

Budapesti Corvinus Egyetem, Számítástudományi Tanszék

Adatbázis rendszerek előadás jegyzet 2017


Kardkovács Zsolt előadása, Kerepes Tamás jegyzeteivel

Relációs adatmodell és az SQL

*„Én sosem tanítom a diákjaimat,
csak megpróbálom megteremteni a
feltételeket, amelyben tanulni tudnak.”
(Albert Einstein)*



Az első előadás történelmi, majd elméleti fejtegetései után ezúttal gyakorlatiasabb rész következik.



ADATBÁZISOK

Relációs adatmodell

- **Adatmodell:**
 - „**logikai struktúra** és azon értelmezett **kényszerek és műveletek** összessége”

Relációs adatmodell fogalmai

- séma, attribútum, reláció
- unió, különbség, Descartes-szorzat, szűrés, vetítés
- szuperkulcs, kulcs, egyediség, funkcionális függőség

Szubjektív kiemelések:

- Halmaz szemantika(!)
- Minden érték ismert...
- Rendelkezünk minden külső ismerettel...

Ilyen a valóság?

Fog ez így működni?

Amint azt már az első előadásban is említettük: az adatmodell tulajdonképpen a logikai struktúra és ezen a logikai struktúrán értelmezett kényszerek és műveletek összessége.

A több lehetséges adatmodell közül a gyakorlatban a relációs adatmodell maradt meg a mai napig, mint szinte egyeduralkodó a komoly üzleti alkalmazásokban.

Ebben a relációt mint az „attribútumok Descartes-szorzatának a részhalmazát” definiáljuk, de sokkal könnyebb a vizuális (táblázatos) ábrázolásra gondolni: a „relációnk” egy „táblázat” amelyben oszlopok vannak és sorok. Minden oszlop egy attribútum. Ha a relációnknak „4 sora van”, akkor az azt jelenti, hogy a táblázatunk 4 soros, amint azt a következő dia mutatja.

A műveleteink: az únió, a különbség, a Descartes-szorzat, a vetítés, a szűrés, stb. lesznek, és ezeket egyesével ismerjük majd meg.



ADATBÁZISOK

Reláció (példa) az előző előadásból

Vizuális (táblázatos) ábrázolás

Rendezett halmaz?

Hallgatók	Név	Neptun	Telefon	Cím	Kor	Átlag
	Gipsz Jakab	ABC123	4632031	Sóház u. 13.	22	3,52
	Ency Klopédia	ABC124	4632045	Sóház u. 13.	23	4,1
	Kövér Margó	CBA123	4632027	Sóház u. 13.	22	3,78
	Sánta Kutya	ACB123	4632976	Sóház u. 13.	24	4,8

Lényeges észrevenni, hogy a „reláció” egy halmaz. A sorok sorrendjének nincs jelentősége.

Problémákba fogunk ütközni, ha pl. valakinek nincs telefonszáma, vagy pl. nem ismerjük az életkorát.

Szerencsénkre a Név is egyedi a fenti táblában, meg a Neptun kód is. A tábla tartalma azonban időnként változik és a kérdés: számíthatunk-e arra, hogy a Név mindig egyedi marad?

A Neptun-kódban bízhatunk, tehát az alkalmas arra, hogy kulcsként tekintsünk rá.

Életszerűségi kérdések

- **Nem csak egyfajta világmodell létezik...**
 - Elvileg lehet több azonos nevű, azonos nevű anyától és születési idejű ember?
 - Kicsit mindenképpen változtatni kell az adatbevitelen?
 - Vegyek fel mindenképpen új attribútumot?
- **Funkcionális függőségek megadása lehet hibás...**
 - Rejtett függőségek feltárása
 - Mária névből következik, hogy valaki nő. Vagy mégse?
- **Ismeretlen vagy értelmezhetetlen értékek kezelése...**
 - Gyerekek számának megadása egy 2 évesnél?
 - Shakespeare életkora halálakor?
- **Típusok**
 - Szám, szöveg, dátum, különféle LOB (vö. bazinagyadat)

Egy ország állampolgárait gyakran a (Név, Lakcím, Anyja leánykori neve, Születési dátuma) négyessel próbálunk egyedivé tenni. Kérdés, hogy lehetséges-e, hogy ezek mégsem egyediek.

Gyakran kényszerülünk egy új attribútum (egy mesterséges kulcs) felvételére.

A táblák egyes attribútumainak értékei függhetnek egymástól. Pl. képzeljünk el egy olyan attribútumot, hogy Név, és egy olyat, hogy Nem. A Mária névből elvileg következik, hogy az illető nő, de pl. az Alex név már nem biztos, hogy egyértelmű. Ezekre a funkcionális függőségekre egy későbbi előadáson visszatérünk majd.

A relációk attribútumai értékeket vehetnek fel. A gyakorlati szakember ezt úgy mondja: a táblák sorainak és oszlopainak a metszetében, a mezőkben értékek találhatók. Nos, ezek az értékek különböző típusúak lehetnek.

Halmaz vs. zsák szemantika

Halmaz szemantika

- Nincs két azonos elem
 - Unió elemszáma kisebb
 - Tartalmazás, metszet OK
- Egyszerűbb
- Könnyen implementálható

Zsák szemantika

- Bármiből lehet egynél több
 - Unió elemszáma: $N + M$
 - Tartalmazás nem egyértelmű
- Zsák \rightarrow Halmaz
- Ésszerű, de nem logikus...


Műveletek elemszáma?
Legyen r_1 reláció N , r_2 pedig M elemű

A reláció matematikai értelemben egy halmaz (mert a Descartes szorzat részhalmaza). Így tehát ha két reláció únióját képezzük, és az egyiknek is meg a másiknak is 2-2 sora van, akkor az úniónak lehet 2, 3 vagy 4 sora. (Két halmaz úniója egy harmadik halmaz).

A szoftvercégek többsége úgy implementálta a relációs adatbázisokat, hogy azokat módunkban legyen ne csak a „halmaz szemantika”, hanem a „zsák szemantika” szerint is használni.

A „zsák” (bag) annyiban különbözik a halmaztól, hogy benne egy elem többször is szerepelhet.

Ez itt annyit jelent, hogy ha két táblánk van, és mindegyikben két sor, akkor az „únió”-ban 4 sor lesz (nem próbálja a szoftver kiszűrni az egyformákat).



CORVINUS
UNIVERSITY OF
BUDAPEST

ADATBÁZISOK

SQL alapok

SEQUEL (Structured/Standard English QUery Language)

- Nyelvi köntösben
- Logikai nyelv: mit?

Nyelvi szerkezet

- Mondatok vannak (SELECT, CREATE, UPDATE, INSERT, DELETE, ...)
- Vezérlési elemek
 - , szerepe
 - ; szerepe
 - -- szerepe
 - . szerepe
 - ' szerepe
 - operátorok, precedencia

Bármely adatbázis esetén szükség van egy lekérdező nyelvre.

Ha az adatbázis relációs, akkor háromféle lekérdező nyelvet is el lehet képzelni:

- a) Relációs algebra
- b) Relációs sorkalkulus
- c) Relációs oszlopkalkulus

A relációs algebra segítségével megfogalmazott adatbázis lekérdezéseknél explicit módon elő kell írunk, hogy mely relációkon milyen műveleteket, milyen sorrendben kell elvégeznünk.

Ezzel szemben a sor és oszlopkalkulus alapú nyelvek csak azt igénylik, hogy az eredményhalmazra vonatkozóan megfogalmazzuk az elvárásunkat. (Nem azt mondjuk meg, hogy **hogyan** kell az eredményt előállítani, hanem azt, hogy **mi** legyen az eredmény).

Mindez a fenti elmélet fontosnak és okosnak hangzik ugyan, de a gyakorlat ennél jóval egyszerűbb:

Az IBM az 1970-es évek elején (1974): SEQUEL (Structured English Query Language)
Később jogi okokból átnevezték SQL-re.

Oracle: 1979 Az első piaci megjelenés.

Az elméleti szakértők nem túl lelkesek az SQL iránt. Gondnak látják, hogy mind a tábla, mind az SQL lekérdezés eredménye „sorok listája”, tehát zsákszemantika.

Az SQL nyelv azért nagyszerű mert:

- a) Lekérdezéskor angol nyelven szinte helyes mondatokat fogalmazunk meg
- b) A nyelv „nem procedurális”, tehát azt kell benne megadnunk, hogy mi legyen az eredmény, és nem azt, hogy hogyan jussunk el odáig.

Ha már azt mondjuk el, hogy „mit szeretnénk”, akkor a hogyan-t, az „algoritmust” az adatbázis-kezelő rendszer keresi ki, majd hajtja végre.

Az SQL ugyan egy ANSI és egy ISO szabvány, de sajnos túlságosan megengedő. Az egyes gyártók egymástól enyhén eltérő „dialektusokat” beszélnek, és a szabvány ezt mind elfogadja – ami nem jó.



CORVINUS
UNIVERSITY OF
BUDAPEST

ADATBÁZISOK

Szerkezet felől nézve...

Összetett példa

- **CREATE TABLE** Hallgatók (-- DESC <táblanév>

Név	VARCHAR2(255)	NOT NULL -- példa megjegyzés ez is
, Neptun	CHAR(6)	PRIMARY KEY
, Telefon	VARCHAR(16)	UNIQUE
, ZIPCode	INTEGER	FOREIGN KEY REFERENCES Körzet(ID)
, Képzés	VARCHAR2(255)	
, Kor	NUMBER(3,0)	CHECK (Kor > 18 AND Kor < 150)
, Átlag	NUMBER(4,3)	
, Változás	DATE	DEFAULT SYSDATE
, CONSTRAINT átlag_feltétel CHECK (Átlag > 1000 AND Átlag < 5000)		
, CONSTRAINT külső_kulcs FOREIGN KEY (Képzés) REFERENCES Képzés(ID)		
ON DELETE CASCADE		

);

Figyelj a vesszőkre!!!

Az SQL utasítások csoportosíthatóak:

- a) DML (Data Manipulation Language): INSERT, DELETE, UPDATE, SELECT, ...
- b) DDL (Data Definition Language): CREATE, DROP, ALTER, ...
- c) DCL (Data Control Language): GRANT, REVOKE, ...
- d) ...

Fenti dián egy közepesen bonyolult tábla kreálása (létrehozása) látható. Ez egy DDL parancs.

Nem szerencsés ezzel kezdeni, mert túl sok szintaktikai elem jelenik meg a dián, és még elmegy a kedvünk az egésztől, de ez van ☹

Alapötlet

- Projekció + Descartes-szorzat bárhol
- Halmazműveletek és szelekció sehol

Relációs alpművelet megfeleltetés

- SELECT DISTINCT = vetítés
 - **SELECT = SELECT ALL** = vetítés/projekció zsákszemantikával
- FROM -> Descartes-szorzat (üres halmazzal)
- WHERE = kiválasztás
- UNION = unió
 - **UNION ≠ UNION ALL** = zsák szemantika
- MINUS = különbség

Az SQL „lekérdező nyelve” valójában egyetlen utasítás: SELECT

Létezik SELECT DISTINCT és SELECT

Az első a matematikai értelemben vett vetítés (halmaz szemantikával), míg a másik a zsák-szemantika.

Mindkét lekérdezés angolul elolvasva teljesen érthető. A nyelv könnyen megjegyezhető és megtanulható. Nagyon rövid taulás után az ember képes ilyen lekérdezéseket megfogalmazni. Világszerte több millióan ismerik, használják ezt a nyelvet.

Az SQL nyelvben létezik UNION és UNION ALL utasításrész egyaránt (a második az zsák-szemantika)

Egyszerű vetítés példa

Vetítés

- (Algebra) $\Pi[\text{Neptun, Cím}](\text{Hallgatók}) = \Pi[\text{Neptun, Cím}](\emptyset \times \text{Hallgatók})$
- (SQL) `SELECT DISTINCT Neptun, Cím FROM Hallgatók`
- (SQL) `SELECT Neptun, Cím FROM Hallgatók`

Hallgatók	Név	Neptun	Telefon	Cím	Kor	Átlag
	Gipsz Jakab	ABC123	4632031	Sóház u. 13.	22	3,52
	Hibás Elemér	ABC123	4632976	Sóház u. 13.	24	4,8

Vetítés	Neptun	Cím
	ABC123	Sóház u. 13.

Vetítés2	Neptun	Cím
	ABC123	Sóház u. 13.
	ABC123	Sóház u. 13.

Ha a Hallgatók tábla a fenti szerkezetű, akkor lekérdezhethetjük belőle pl. a Neptun kódot és a címet a következő két változattal:

`SELECT DISTINCT Neptun, Cím FROM Hallgatók;`

`SELECT Neptun, Cím FROM Hallgatók;`

Mindkét fenti utasítást teljes egészében „megértjük” angolul. Ez az SQL fő előnye.



CORVINUS
UNIVERSITY OF
BUDAPEST

ADATBÁZISOK

Egyszerű szűrés példa

Szűrés + egyéb dolgok...

- **Algebra**
 - $\sigma(\text{Hallgatók} \mid \text{Neptun} = \text{'ABC123'})$
 - $\Pi[\text{Név, Neptun, Telefon, Cím, Kor, Átlag}] \sigma(\text{Hallgatók} \mid \text{Neptun} = \text{'ABC123'})$
- **SQL**
 - `SELECT DISTINCT * FROM Hallgatók WHERE Neptun = 'ABC123'`
 - `SELECT * FROM Hallgatók WHERE Neptun = 'ABC123'`

Hallgatók	Név	Neptun	Telefon	Cím	Kor	Átlag
	Gipsz Jakab	ABC123	4632031	Sóház u. 13.	22	3,52
	Sánta Kutya	ACB123	4632976	Sóház u. 13.	24	4,8

Kiválasztás	Név	Neptun	Telefon	Cím	Kor	Átlag
	Gipsz Jakab	ABC123	4632031	Sóház u. 13.	22	3,52

Ha nincs vetítés, akkor a tábla minden oszlopa érdekel minket. Megtehetjük, hogy ezeket mind felsoroljuk a SELECT listában, de mivel ez egy elég gyakori igény, létezik rá egy rövid jelölés:

`SELECT * FROM Hallgatók;`

vagy

`SELECT DISTINCT * FROM Hallgatók;`

Fenti dián a szűrés (szelekció) látható:

`SELECT * FROM Hallgatók WHERE Neptun = 'ABC123'`

Egyszerű halmazos műveletek

Halmazműveletek

- **Unió**
 - (Halmaz) SELECT * FROM Alap **UNION** SELECT * FROM Alap
 - (Zsák) SELECT * FROM Alap **UNION ALL** SELECT * FROM Alap
- **Különbség**
 - SELECT * FROM Alap **MINUS** SELECT * FROM Alap
- **Metszet**
 - SELECT * FROM Alap **INTERSECT** SELECT * FROM Alap

Mi van az
attribútumnevekkel?

Alap	Név	Neptun	Telefon	Cím	Kor	Átlag
	Gipsz Jakab	ABC123	4632031	Sóház u. 13.	22	3,52
	Ency Klopédia	ABC124	4632045	Sóház u. 13.	23	4,1
	Sánta Kutya	ACB123	4632976	Sóház u. 13.	24	4,8

Ha a két tábla „azonos” szerkezetű (tehát azonos nevű és adattípusú oszlopaik vannak, akkor definiáltak a UNION, a UNION ALL, az INTERSECT és a MINUS műveletek.



ADATBÁZISOK

Hogyan van ez a másik formában?

Relációs algebra

- $\Pi[\text{Név, Neptun, Tárgy, Jegy}](\sigma(\sigma(\text{Hallgatók} \mid \text{Félév} = '2014-2015/2') \times \Pi[\text{Tárgy, Jegy}](\text{Tárgyak}) \mid \text{Hallgatók.Neptun} = \text{Tárgy.Hallgató}))$

SQL

- `SELECT a.Név, b.Név, a.Születésnap FROM Hallgatók a, Hallgatók b
WHERE a.Születésnap = b.Születésnap
AND a.Neptun <> b.Neptun;`

Szöveges megfogalmazás

- Keressük azon hallgatókat, akik Adatbázis rendszerek és Szoftvertechnológia I, valamint Szoftvertechnológia II. tárgyakból is jelest kaptak.

Haladó gyakorlatként megpróbálkozhat valaki a fenti relációs algebrai kifejezést magyar mondattá átfordítani.

A gyakorlati munka szempontjából sokkal fontosabb azonban az SQL lekérdezést figyelmesen megvizsgálni. Valójában a Hallgatók tábla illesztése történik önmagával. Ez egy „equi-join” azaz egyen-illesztés, vagy egyen-összekapcsolás. Az azonos születésnapúakat illesztjük, de úgy, hogy önmagával senkit sem párosítunk.

Ha fenti SELECT esetén pl. JANCSI és JULISKA születésnapja azonos (de a Neptun kódjuk természetesen különböző), akkor két alkalommal is ki lesznek listázva: úgy mint

JANCSI, JULISKA

Majd

JULISKA, JANCSI

Ennél durvább, ha pl. 4 hallgatónak azonos a születésnapja, akkor ez $4 * 4 - 4 = 12$ sornyi eredményt jelent.

Azt mondhatjuk, hogy azért van szükségünk a táblát önmagával illeszteni, mert így tudunk a tábla különböző soraiban lévő adatokat egymással relációba hozni. Ilyenkor „segédnevet” kell adnunk a táblánknak, azaz tábla „alias”-t.

Szöveges megfogalmazás ismét

- Keressük azon hallgatókat, akik Adatbázis rendszerek és Szoftvertchnológia I, valamint Szoftvertchnológia II. tárgyakból is jelest kaptak.
- **Megoldási javaslat:**
 - Keressük azokat, akik nem
 - És vonjuk ki az eredeti halmazból...
 - 1. lépés:

$$A = \sigma_{\{ \Pi[\text{Név, Jegy}](\text{Hallgató} \bowtie \text{Tárgy}) \mid \text{Jegy} = 5 \}}$$
 - 2. lépés

$$B = \Pi[\text{Név, Jegy}](A \times \Pi[\text{Tárgy}](\text{Tárgyak}))$$
 - 3. lépés

$$C = B - \Pi[\text{Név, Jegy}](\text{Hallgató} \bowtie \text{Tárgy})$$
 - 4. lépés

$$\text{Megoldás} = \Pi[\text{Név, Jegy}](\text{Hallgató} \bowtie \text{Tárgy}) - C$$

Ismét a relációs algebra szépségeit meghagynám az ilyen irányban affinitást érzőknek. Aki Gazdasági informatikus, annak majd SQL nyelven kell ezt megfogalmaznia, vagy megértenie, ha azt mások fogalmazzák meg.

SQL-ben tehát így is megfogalmazható ez:

```
SELECT Nev, Neptun FROM Hallgatok, Jegyek
WHERE Hallgatok.Neptun = Jegyek.Neptun
  AND Targy = 'Adatbázis rendszerek'
  AND Jegy = 5
INTERSECT
SELECT Nev, Neptun FROM Hallgatok, Jegyek
WHERE Hallgatok.Neptun = Jegyek.Neptun
  AND Targy = 'Szoftvertchnológia I'
  AND Jegy = 5
INTERSECT
SELECT Nev, Neptun FROM Hallgatok, Jegyek
WHERE Hallgatok.Neptun = Jegyek.Neptun
  AND Targy = 'Szoftvertchnológia II'
  AND Jegy = 5;
```

Fontos: ez messze nem az egyedüli megoldási lehetőség SQL-ben!



ADATBÁZISOK

Segédműveletek

Alternatív hivatkozások megadása

- **Eredményoszlop átnevezése**
 - `SELECT Név AS Name FROM Alap`
- **Tábla átnevezése lekérdezésben**
 - `SELECT Név FROM Alap a`
 - `SELECT a.Név FROM Alap a`

Egyértelműsítés

- `SELECT * FROM Alap a, Alap b`
- `SELECT a.*, b.* FROM Alap a, Alap b`

Hivatkozás „útvonallal”:

- **Konténer.objektum.attribútum**
(pl. `SCOTT.EMP.NAME` / `wordpress.wp_posts.post_type`)

Átnevezés hatóköre?

Átnevezés hatóköre?

Hogyan értelmezi a gép?

Oszlop-alias (segédnév) és tábla-alias.

A `SCOTT.EMP.ENAME` értelme: sémanév.táblanév.oszlopnév.

Rendezés

- **ORDER BY = ORDER BY ... ASC = ORDER BY ... ASCENDING**
 - SELECT * FROM Alap **ORDER BY** Név
 - SELECT * FROM Alap **ORDER BY** Név **DESC**, Kor **ASC**?
 - SELECT Név AS Name FROM Alap ORDER BY Name?

Műveletek

- **Aritmetikai műveletek**
 - SELECT 1 + Kor / 2 FROM Alap
 - SQRT / ABS / SIGN / ...
- **Szövegmanipulációs műveletek**
 - SELECT 'Az én nevem: ' || Név FROM Alap
 - SUBSTR / UPPER / LPAD / TRIM / MATCH / ...

Segédműveletekre példák

Műveletek (folyt.)

Itt kezdünk megörülni...

▪ Dátumkezelő műveletek

- SELECT 1 + **TO_DATE**('1848-03-15', 'YYYY-MM-DD') as Dátum FROM Alap?
- SELECT 1 + **CAST**('1848-03-15' **AS DATETIME**) as Dátum FROM Alap?
- ??? TO_CHAR vagy CONVERT / ADD_MONTHS / MONTHS_BETWEEN / ...
- Különböző naptárak kezelése

Konstansok és segédtablák

▪ Mai nap

- ☺... SYSDATE / NOW() / SYSDATETIME / GETDATE() / ...

▪ Segédtablák eltérő használata

- (MSSQL, MySQL, Teradata) SELECT 1 + 1
- (Oracle, DB/2) SELECT SYSDATE **FROM DUAL**





ADATBÁZISOK

Illesztés példa (ismétlés)

Illesztés

- Matematikailag
 - Illesztés = Hallgatók \bowtie Jegyek =

Π [Név, Neptun, Átlag, Tárgy, Jegy] (

σ (Hallgatók \times Jegyek | Hallgatók.Neptun = Jegyek.Neptun)

)

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52
	Sánta Kutya	ACB123	4,8

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4
	Sánta Kutya	ACB123	4,8	Adatbázisok	5

Fenti dián a Jegyek és Hallgatók táblák Theta-illesztése szerepel. Ezen belül is ez egy equi-join, egyen-illesztés, vagy egyen-összekapcsolás. Az azonos értékűeket helyezzük egymással relációba.

Emellett ez még egy természetes illesztés is (natural join), hiszen az azonos nevű és típusú attribútumok szerint kapcsoljuk őket össze.

Illesztés egyszerű változata...

▪ **Matematikailag**

- $A \bowtie B = \prod [\dots] (\sigma (A \times B \mid \dots))$

Nem azt mondtad,
hogy szabványos??!?!?

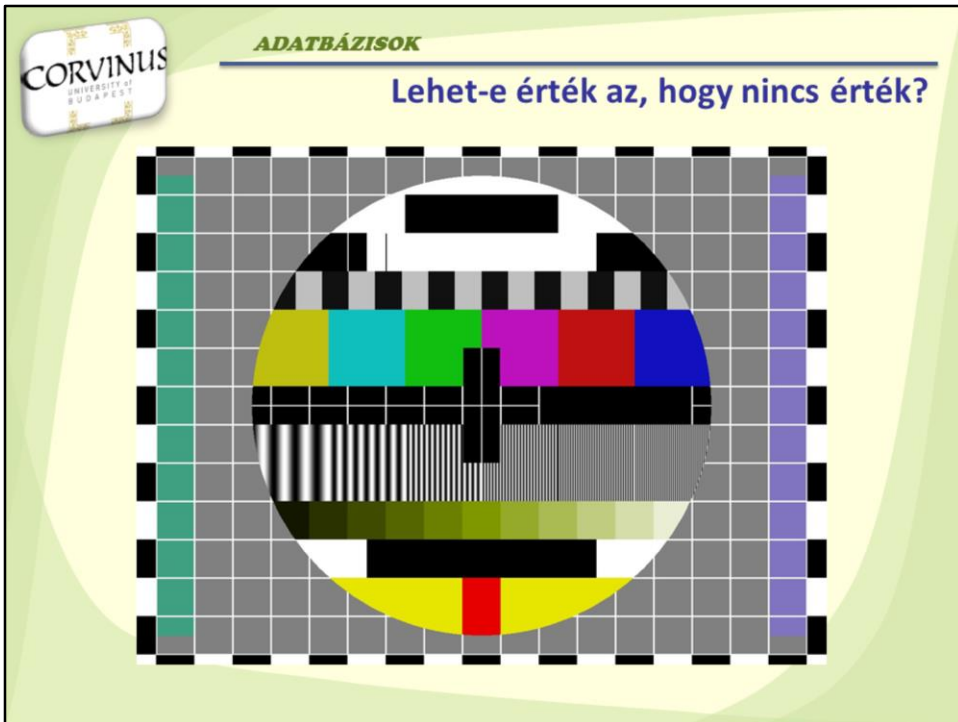
▪ **Szokásos átírások**

- (Ekvivalens 1) **SELECT ... FROM A, B WHERE ...**
-> **SELECT A.*, Tárgy, Jegy FROM Hallgatók A, Jegyek B**
WHERE A.Neptun = B.Neptun
- (Ekvivalens 2) **SELECT ... FROM A INNER JOIN B ON ...**
- (Ekvivalens 3) **SELECT ... FROM A JOIN B USING (...)**
- ...

Hol az információ? Az adatokban elrejtve.
Hol vannak az adatok? A #@\$%?!&
adatbázisban elrejtve. (Joe Celko)



Sajnos az SQL szabvány túlságosan megengedő volt. Az eredmény: az illesztésnek (nevezik még összekapcsolásnak is), tehát a JOIN-nak néhány alternatív szintaxisa megengedett. Ez sokat levon az SQL értékéből.



A NULL érték: ha valódi, matematikai értelemben vett relációkról beszélünk, akkor ott nem adódik ilyen helyzet.

A való életben a tábláinkban ez nem ritkán megesik, mert megengedjük ezt az adatmodell választásakor.

NULL érték jelentései

- Nem ismert adat
- Nem értelmezett adat
- Törölt adat

Műveletek

- Egyenlőség problémája
 - 1 = NULL vagy 1 <> NULL?
 - 1 **IS NULL** vagy 1 **IS NOT NULL**
- Logikai értékek bővülnek:
 - TRUE, FALSE és UNKNOWN
 - NVL(<változó>, <érték>) / ISNULL(<változó>, <érték>) / IFNULL / COALESCE

Illesztéssel akkor mi van?

Nem ismert adat: pl. az ügyfeleink lakcíme. NULL érték, mert még nem ismerjük.

Nem értelmezett adat: a DOLGOZÓK táblában a dolgozók kódja, neve, egyéb attribútuma szerepel, és ezek között van a MUNKAKÖR, a FIZETÉS és a JUTALÉK. Akinek a MUNKAKÖR-e az, hogy „KERESKEDŐ”, annak FIZETÉS-s is van, meg JUTALÉKA is. Aki azonban nem kereskedő, hanem pl. „OKTATÓ”, annak csak fizetése van, és az ő sorában a JUTALÉK nem értelmezett. NULL érték, mert nincs értelme.

Törölt adat: ott volt az a lakcím, csak utólag töröltük ezt az információt. Ezzel együtt azonban az adat több része (pl. az ügyfél neve, kódja, életkora) megmaradtak.