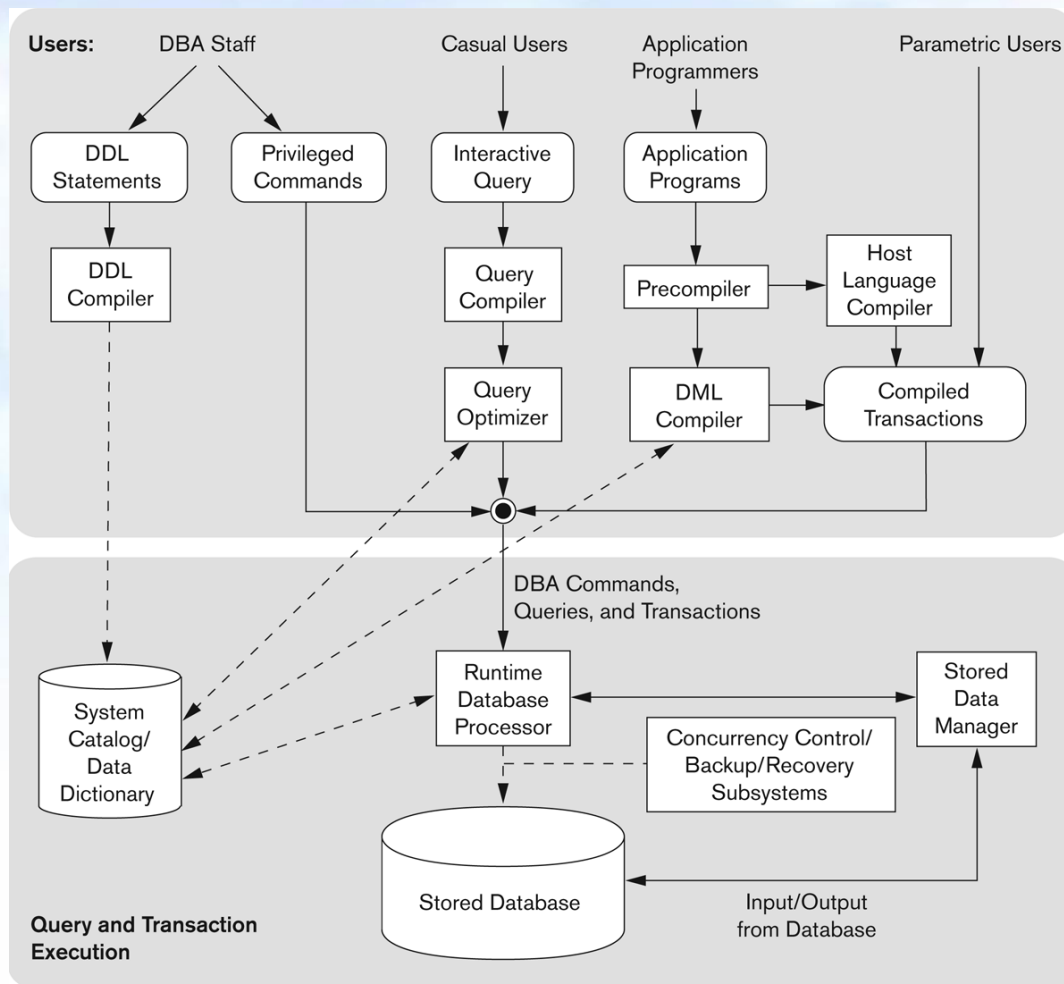
- fizikai adatstruktúrák, táblák, indexek, pufferek kezelése

Adatbázis-kezelő rendszer felépítése



Ullman-Widom Tankönyv 1.1. ábra
Egy adatbázis-kezelő rendszer részei

Adatbázis-kezelő rendszer felépítése



Forrás:

Elmasri-Navathe: Fundamentals of Database Systems

Figure 2.3

Component modules of a DBMS and their interactions.

SQL története, szabványok

- Szabvány adatbázis-kezelő nyelv: SQL
- SQL (angol kiejtésben SEQUEL) uis az SQL előfutára IBM fejlesztette ki a 70-es években: SEQUEL → SQL más is volt pl. Ingres : QUEL (ez kalkulus alapú lekérdezés)
- Szabványok (ANSI, ISO)
SQL86, SQL89, SQL92 (SQL2), **SQL:1999** (SQL3),
SQL: 2003, SQL:2006, SQL:2008
- Nyelvjárások (**Oracle**, Sybase, DB2, Progress, MSSQL, mySQL, SQL Server, PostgreSQL, Access,...)
- Az SQL megvalósítások között vannak különbségek, gyakorlatokon az **Oracle SQL**-t nézzük meg részletesen.

SQL fő komponensei

- **Az SQL elsődlegesen lekérdező nyelv** (Query Language)
SELECT utasítás (az adatbázisból információhoz jussunk)
- **Adatkezelő nyelv, DML** (Data Manipulation Language)
INSERT, UPDATE, DELETE, SELECT
- **Sémaleíró nyelv, DDL** (Data Definition Language)
CREATE, ALTER, DROP
- **Adatvezérlő nyelv, DCL** (Data Control Language)
GRANT, REVOKE
- **Tranzakció-kezelés**
COMMIT, ROLLBACK, SAVEPOINT
- **Procedurális kiterjesztések**
SQL/PSM és a gyakorlatban Oracle PL/SQL

SQL DDL

Adatbázis relációsémák definiálása

- Az SQL tartalmaz **adateleíró részt (DDL)**, az adatbázis **objektumainak** a leírására és megváltoztatására. **Objektumok** leíró parancsa a **CREATE** utasítás.
- A relációt az SQL-ben táblának (TABLE) nevezik, az SQL alapvetően háromféle táblát kezel:
 - Alaptáblák (permanens) CREATE TABLE
 - Nézet táblák CREATE VIEW
 - Átmeneti munkatáblák (WITH utasítás)
- **Alaptáblák** megadása: **CREATE TABLE**

Tábla/reláció sémák SQL-ben

- A legegyszerűbb formája:

CREATE TABLE relációnév (
Attribútum deklarációk listája,
Kiegészítő lehetőségek
);

- Az attribútum deklaráció legalapvetőbb elemei:
Attribútumnév típus [kiegészítő lehetőségek]
-- itt: a **típus** olyan, amit az SQL konkrét megvalósítása támogat (gyakorlaton Oracle környezetben nézzük meg),
Típusok, pl: INTEGER, REAL, CHAR, VARCHAR, DATE
-- **Kiegészítő lehetőségek** például [DEFAULT], [UNIQUE], [PRIMARY KEY], [FOREIGN KEY, REFERENCES], stb.

Egyszerű példák táblák létrehozására

```
CREATE TABLE Sörözők (  
    név CHAR(20) ,  
    város VARCHAR2(40) ,  
    tulaj CHAR(30) ,  
    engedély DATE DEFAULT SYSDATE  
);
```

```
CREATE TABLE Felszolgál (  
    söröző CHAR(20) ,  
    sör VARCHAR2(20) ,  
    ár NUMBER(10,2) DEFAULT 100  
);
```

Az SQL értékekről (bővebben gyakorlaton)

- INTEGER, REAL, stb, a szokásos értékek, számok.
- STRING szintén, de itt egyes-aposztróf közé kell tenni a 'szöveget' (vagyis nem „macskaköröm” közé).
Két egyes-aposztróf = egynek felel meg, például
' **Joe** ' ' **s Bar** ' megfelel a Joe's Bar szövegnek.
- DATE és TIME típusok is vannak az SQL-ben.
- A dátum formátumát meg kell adni DATE 'yyyy-mm-dd'
Például: DATE '2007-09-30' (2007. szept. 30)
- Az idő formátumát is meg kell adni TIME 'hh:mm:ss'
Például: TIME '15:30:02.5' (délután fél 4 múlt két és fél másodperccel)
- Bármely érték lehet **NULL** hiányzó érték:

Hiányzó értékek: NULL

- Az SQL lehetővé teszi a táblákban **a hiányzó értékeket**, vagyis a relációk soraiban az attribútum értéke ne legyen megadva, hanem egy speciális **NULL** nullérték legyen.
- **A nullérték értelmezésére** több lehetőségünk is van:
 - **Nem-ismert érték**: például tudom, „Joe’s Bár”-jának van valamilyen címe, de nem tudom, hogy mi az.
 - **Nem-definiált érték**: például a házastárs attribútumnak egyedülálló embereknél nincs olyan értéke, aminek itt értelme lenne, nincs házastársa, ezért nullérték.
 - stb (van olyan cikk, amely több százféle okot felsorol)

SQL DML

Adatbázis tartalmának módosítása

- **A módosító DML utasítások** az adatbázisban tárolt táblák tartalmát változtatják meg.

- 3-féle módosító utasítás létezik:

INSERT - sorok beszúrása

DELETE – sorok törlése

UPDATE – sorok komponensei értékeinek módosítása

- Most a tábla előkészítésénél csak az elsőt nézzük meg, hogy viszünk fel adatsorokat:

A táblák feltöltése adatokkal

- **Példa:** A Szeret táblába beírjuk, Zsu szereti a Bud sört.

```
INSERT INTO Szeret  
VALUES('Zsu', 'Bud');
```

- A reláció neve után megadhatjuk az attribútumait.
- Ennek alapvetően két oka lehet:
 1. elfelejtettük, hogy a reláció definíciójában, milyen sorrendben szerepeltek az attribútumok.
 2. Nincs minden attribútumnak értéke, és azt szeretnénk, ha a hiányzó értékeket NULL vagy default értékkel helyettesítenék.

Példa:

```
INSERT INTO Szeret(sör, név)  
VALUES('Bud', 'Zsu');
```

Default értékek megadása

- A CREATE TABLE utasításban az oszlopnevet **DEFAULT** kulcsszó követheti és egy érték.
- Ha egy beszúrt sorban hiányzik az adott attribútum értéke, akkor a default értéket kapja.

```
CREATE TABLE Sörivók(  
    név CHAR(30)  
    cím CHAR(50) DEFAULT 'Sesame St'  
    telefon CHAR(16) );  
INSERT INTO Sörivók(név)  
VALUES ('Zsu' );
```

Az eredmény sor:

név	cím	telefon
Zsu	Sesame St	NULL

Tankönyv példa: Filmek séma

Filmek(

cím:string,
év:integer,
hossz:integer,
műfaj:string,
stúdióNév:string,
producerAzon:integer)

FilmSzínész(

név:string,
cím:string,
nem:char,
születésiDátum:date)

Stúdió(

név:string,
cím:string,
elnökAzon:integer)

Mit jelentenek az aláhúzások?

Tankönyv példája, hibás fordítás:

title=(film)cím és address=(lak)cím

Tervezéssel később foglalkozunk, ez a példa hibás, az elnevezések, de így jó lesz, hogy a lekérdezéseknél megnézzük hogyan kezeljük.

SzerepelBenne(

filmCím:string,
filmÉv:integer,
színészNév:string)

GyártásIrányító(

név:string,
cím:string,
azonosító:integer,
nettóBevétel:integer)

Példa megszorításokra: Kulcs

- Előző példában: attribútumok aláhúzása mit jelent?
- Filmek: elvárjuk, hogy ne legyen a megengedett előfordulásokban két különböző sor, amelyek megegyeznek cím, év attribútumokon.
- Egyszerű kulcs egy attribútumból áll, de egy kulcs nem feltétlenül áll egy attribútumból, ez az összetett kulcs. Például a **Filmek** táblában a cím és év együtt alkotják a kulcsot, nem elég a cím, ugyanis van például (King Kong, 1933), (King Kong, 1976) és (King Kong, 2005).
- A kulcsot aláhúzás jelöli: **Filmek (cím, év, hossz, ...)**

Kulcs megadása

- **PRIMARY KEY** vagy **UNIQUE**
- Nincs a relációnak két olyan sora, amely a lista minden attribútumán megegyezne.
- Kulcs esetén nincs értelme a DEFAULT értéknek.
- Kulcsok megadásának két változata van:
 - Egyszerű kulcs (egy attribútum) vagy
 - Összetett kulcs (attribútumok listája)

Egyszerű kulcs megadása

- Ha a kulcs egyetlen attribútum, akkor ez az attribútum deklarációban megadható

<attribútumnév> <típus> **PRIMARY KEY**

vagy <attribútumnév> <típus> **UNIQUE**

- Példa:

```
CREATE TABLE Sörök (  
    név          CHAR(20)  UNIQUE,  
    gyártó       CHAR(20)  
);
```

Összetett kulcs megadása

- Ha a kulcs több attribútumból áll, akkor a CREATE TABLE utasításban az attribútum deklaráció után a kiegészítő részben meg lehet adni további tábla elemeket: **PRIMARY KEY** (attrnév₁, ... attrnév_k)
- Példa:

```
CREATE TABLE Felszolgál (  
    söröző      CHAR(20) ,  
    sör         VARCHAR2(20) ,  
    ár          NUMBER(10,2) ,  
    PRIMARY KEY (söröző, sör)  
);
```

PRIMARY KEY vs. UNIQUE

- Csak egyetlen **PRIMARY KEY** lehet a relációban, viszont **UNIQUE** több is lehet.
- PRIMARY KEY egyik attribútuma sem lehet **NULL** egyik sorban sem. Viszont UNIQUE-nak deklarált attribútum lehet NULL, vagyis a táblának lehet olyan sora, ahol a UNIQUE attribútum értéke **NULL vagyis hiányzó érték**.
- az SQL lekérdezésnél adjuk meg hogyan kell ezzel a speciális értékkel gazdálkodni, hogyan lehet NULL-t kifejezésekben és hogyan lehet feltételekben használni
- Következő héten visszatérünk a megszorítások és a hivatkozási épség megadására.

Kérdés/Válasz

- Ehhez az előadáshoz kapcsolódó gyakorlaton az Oracle adatbázisok elérését nézzük meg, és az SQL lekérdezésekhez előkészítjük a tábláinkat az Oracle adatbázisokban.
- Köszönöm a figyelmet! Kérdés/Válasz?