Kerepes Tamás – Czinkóczki László

Adatbázisok

Táblák összekapcsolása (Join) és adatok aggregálása

„Igyekezz megtanulni valamit mindenről
és mindent valamiről.”
(Thomas Henry Huxley)

SQL szintaxis… eddig

Lekérdezések eddigi formái (messze nem ez a teljeskörű szintaxis):

- **SELECT … FROM …**
  - [WHERE …]
  - [ORDER BY …]
- **[<UNION | UNION ALL | INTERSECT | MINUS>**
- **SELECT … FROM … [WHERE …] [ORDER BY …]]**

Főbb megállapítások

- **Szabványos ☺**
- **Angolszerű mondatok**
- **NULL értékek problémái szelekciók és összekapcsolások esetén**

NULL – „nincs adat..”

NULL értékek problémája (a NULL nem egyenlő a zérussal)

- SELECT * FROM Hallgatók WHERE Neptun = NULL - üres eredményhalmaz
- SELECT * FROM Hallgatók WHERE Neptun IS NULL - ez a helyes

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52
	Sánta Róka		4,03

Összekapcsolások típusai

Az SQL:1999 szabvány szerinti összekapcsolások és Oracle-specifikusak

- **Egyen-összekapcsolások (Equi-Join)**
  - Natural joins
  - `USING` klauzula
  - Oracle szintaxis szerinti
  - ON klauzula használata
- **Nem egyen-összekapcsolások (Non equi-join)**
- **Külső összekapcsolások (Outer Join)**
  - Left Outer Join
  - Right Outer Join
  - Full Outer join
- **Descartes szorzat (Cross join)**

A természetes összekapcsolás (Natural Join)

```
SELECT Név, Neptun, Átlag, Tárgy, Jegy
FROM   Hallgatók NATURAL JOIN Jegyek
WHERE  Átlag > 3.00;
```

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4

USING klauzula használata

```
SELECT Név, Neptun, Átlag, Tárgy, Jegy
FROM    Hallgatók JOIN Jegyek
USING (Neptun)
WHERE  Átlag > 3.00;
```

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4

Hagyományos (Oracle) szintaxis szerint

```
SELECT Név, Hallgatók.Neptun, Átlag, Tárgy, Jegy
FROM   Hallgatók, Jegyek
WHERE Hallgatók.Neptun = Jegyek.Neptun AND
          Átlag > 3.00;
```

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4

Egyenösszekapcsolás ON klauzulával

```
SELECT Név, Hallgatók.Neptun, Átlag, Tárgy, Jegy
FROM   Hallgatók JOIN Jegyek
ON         Hallgatók.Neptun = Jegyek.Neptun
WHERE  Átlag > 3.00;
```

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4

Nem egyen-összekapcsolások (Non-Equijoins)

WORKERS

	FIRST_NAME	LAST_NAME	SALARY
1	Ernest Miller	HEMINGWAY	8300
2	Carl Friedrich	GAUSS	24000
3	Leonard	EULER	17000
4	Johann	BERNOULLI	17000
5	Daniel	BERNOULLI	9000
6	Bruce	WILLIS	6000
7	Giuseppe	VERDI	4800
8	Hendrik	LORENTZ	4200
9	Edvard	GRIEG	12000
10	Gustave	FLAUBERT	9000
11	John	LENNON	8200

JOB_GRADES

GRA	LOWEST_SAL	HIGHEST_SAL
A	1000	2999
B	3000	5999
C	6000	9999
D	10000	14999
E	15000	24999
F	25000	40000

Minden dolgozó (WORKERS) 1-1 fizetési kategóriába tartozik

Példa a nem egyen-összekapcsolásra

```
SELECT e.first_name,e.last_name, e.salary, j.grade_level
FROM   workers e JOIN job_grades j
ON     e.salary
       BETWEEN j.lowest_sal AND j.highest_sal;
```

	FIRST_NAME	LAST_NAME	SALARY	GRADE_LEVEL
1	Steven	SEAGAL	2200	A
2	Indira	GANDHI	2600	A
3	Nobby	STILES	5000	B
4	Carlos	SANTANA	6500	C
5	Antoni	GAUDI	7000	C
6	William	SHAKESPEARE	7400	C
7	Isaac	STERN	7700	C
8	Gyula	GROSICS	10500	D
9	Ben	GURION	11000	D
10	Edvard	GRIEG	12000	D
11	Carl Friedrich	GAUSS	24000	E

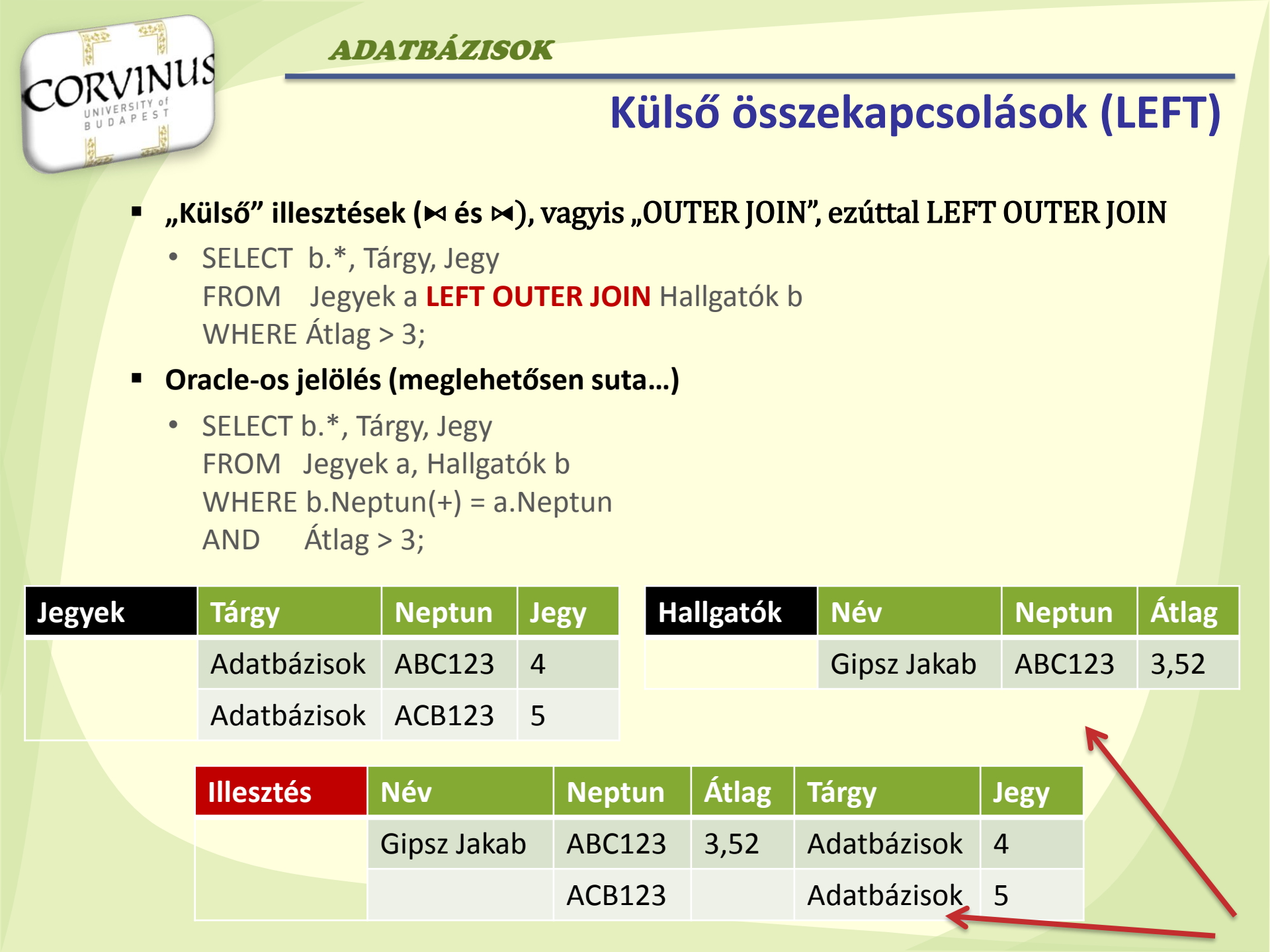
Külső összekapcsolások (LEFT)

- **„Külső” illesztések (⋈ és ⋈), vagyis „OUTER JOIN”, ezúttal LEFT OUTER JOIN**
  - SELECT b.*, Tárgy, Jegy
    FROM Jegyek a **LEFT OUTER JOIN** Hallgatók b
    WHERE Átlag > 3;
- **Oracle-os jelölés (meglehetősen suta…)**
  - SELECT b.*, Tárgy, Jegy
    FROM Jegyek a, Hallgatók b
    WHERE b.Neptun(+) = a.Neptun
    AND Átlag > 3;

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52

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



Külső összekapcsolások (RIGHT)

- **„Külső” illesztések (⋈ és ⋈), ezúttal RIGHT OUTER JOIN**
  - SELECT b.*, Tárgy, Jegy
    FROM Jegyek a **RIGHT OUTER JOIN** Hallgatók b
    WHERE Átlag > 3

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok	ACB123	5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52
	Sánta Róka		4,03

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4
	Sánta Róka		4,03		

Teljes külső összekapcsolások (FULL)

Teljes külső összekapcsolás (⋈)

- SELECT a.*, b.Tárgy, b.Jegy FROM Hallgatók a **FULL OUTER JOIN** Jegyek b;

Jegyek	Tárgy	Neptun	Jegy
	Adatbázisok	ABC123	4
	Adatbázisok		5

Hallgatók	Név	Neptun	Átlag
	Gipsz Jakab	ABC123	3,52
	Sánta Róka		4,03

Illesztés	Név	Neptun	Átlag	Tárgy	Jegy
	Gipsz Jakab	ABC123	3,52	Adatbázisok	4
	Sánta Róka		4,03		
				Adatbázisok	5

Descartes szorzat (Cross Join)

CROSS JOIN

- **A „szabványos” Descartes szorzat**
  - SELECT b.*, Tárgy, Jegy
    FROM Jegyek a **CROSS JOIN** Hallgatók b
    WHERE Átlag > 3;

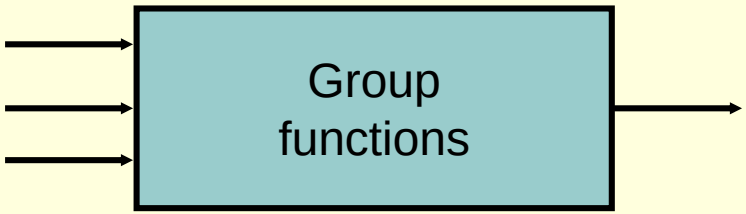
- **Oracle-os jelölés**
  - SELECT b.*, Tárgy, Jegy
    FROM Jegyek a, Hallgatók b
    WHERE Átlag > 3;

Csoportfüggvények és csoportok képzése

A csoportfüggvények típusai

A csoportfüggvények több sorhoz egy soros eredményt adnak

- **AVG**
- **COUNT**
- **MAX**
- **MIN**
- **STDDEV**
- **SUM**
- **VARIANCE**
- …



AVG, SUM , MIN, MAX és COUNT függvények használata

COUNT (*) a lekérdezés által kiválasztott sorok számát adja vissza

```
SELECT AVG(salary), MAX(salary),
       MIN(salary), SUM(salary),count(*)
FROM   workers
WHERE  position_id LIKE '%REP%';
```

AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)	COUNT(*)
8135.71429	11500	6200	113900	14

```
SELECT MIN(last_name),MAX(last_name),
MIN(start_date), MAX(start_date)
FROM workers;
```

	MIN(LAST_NAME)	MAX(LAST_NAME)	MIN(START_DATE)	MAX(START_DATE)
1	ABEL	WILLIS	17-JUN-1987	17-MAR-2014

A DISTINCT kulcsszó használata a csoportfüggvényekben

COUNT(*expr*) a NULL-tól különböző értékek számát adja vissza:

```
SELECT COUNT(commission)
FROM   workers
WHERE  division_id = 80;

COUNT(COMMISSION)
-----------------
                9
```

COUNT(DISTINCT expr) – ha több egyforma van, azok egynek számítanak

```
SELECT COUNT(DISTINCT commission)
FROM   workers
WHERE  division_id = 80;

COUNT(DISTINCTCOMMISSION)
-------------------------
                        6
```

Csoportfüggvények és Null értékek

A csoportfüggvények a NULL-értékeket figyelmen kívül hagyják:

```
SELECT TO_CHAR(AVG(commission),'0.9999') AVERAGE_SALARY
FROM   workers;

AVERAGE_SALARY
--------------
 0.2286
```

Az NVL függvény segítségével a NULL értékűeket is figyelembe vesszük:

```
SELECT
 TO_CHAR(AVG(NVL(commission,0)),'0.9999') AVERAGE_SALARY
FROM   workers;

AVERAGE_SALARY
--------------
 0.0604
```

Csoportok képzése a GROUP BY klauzulával

```
SELECT      column, group_function(column)
FROM        table
[WHERE      condition]
[GROUP BY   group_by_expression]
[ORDER BY   column];
```

A tábla sorait logikailag csoportosítjuk a GROUP BY utasításrész segítségével

Using the GROUP BY Clause

A SELECT listában lévő elemek vagy a GROUP BY listájában szerepelnek, vagy csoportfüggvények argumentumai

```
SELECT    division_id,
TO_CHAR(AVG(salary),'99,999.99') avg_sal
FROM      workers
GROUP BY division_id ;
```

	DIVISION_ID	AVG_SAL
1	100	6,600.00
2	30	5,170.00
3		7,000.00
4	120	2,500.00
5	210	8,525.00
6	90	19,333.33
7	20	13,000.00
8	70	10,000.00
9	230	6,500.00

A csoportfüggvények szabálytalan használata

```
SELECT division_id, COUNT(last_name)
FROM   workers;
```

```
Error starting at line : 1 in command -
SELECT division_id, COUNT(last_name)
FROM   workers
Error at Command Line : 1 Column : 8
Error report -
SQL Error: ORA-00937: not a single-group group function
00937. 00000 -  "not a single-group group function"
```

Hiányzó GROUP BY klauzula (division_id)

További tipikus hibák

- Csoportfüggvényt **WHERE** klauzulában nem használatunk
- Csoportok szűrése a **HAVING** opcióval történik.

```
SELECT   division_id, AVG(salary)
FROM     workers
WHERE    AVG(salary) > 8000
GROUP BY division_id;
```

```
Error starting at line : 1 in command -
SELECT   division_id, AVG(salary)
FROM     workers
WHERE    AVG(salary) > 8000
GROUP BY division_id
Error at Command Line : 3 Column : 10
Error report -
SQL Error: ORA-00934: group function is not allowed here
00934. 00000 -  "group function is not allowed here"
```

Csoportok szűrése: a HAVING klauzula

A következő lépések történnek a HAVING használatakor:

1. **A sorokat csoportokba helyezzük**
2. **A csoportokra alkalmazzuk a csoportfüggvényeket**
3. **A szükséges csoportokat kiválasztjuk a HAVING opcióval**

```
SELECT     column, group_function
FROM       table
[WHERE     condition]
[GROUP BY  group_by_expression]
[HAVING    group_condition]
[ORDER BY  column];
```

HAVING használata

```
SELECT   position_id, SUM(salary) PAYROLL
FROM     workers
WHERE    position_id  LIKE '%S%'
GROUP BY position_id
HAVING   SUM(salary) > 13000
ORDER BY SUM(salary);
```

```
SELECT T.qualified_salary,COUNT(*),SUM(SALARY),COUNT(*)
FROM (SELECT last_name,salary,
      CASE WHEN salary<5000  THEN 'Low'
           WHEN salary<10000 THEN 'Medium'
           WHEN salary<20000 THEN 'Good'
           ELSE 'Excellent'   END qualified_salary
FROM WORKERS) T
GROUP BY T.qualified_salary
HAVING COUNT(*)>1;
```

	QUALIFIED_SALARY	COUNT(*)	SUM(SALARY)	COUNT(*)_1
1	Good	12	151000	12
2	Low	18	58680	18
3	Medium	22	162200	22

Csoportfüggvények egymásba ágyazása

Maximum két szinten ágyazhatók egymásba:

```
SELECT   MAX(AVG(salary))
FROM     workers
GROUP BY division_id;
```

	MAX(AVG(SALARY))
1	18834.333333333333333333333333333333333

Hierarchikus lekérdezések…

Fastruktúrájú adatokat relációs táblákban helyezhetünk el, és hierarchikus lekérdezésekkel tudjuk rekonstruálni a hierarchiát

- **Termékkategória-alkategória / dolgozók-főnökök / …**
  - SELECT Alkategória AS Elem FROM Teáor
    START WITH Főkategória = ‘Oktatás’ -- a fa gyökérponjának a kijelölése
    CONNECT BY Főkategória = **PRIOR** Alkategória -- a bejárás iránya

TEÁOR	Főkategória	Alkategória
	Oktatás	Alapfokú oktatás
	Oktatás	Egyéb oktatás
	Oktatás	Felső szintű oktatás
	Egyéb oktatás	Járművezető-oktatás
	Egyéb oktatás	Sport, szabadidő képzés
	Egyéb oktatás	Kulturális képzés
	Felső szintű oktatás	Felső szintű, nem felsőfokú oktatás
	Felső szintű oktatás	Felsőfokú oktatás

Bonyolítsunk…

Észrevételek

- **Lekérdezések eredményei relációk…**
- **…de olykor lehetnek értékek is**

Összetett lekérdezéseket…

- **INLINE nézetek**
  - SELECT a.Név **FROM (**
    SELECT Név, Lakcím FROM Hallgatók
    WHERE Átlag > 4.2 **) a**
    WHERE UPPER(Lakcím) LIKE ‘%BUDAPEST%’;
- **Többoszlopos allekérdezés**
  - SELECT Név FROM Hallgatók
    WHERE **(** Átlag, Lakcím **) IN (**
    SELECT Átlag, Lakcím FROM Hallgatók
    WHERE UPPER(Lakcím) LIKE ‘%BUDAPEST%’ AND Átlag > 4.2 **)**;

Tárolt lekérdezések

Ha minden lekérdezés reláció…

Miért is jó ez nekünk?

- **Eredménytárolás (statikus)**
  - **CREATE TABLE** Újtábla **AS (** SELECT … **)**
- **Nézet létrehozása**
  - **CREATE VIEW** Újnézet **AS** SELECT …
- **Materializált nézet: a fenti kettő kombinációja**
  - **CREATE MATERIALIZED VIEW** Matview
    REFRESH FAST NEXT SYSDATE + 7 **AS** (SELECT * FROM Hallgatók);
  - **CREATE MATERIALIZED VIEW** Matview **BUILD IMMEDIATE**
    REFRESH FAST ON COMMIT AS (SELECT * FROM Hallgatók);