SQL alapok II. - Lekérdezések alapjai

SELECT

Egy vagy több tábla vagy nézettábla tartalmát kérdezhetjük le. Használhatjuk beépítve más SQL utasításban is (allekérdezés).

Általános alak.

```
1.  SELECT
2.      [ALL | DISTINCT | DISTINCTROW ]
3.      oszlopnév(ek) [, kifejezések ...]
4.      [FROM táblanév(ek)]
5.      [WHERE feltételes sorokra]
6.      [GROUP BY {oszlopnév(ek) | kifejezés }]
7.      [HAVING feltétel]
8.      [ORDER BY {oszlopnév | kifejezés}
9.        [ASC | DESC]]
10.     [LIMIT {sorszám}]
```

Mint látható az általános alak összetett, így nézzük meg részletesebben részenként. A lekérdezésekhez a következő foldrajz adatbázis orszagok adattábláját használjuk.

Id	INT	azonosító
Orszag	VARCHAR	az ország neve
Fovaros	VARCHAR	az ország fővárosa
Foldr_hely	VARCHAR	földrajzi elhelyezkedés
Terulet	DECIMAL	terület km^2-ben
Allamforma	VARCHAR	államforma
Nepesseg	INT	népesség (1000 főben van megadva!)
Nep_fovaros	INT	a főváros népessége (1000 főben van megadva!)
Autojel	VARCHAR	autójel
Country	VARCHAR	országnév ékezetes írás nélkül, idegen elnevezésekkel is
Capital	VARCHAR	főváros ékezetes írás nélkül, idegen elnevezésekkel is
Penznem	VARCHAR	pénznem
Penzjel	VARCHAR	pénzjel
Valtopenz	VARCHAR	váltópénz a váltószámmal együtt
Telefon	INT	nemzetközi telefon-hívószám
GDP	INT	egy főre jutó bruttó hazai termék USA dollárban (GDP: gross domestic product)
Kat	N 1	ismertségi kategória (1,2,3)

Egyszerű alak

```
1.  SELECT oszlop1, oaszlop2, ...
2.  FROM tablaNeve;
```

Illetve, ha mindent meg akarunk jeleníteni.

```
1.  SELECT * FROM tablaNeve;
```

Példa

```
1.  SELECT orszag, fovaros FROM orszagok;
```

Eredmény

	ABC orszag	ABC fovaros
1	SPANYOLORSZÁG	MADRID
2	PORTUGÁLIA	LISSZABON
3	FRANCIAORSZÁG	PÁRIZS
4	NAGY-BRITANNIA	LONDON
5	NORVÉGIA	OSLO
6	SVÉDORSZÁG	STOCKHOLM
7	FINNORSZÁG	HELSINKI

DISTINCT

A SELECT DISTINCT utasítás csak a különböző értékek visszaadására szolgál. A teljes sorra vonatkozik.

```
1.  SELECT DISTINCT oszlop1, oszlop2, ...
2.  FROM tablaNeve;
```

Példa

```
1.  SELECT allamforma FROM orszagok;
```

Eredmény

	ABC allamforma
1	alkotmányos monarchia
2	köztársaság
3	köztársaság
4	alkotmányos monarchia
5	alkotmányos monarchia
6	alkotmányos monarchia
7	köztársaság

```
1.  SELECT DISTINCT allamforma FROM orszagok;
```

Eredmény

	ABC allamforma
1	alkotmányos monarchia
2	köztársaság
3	szövetségi köztársaság
4	nagyhercegség
5	hercegség
6	elnöki köztársaság
7	népköztársaság
8	szövetségi állam

A DISTINCT COUNT-tal együtt is használható.

Példa

Írassuk ki, hogy hány különböző államforma van!

```sql
SELECT COUNT(DISTINCT allamforma) FROM orszagok;
```

Eredmény

	count(distinct allamforma)
1	24

WHERE

A sorok szűrésére használhatjuk.

```sql
SELECT oszlop1, oszlop2, ...
FROM tablaNev
WHERE feltetel;
```

Példa

Írassuk ki a köztársaságok adatait!

```sql
SELECT *
FROM orszagok
WHERE allamforma = "köztársaság";
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma
1	2	PORTUGÁLIA	LISSZABON	Dél-Európa:Ibériai-félsziget	92,082.00	köztársaság
2	3	FRANCIAORSZÁG	PÁRIZS	Nyugat-Európa	547,026.00	köztársaság
3	7	FINNORSZÁG	HELSINKI	Észak-Európa	338,107.00	köztársaság
4	11	OLASZORSZÁG	RÓMA	Dél-Európa:Appennini-félsziget	301,252.00	köztársaság
5	12	MAGYARORSZÁG	BUDAPEST	Közép-Európa:Kárpát-medence	93,036.00	köztársaság
6	14	BULGÁRIA	SZÓFIA	Dél-Európa:Balkán-félsziget	110,912.00	köztársaság
7	15	ROMÁNIA	BUKAREST	Kelet-Európa	237,500.00	köztársaság

Feltételek megadásakor operátorokat használhatunk: =,>,<,<> vagy !=, >=, <=, BETWEEN, LIKE, IN.

Példa

Írassuk ki azon országok adatait, amik területe 1000 m^2 alatt van!

```sql
SELECT *
FROM orszagok
WHERE terulet < 1000;
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma
1	28	LIECHTENSTEIN	VADUZ	Közép-Európa:Alpok(törpeállam	160.00	hercegség
2	29	MÁLTA	VALLETTA	Európa:Földközi-tenger (szigetor	316.00	köztársaság
3	45	MONACO	MONACO	Európa:Francia-Riviéra (törpeálla	1.95	alkotmányos monarc
4	46	ANDORRA	ANDORRA LA VE	Európa:Pireneusok (törpeállam)	468.00	autonóm hercegség
5	70	VATIKÁN	VATIKÁN	Európa (városállam Rómánál)	0.44	teokratikus abszolút
6	134	TUVALU	FUNAFUTI	Óceánia:Melanézia:Ellice-szk.	26.00	alkotmányos monarc
7	135	TONGA	NUKU'ALOFA	Óceánia:Melanézia-Polinézia	748.00	alkotmányos királysá

Írassuk ki azon országok adatait, amik népessége 1 és 2 millió között van!

```sql
SELECT *
FROM orszagok
```

```
3. WHERE nepesseg BETWEEN 1000 AND 2000;
```

Eredmény

	fovaros	foldr_hely	terulet	allamforma	nepesseg
3	MASERU	Dél-Afrika	30,355.00	alkotmányos monarc	1,800
4	WINDHOEK	Dél-Afrika	823,168.00	köztársaság	1,770
5	LIBREVILLE	Közép-Nyugat-Afrika	267,667.00	elnöki köztársaság	1,208
6	MBABANE	Dél-Kelet-Afrika	17,364.00	alkotmányos monarc	1,080
7	BANJUL	Nyugat-Afrika	11,295.00	köztársaság	1,400
8	BISSAU	Nyugat-Afrika	36,125.00	köztársaság	1,285
9	PORT LOUIS	Indiai-óceán (Kelet-Afrika)	2,045.00	alkotmányos monarc	1,200

Írassuk ki azon országok adatait, ahol a pénznem euró, dollár vagy forint!

```
1. SELECT *
2. FROM orszagok
3. WHERE penznem IN ("euró","dollár","forint");
```

Eredmény

	capital	penznem	penzjel	valtopenz	telefon	gdp
6	WIEN (BECS)	euró	EUR	100 eurocent	43	30,180
7	ROMA	euró	EUR	100 eurocent	39	24,390
8	BRATISLAVA POZSONY	euró	EUR	100 eurocent	421	5,810
9	MOSZKVA	rubel	RUR	100 kopejka	7	2,910
10	ATHINE ATHENAI	euró	EUR	100 eurocent	30	15,060
11	BRUXELLES BRUSSZEL	euró	EUR	100 eurocent	32	28,800
12	AMSZTERDAM - HAGA	euró	EUR	100 eurocent	31	30,800

Írassuk ki azon országok adatait, amely nevében szerepel az ország szó!

```
1. SELECT *
2. FROM orszagok
3. WHERE orszag LIKE "%ország%";
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma
6	11	OLASZORSZÁG	RÓMA	Dél-Európa:Appennini-félsziget	301,252.00	köztársaság
7	12	MAGYARORSZÁG	BUDAPEST	Közép-Európa:Kárpát-medence	93,036.00	köztársaság
8	17	CSEHORSZÁG	PRÁGA	Közép-Európa	78,864.00	köztársaság
9	18	LENGYELORSZÁG	VARSÓ	Közép-Európa	312,677.00	köztársaság
10	19	OROSZORSZÁG	MOSZKVA	Eurázsia	17,075,400.00	szövetségi köztársas
11	20	GÖRÖGORSZÁG	ATHÉN	Dél-Európa:Balkán-félsziget	131,944.00	köztársaság
12	21	TÖRÖKORSZÁG	ANKARA	Eurázsia	780,576.00	köztársaság
13	31	ÍRORSZÁG	DUBLIN	Európa:Britt-szigetek	70,283.00	köztársaság

Írassuk ki azon országok adatait, amely nevének második betűje „a”!

```
1. SELECT *
2. FROM orszagok
3. WHERE orszag LIKE "_a%";
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma
6	35	JAPÁN	TOKIÓ	Ázsia:Távol-Kelet	372,769.00	alkotmányos monar
7	37	KANADA	OTTAWA	Észak-Amerika	9,976,139.00	szövetségi állam
8	59	MADAGASZKÁR	ANTANANARIVO	Kelet-Afrika (szigetország)	587,041.00	elnöki köztársaság
9	65	PARAGUAY	ASUNCION	Dél-Amerika	406,752.00	elnöki köztársaság
10	70	VATIKÁN	VATIKÁN	Európa (városállam Rómánál)	0.44	teokratikus abszolút
11	71	PAKISZTÁN	ISLAMABAD	Közép-Ázsia	803,943.00	köztársaság
12	73	BANGLADES	DHAKA	Ázsia:Hindusztáni-alföld	147,570.00	köztársaság
13	77	KAMBODZSA	PHNOM PENH	Ázsia:Indokinai-félsziget	181,035.00	köztársaság

Logikai operátorok

Logikai operátorok segítségével alakíthatjuk aki a logikai „és”, „vagy” és a „tagadás” műveletet.

AND

```
1. SELECT oszlop1, oszlop2, ...
2. FROM tablaNeve
3. WHERE feltétel1 AND feltétel2 AND feltétel3 ...;
```

OR

```
1. SELECT oszlop1, oszlop2, ...
2. FROM tablaNeve
3. WHERE feltétel1 OR feltétel2 OR feltétel3 ...;
```

NOT

```
1. SELECT oszlop1, oszlop2, ...
2. FROM tablaNeve
3. WHERE NOT feltétel;
```

Példa

Írassuk ki azon országok adatait, amely területe 30000 km^2 felett van és euro a pénzem!

```
1. SELECT *
2. FROM orszagok
3. WHERE penznem = "euro" AND terulet > 300000;
```

	id	orszag	fovaros	foldr_hely	terulet	allamforma
1	1	SPANYOLORSZÁG	MADRID	Dél-Európa:Ibériai-félsziget	504,782.00	alkotmányos monarchia
2	3	FRANCIAORSZÁG	PÁRIZS	Nyugat-Európa	547,026.00	köztársaság
3	7	FINNORSZÁG	HELSINKI	Észak-Európa	338,107.00	köztársaság
4	8	NÉMETORSZÁG	BERLIN	Nyugat-Európa	357,042.00	szövetségi köztársaság
5	11	OLASZORSZÁG	ROMA	Dél-Európa:Appennini-félsziget	301,252.00	köztársaság

Írassuk ki azon országok adatait, amely Európában vagy Amerikában találhatóak!

```
1. SELECT *
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Európa%" OR foldr_hely LIKE "%Amerika%";
```

	id	orszag	fovaros	foldr_hely	terulet	allamforma
27	29	MÁLTA	VALLETTA	Európa:Földközi-tenger (szigetor	316.00	köztársaság
28	30	CIPRUS	NICOSIA	Európa:Földközi-tenger (szigetor	9,251.00	köztársaság
29	31	ÍRORSZÁG	DUBLIN	Európa:Britt-szigetek	70,283.00	köztársaság
30	37	KANADA	OTTAWA	Észak-Amerika	9,976,139.00	szövetségi állam
31	38	MEXIKÓ	MEXIKÓVÁROS	Közép-Amerika	1,972,547.00	szövetségi köztársaság
32	39	KUBA	HAVANNA	Közép-Amerika (szigetország)	110,922.00	köztársaság
33	40	BRAZÍLIA	BRASÍLIA	Dél-Amerika	8,511,965.00	szövetségi köztársaság
34	41	ARGENTÍNA	BUENOS AIRES	Dél-Amerika	2,776,889.00	köztársaság
35	45	MONACO	MONACO	Európa:Francia-Riviéra (törpeálla	1.95	alkotmányos monarchia

A matematikában megszokott műveleti sorrend itt is érvényes. Természetesen zárójelekkel ez módosítható.

Írassuk ki azon országok adatait, amely félszigeten fekszenek és az államformája köztársaság vagy alkotmányos monarchia!

```
1. SELECT *
2. FROM orszagok
3. WHERE foldr_hely LIKE "%félsziget%" AND (allamforma = "köztársaság" OR allamforma =
   "alkotmányos monarchia");
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma
7	20	GÖRÖGORSZÁG	ATHÉN	Dél-Európa:Balkán-félsziget	131,944.00	köztársaság
8	24	ALBÁNIA	TIRANA	Dél-Európa:Balkán-félsziget	28,748.00	köztársaság
9	76	THAIFÖLD	BANGKOK	Ázsia:Indokinai-félsziget	513,115.00	alkotmányos monarchia
10	77	KAMBODZSA	PHNOM PENH	Ázsia:Indokinai-félsziget	181,035.00	köztársaság
11	78	VIETNAM	HANOI	Ázsia:Indokinai-félsziget	329,556.00	köztársaság
12	131	DÉL-KOREA	SZÖUL	Ázsia:Koreai-félsziget	98,484.00	köztársaság
13	145	KUVAIT	KUVAIT	Ázsia:Arab-félsziget	17,818.00	alkotmányos monarchia

Írassuk ki azon európai országok adatait, amely nem az eurót használják pénzként!

```
1. SELECT *
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Európa%" AND NOT penznem="euró";
```

Eredmény

	country	capital	penznem	penzjel	valtopenz
7	BULGARIA	SZOFIJA (SZOFIA)	leva	BGL	100 sztotinka
8	ROMANIA	BUCURESTI BUKAREST	lei	ROL	100 bani
9	CSEHORSZAG CSEHSZLOVAKIA	PRAHA (PRAGA)	cseh korona	CZK	100 haler
10	LENGYELORSZAG POLAND	WARSZAWA (VARSO)	zloty	PLZ	100 grosz
11	DANIA DENMARK	KOPPENHAGA	dán korona	DKK	100 öre
12	IZLAND ISLAND ICELAND	REYKJAVIK	izlandi korona	ISK	100 aurar
13	ALBANIA	TIRANA	lek	ALL	100 quindarka
14	LIECHTENSTEIN	VADUZ	svájci frank	CHF	100 rappen

Rendezés - ORDER BY

Az ORDER BY segítségével lehet rendezni a lekérdezés eredményét növekvő vagy csökkenő sorrendbe. Alapértelmezés a növekvő (ASC) sorrend. Csökkenő sorrend esetén a DESC szót kell használnunk.

Általános alak

```
1. SELECT oszlop1, oszlop2, ...
2. FROM tableNeve
3. ORDER BY column1, column2, ... ASC|DESC;
```

Példa

Írassuk ki európai országok adatait nevük szerint ábécé sorrendbe!

```
1. SELECT *
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Európa%"
4. ORDER BY orszag;
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma	nepesseg
1	24	ALBÁNIA	TIRANA	Dél-Európa:Balkán-	28,748.00	köztársaság	3,490
2	46	ANDORRA	ANDORRA LA VE	Európa:Pireneusok	468.00	autonóm hercegség	70
3	10	AUSZTRIA	BÉCS	Közép-Európa:Alpo	83,858.00	szövetségi köztársasá	8,130
4	182	BELARUSZ	MINSZK	Kelet-Európa	207,600.00	köztársaság	10,300
5	25	BELGIUM	BRÜSSZEL	Nyugat-Európa	30,519.00	alkotmányos monarc	10,300
6	177	BOSZNIA-HERC	SARAJEVO	Dél-Európa:Balkán	51,129.00	köztársaság	4,400
7	14	BULGÁRIA	SZÓFIA	Dél-Európa:Balkán-	110,912.00	köztársaság	7,900
8	30	CIPRUS	NICOSIA	Európa:Földközi-te	9,251.00	köztársaság	758
9	17	CSEHORSZÁG	PRÁGA	Közép-Európa	78,864.00	köztársaság	10,300
10	22	DÁNIA	KOPPENHÁGA	Nyugat-Európa:Jyll	43,075.00	alkotmányos monarc	5,300

Írassuk ki európai országok adatait államforma, majd azon belül nevük szerint ábécé sorrendbe!

```
1. SELECT *
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Európa%"
4. ORDER BY allamforma, orszag;
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma	nepesseg
1	25	BELGIUM	BRÜSSZEL	Nyugat-Európa	30,519.00	alkotmányos monarchia	10,300
2	22	DÁNIA	KOPPENHÁGA	Nyugat-Európa:Jyll	43,075.00	alkotmányos monarchia	5,300
3	26	HOLLANDIA	AMSZTERDAM	Nyugat-Európa	41,548.00	alkotmányos monarchia	16,100
4	45	MONACO	MONACO	Európa:Francia-Rivi	1.95	alkotmányos monarchia	30
5	4	NAGY-BRITANN	LONDON	Európa:Britt-szigete	244,046.00	alkotmányos monarchia	65,200
6	5	NORVÉGIA	OSLO	Észak-Európa:Skanc	324,219.00	alkotmányos monarchia	4,600
7	1	SPANYOLORSZ	MADRID	Dél-Európa:Ibériai-t	504,782.00	alkotmányos monarchia	42,700
8	6	SVÉDORSZÁG	STOCKHOLM	Észak-Európa:Skanc	449,964.00	alkotmányos monarchia	8,870
9	46	ANDORRA	ANDORRA LA VE	Európa:Pireneusok	468.00	autonóm hercegség	70
10	28	LIECHTENSTEIN	VADUZ	Közép-Európa:Alpo	160.00	hercegség	30

Írassuk ki európai országok adatait államforma szerint növekvő, majd azon belül népesség szerint csökkenő sorrendbe!

```
1. SELECT *
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Európa%"
4. ORDER BY allamforma, nepesseg DESC;
```

Eredmény

	id	orszag	fovaros	foldr_hely	terulet	allamforma	nepesseg
1	4	NAGY-BRITANN	LONDON	Európa:Britt-szigete	244,046.00	alkotmányos monarchia	65,200
2	1	SPANYOLORSZ	MADRID	Dél-Európa:Ibériai-t	504,782.00	alkotmányos monarchia	42,700
3	26	HOLLANDIA	AMSZTERDAM	Nyugat-Európa	41,548.00	alkotmányos monarchia	16,100
4	25	BELGIUM	BRÜSSZEL	Nyugat-Európa	30,519.00	alkotmányos monarchia	10,300
5	6	SVÉDORSZÁG	STOCKHOLM	Észak-Európa:Skanc	449,964.00	alkotmányos monarchia	8,870
6	22	DÁNIA	KOPPENHÁGA	Nyugat-Európa:Jyll	43,075.00	alkotmányos monarchia	5,300
7	5	NORVÉGIA	OSLO	Észak-Európa:Skanc	324,219.00	alkotmányos monarchia	4,600
8	45	MONACO	MONACO	Európa:Francia-Rivi	1.95	alkotmányos monarchia	30
9	46	ANDORRA	ANDORRA LA VE	Európa:Pireneusok	468.00	autonóm hercegség	70
10	28	LIECHTENSTEIN	VADUZ	Közép-Európa:Alpo	160.00	hercegség	30

Írassuk ki európai országok nevét, népességét, fővárosának lakosságát. Rendezzük a főváros adatai alapján csökkenően, majd azon belül népesség szerint növekvő sorrendbe!

```
1. SELECT orszag,nepesseg, nep_fovaros
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Európa%"
4. ORDER BY 3 DESC, nepesseg ASC;
```

Eredmény

	ABC orszag	123 nepesseg	123 nep_fovaros
1	FRANCIAORSZÁG	66,860	11,300
2	NAGY-BRITANNIA	65,200	7,200
3	NÉMETORSZÁG	82,400	5,900
4	SPANYOLORSZÁG	42,700	5,100
5	OLASZORSZÁG	60,600	3,600
6	GÖRÖGORSZÁG	11,000	3,300
7	UKRAJNA	49,000	2,800
8	PORTUGÁLIA	10,000	2,600
9	MAGYARORSZÁG	10,100	2,600
10	ROMÁNIA	22,410	2,200
11	LENGYELORSZÁG	38,600	2,200

Megjegyzés: a rendezések során használhatjuk az oszlopok sorszámát is.

LIMIT

A LIMIT segítségével megadhatjuk, hogy az eredményhalmaz hány rekordot tartalmazzon.

Általánosan

```
1.  SELECT oszlop1, oszlop2, …
2.  FROM tablaNeve
3.  WHERE feltételek
4.  LIMIT szam;
```

Példa

Írassuk ki a 3 legnagyobb lélekszámú európai ország nevét, népességét, fővárosának lakosságát.

```
1.  SELECT orszag,nepesseg, nep_fovaros
2.  FROM orszagok
3.  WHERE foldr_hely like "%Európa%"
4.  ORDER BY nepesseg DESC
5.  LIMIT 3;
```

	ABC orszag	123 nepesseg	123 nep_fovaros
1	NÉMETORSZÁG	82,400	5,900
2	FRANCIAORSZÁG	66,860	11,300
3	NAGY-BRITANNIA	65,200	7,200

Megjegyzés: a LIMIT nem veszi figyelembe, ha megegyezik az utolsó érték.

Listák részleges megjelenítésekor gyakran van szükségünk arra, hogy egy adott pozíciótól jelenítsünk valahány darab sort a lekérdezésünk eredményéből.

Példa

```
1.  SELECT *
2.  FROM orszagok
3.  LIMIT 20,10;
```

Eredmény

	id	ország	fővaros	földr_hely	terület	államforma
1	21	TÖRÖKORSZÁG	ANKARA	Eurázsia	780,576.00	köztársaság
2	22	DÁNIA	KOPPENHÁGA	Nyugat-Európa:Jylland	43,075.00	alkotmányos monarchia
3	23	IZLAND	REYKJAVIK	Európa:Atlanti-óceán (szigetország)	102,829.00	köztársaság
4	24	ALBÁNIA	TIRANA	Dél-Európa:Balkán-félsziget	28,748.00	köztársaság
5	25	BELGIUM	BRÜSSZEL	Nyugat-Európa	30,519.00	alkotmányos monarchia
6	26	HOLLANDIA	AMSZTERDAM	Nyugat-Európa	41,548.00	alkotmányos monarchia
7	27	LUXEMBURG	LUXEMBOURG	Nyugat-Európa	2,586.40	nagyhercegség
8	28	LIECHTENSTEIN	VADUZ	Közép-Európa:Alpok(törpeállam)	160.00	hercegség
9	29	MÁLTA	VALLETTA	Európa:Földközi-tenger (szigetország)	316.00	köztársaság
10	30	CIPRUS	NICOSIA	Európa:Földközi-tenger (szigetország)	9,251.00	köztársaság

SQL függvények

A MIN(),MAX() függvény visszaadja az adott oszlop legkisebb, illetve a legnagyobb értékét.

Általánosan

```
1. SELECT MIN(oszlop)
2. FROM tablaNeve
3. WHERE feltetel;
```

illetve,

```
1. SELECT MAX(oszlop)
2. FROM tablaNeve
3. WHERE feltetel;
```

Példa

Írassuk amilyen nagy területű legnagyobb amerikai ország!

```
1. SELECT MAX(terulet)
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Amerika%";
```

	max(terulet)
1	9,976,139.00

Gyakori tévedést mutat be a következő példa.

Írassuk ki a legnagyobb népességű ország népességét és nevét!

```
1. #1. lekérdezés
2. SELECT MAX(nepesseg),orszag
3. FROM orszagok
4. WHERE foldr_hely LIKE "%Amerika%";
5.
6. #2. lekérdezés
7. SELECT nepesseg,orszag
8. FROM orszagok
9. WHERE foldr_hely LIKE "%Amerika%"
10. ORDER BY nepesseg DESC;
```

Hasonlítsuk össze a két lekérdezés eredményét!

	MAX(nepesseg)	ország
1	325,200	KANADA

illetve,

	123 nepesseg	ABC orszag
1	325,200	AMERIKAI EGYESÜLT ÁLLAMOK
2	207,000	BRAZÍLIA
3	122,300	MEXIKÓ
4	44,200	KOLUMBIA
5	38,400	ARGENTÍNA
6	31,700	KANADA
7	27,100	PERU

Látható, hogy az első esetben félrevezető és egyben helytelen eredményhalmaz kapunk, hiszen a maximális érték nem Kanadához tartozik. Az is belátható, hogy a maximum több esetben is előfordulhat, így ezeket az eseteket a későbbiekben tanult egymásba ágyazott lekérdezésekkel oldhatjuk meg.

A COUNT(), AVG(), SUM() függvények segítségével darabszámot, átlagot illetve összeget határozhatjuk meg.

Általánosan

```
1.  SELECT COUNT(oszlop)
2.  FROM tablaNeve
3.  WHERE feltetel;
4.
5.  SELECT AVG(oszlop)
6.  FROM tablaNeve
7.  WHERE feltetel;
8.
9.  SELECT SUM(oszlop)
10. FROM tablaNeve
11. WHERE feltetel;
```

Példa

Írassuk ki az ázsiai országok népességét!

```
1.  SELECT SUM(nepesseg)* 1000
2.  FROM orszagok
3.  WHERE foldr_hely LIKE "%Ázsia%";
```

Eredmény

	123 sum(nepesseg)*1000
1	4,480,305,000

Mennyi az európai országok átlagos területe?

```
1.  SELECT AVG(terulet)
2.  FROM orszagok
3.  WHERE foldr_hely LIKE "%Európa%";
```

Eredmény

	123 AVG(terulet)
1	130,502.660000

Hány 20.000 főnél kevesebb lakosú ország van?

```
1. SELECT COUNT(id)
2. FROM orszagok
3. WHERE nepesseg<20;
```

	123 COUNT(id)
1	4

SQL rendszerek ezek mellett is sok függvényt biztosítanak. Csoportosíthatjuk felhasználásuk alapján: sztringkezelő, numerikus, dátumkezelő illetve egyéb függvényekre. MySQL rendszer függvényei.

Sztringkezelő függvények

ASCII, CHAR_LENGTH, CHARACTER_LENGTH, CONCAT, CONCAT_WS, FIELD, FIND_IN_SET, FORMAT, INSERT, INSTR, LCASE, LEFT, LENGTH, LOCATE, LOWER, LPAD, LTRIM, MID, POSITION, REPEAT, REPLACE, REVERSE, RIGHT, RPAD, RTRIM, SPACE, STRCMP, SUBSTR, SUBSTRING, SUBSTRING_INDEX, TRIM, UCASE, UPPER.

Numerikus függvények

ABS, ACOS, ASIN, ATAN, ATAN2, AVG, CEIL, CEILING, COS, COT, COUNT, DEGREES, DIV, EXP, FLOOR, GREATEST, LEAST, LN, LOG, LOG10, LOG2, MAX, MIN, MOD, PI, POW, POWER, RADIANS, RAND, ROUND, SIGN, SIN, SQRT, SUM, TAN, TRUNCATE.

Dátumkezelő függvények

ADDDATE, ADDTIME, CURDATE, CURRENT_DATE, CURRENT_TIME, CURRENT_TIMESTAMP, CURTIME, DATE, DATEDIFF, DATE_ADD, DATE_FORMAT, DATE_SUB, DAY, DAYNAME, DAYOFMONTH, DAYOFWEEK, DAYOFYEAR, EXTRACT, FROM_DAYS, HOUR, LAST_DAY, LOCALTIME, LOCALTIMESTAMP, MAKEDATE, MAKETIME, MICROSECOND, MINUTE, MONTH, MONTHNAME, NOW, PERIOD_ADD, PERIOD_DIFF, QUARTER, SECOND, SEC_TO_TIME, STR_TO_DATE, SUBDATE, SUBTIME, SYSDATE, TIME, TIME_FORMAT, TIME_TO_SEC, TIMEDIFF, TIMESTAMP, TO_DAYS, WEEK, WEEKDAY, WEEKOFYEAR, YEAR, YEARWEEK

Egyéb függvények

BIN, BINARY, CASE, CAST, COALESCE, CONNECTION_ID, CONV, CONVERT, CURRENT_USER, DATABASE, IF, IFNULL, ISNULL, LAST_INSERT_ID, NULLIF, SESSION_USER, SYSTEM_USER, USER, VERSION

Ezeket a függvényeket tesztelhetjük a SELECT segítségével.

Példa

```
1. SELECT NOW();
```

	NOW()
1	2019-02-13 01:17:15

```
1. SELECT UPPER("Minta János");
```

	ABC upper("Minta János")
1	MINTA JÁNOS

```
1. SELECT YEAR(NOW());
```

	123 YEAR(NOW())
1	2,019

```
1. SELECT CURRENT_USER();
```

	ABC current_user()
1	root@localhost

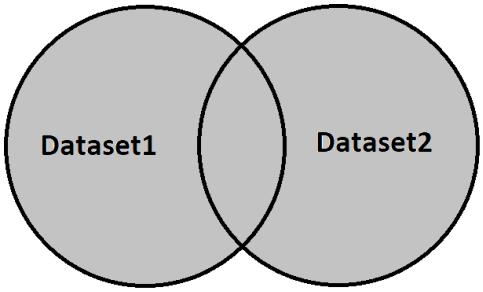
Ezek részletesebben tárgyalása egy későbbi leckében kerül sor.

Halmazműveletek

A halmazműveletek esetén a műveletben részvevő halamazoknak kompatibilisnek kell lennie.

Egyesítés – UNION

Az egyesítés (UNION) operátor használata esetén a két (vagy több) lekérdezés unióját kapjuk vissza eredményként. A többször szereplő sorok csak egyszer jelennek meg.



Általánosan

```
1. SELECT oszlop1, oszlop2, ...
2. FROM tablaNevek
3. [WHERE feltételek]
4. UNION [DISTINCT]
5. SELECT oszlop1, oszlop2, ...
6. FROM tablaNevek
7. [WHERE feltételek];
```

Példa

Írassuk ki az Afrikai országok és az elnöki köztársaságában lévő országok unióját!

```
1. SELECT orszag, foldr_hely
2. FROM orszagok
3. WHERE foldr_hely LIKE "%Afrika%"
4. UNION
5. SELECT orszag, foldr_hely
6. FROM orszagok
7. WHERE allamforma = "elnöki köztársaság"
8. ORDER BY orszag;
```

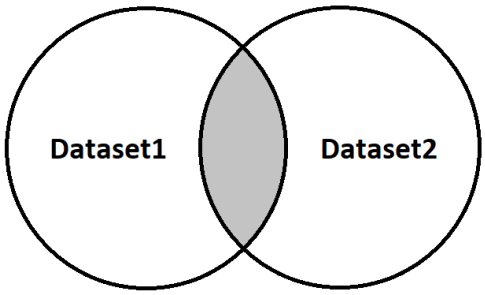
Eredmény

	orszag	foldr_hely
1	ALGÉRIA	Észak-Afrika
2	ANGOLA	Dél-Afrika
3	BENIN	Nyugat-Afrika
4	BISSAU-GUINEA	Nyugat-Afrika
5	BOLÍVIA	Dél-Amerika
6	BOTSWANA	Dél-Afrika:Kalahári-medence
7	BURKINA FASO	Nyugat-Afrika
8	BURUNDI	Közép-Kelet-Afrika
9	COMORE-SZIGET	Kelet-Afrika (Indiai-óceán)
10	COSTA RICA	Közép-Amerika
11	CSÁD	Közép-Afrika
12	DÉL-AFRIKAI KÖz	Dél-Afrika

Amennyiben az összes sorra szükségünk, akkor az UNION ALL operátort kell használnunk.

Metszet - INTERSECT

A lekérdezés eredményeképpen a halmazok metszetét kapjuk meg. Azaz megkapjuk azokat a rekordokat, amik mind a Dataset1, mind a Dataset2-ben szerepelnek. (An INTERSECT query returns the intersection of 2 or more datasets.)



Általánosan

```
SELECT oszlop1, oszlop2, ...
FROM tablaNevek
[WHERE feltételek]
INTERSECT
SELECT oszlop1, oszlop2, ...
FROM tablaNevek
[WHERE feltételek];
```

A MySQL-ben nem használhatjuk a INTERSECT operátort, így más megoldást kell használni. A megoldáshoz használjuk az IN operátort a következő alakban.

Példa

Írassuk ki az Európai országok és a köztársaságok metszetét!

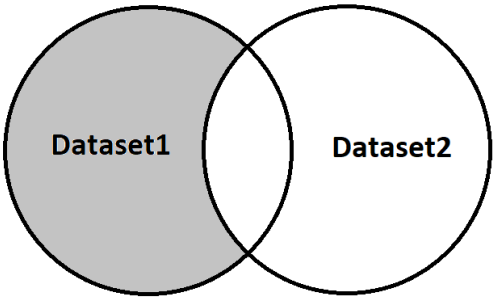
```
SELECT orszag
FROM orszagok
WHERE foldr_hely LIKE "%Európa%"
AND orszag IN (SELECT orszag
               FROM orszagok
               WHERE allamforma LIKE "%köztársaság%" );
```

Eredmény

	orszag
1	PORTUGÁLIA
2	FRANCIAORSZÁG
3	FINNORSZÁG
4	NÉMETORSZÁG
5	SVÁJC
6	AUSZTRIA
7	OLASZORSZÁG
8	MAGYARORSZÁG
9	SZERBIA
10	BULGÁRIA
11	ROMÁNIA

Különbség – MINUS operátor

A különbség (MINUS) operátor az első SELECT (Dataset1) összes olyan során adja meg, ami nem szerepel a második SELECT (Dataset2)-ben.



1. ábra MINUS operator

A MINUS operátor nem minden adatbázis-kezelő támogatja. Használható ORACLE alatt, illetve EXCEPT operátorként MS SQL, PostgreSQL, SQLite esetén.

Általánosan

```
1.  SELECT oszlop1, oszlop2, ...
2.  FROM tablaNevek
3.  [WHERE feltetelek]
4.  MINUS
5.  SELECT oszlop1, oszlop2, ...
6.  FROM tablaNevek
7.  [WHERE feltetelek];
```

A MySQL nem támogatja sem a MINUS, sem a EXCEPT operátor. Megvalósítható a későbbiekben részletezett LEFT JOIN segítségével.

```
1.  SELECT
2.      oszlop1, oszlop2, …
3.  FROM
4.      tablaNev_1
5.      LEFT JOIN tablaNev_2 ON kapcsolatFeltetel
6.  WHERE
7.      tablaNev _2.id IS NULL;
```

Csoportosítás

Csoportosítás során valamelyik oszlop, vagy oszlopok szerint csoportosítva végezhetünk el műveleteket az aggregáló (COUNT, MAX, MIN, SUM, AVG) függvények használatával. Itt az

adott oszlop különböző értékei alapján történik a csoportosítás és ezen belül hajtódik végre az adott számítás.

Általános alak

```
1. SELECT oszloNev(ek)
2. FROM tablaNev
3. WHERE feltétel
4. GROUP BY oszlopNev(ek)
5. ORDER BY oszlopNev(ek);
```

Példa

Írassuk ki az iskola adatbázisból az osztályonkénti létszámokat!

Megoldás

```
1. SELECT
2.     osztaly,
3.     COUNT(NEV) AS fő
4. FROM
5.     tanulo NATURAL JOIN osztaly
6. GROUP BY
7.     OSZTALY;
```

Eredmény

	osztaly	fő
1	1/A	17
2	1/B	25
3	1/C	26
4	1/D	31
5	1/E	25
6	2/A	16
7	2/B	21
8	2/C	25
9	2/D	32
10	2/E	30
11	3/A	23

Megjegyzés: a COUNT esetén a sorok számlálása történik. Így ha felvittünk egy magántanulót, akkor a

```
1. SELECT
2.     COUNT(nev)
3. FROM
4.     tanulo;
```

497-et ad eredményként. De ha

```
1. SELECT
2.     COUNT(osztaly)
3. FROM
4.     tanulo;
```

adunk ki és a magántanulónak nincs kitöltve az osztály, akkor ez 496-t ad vissza. Gyakran használjuk a COUNT(*), illetve helyette a COUNT(1)-t.

Írassuk ki, hogy hány tanulónak van 1,2,3 … testvére!

Megoldás

```
1.  SELECT
2.      testver, COUNT(1) AS fő
3.  FROM
4.      tanulo
5.  GROUP BY
6.      testver;
```

Eredmény

	testver	fő
1	0	127
2	1	110
3	2	126
4	3	130
5	4	3
6	6	1

Látható, hogy az eredményben szerepel a 0 testvéresek száma. Ezt ki kell szűrni.

Megoldás

```
1.  SELECT
2.      testver, COUNT(1) AS fő
3.  FROM
4.      tanulo
5.  WHERE
6.      testver != 0
7.  GROUP BY
8.      testver;
```

Példa

Írassuk ki azon osztályok osztályfőnökei, ahol az osztálylétszám 25 felett van. A lekérdezés eredményében jelenjen meg az osztály, az osztályfőnök neve és a létszám. Az létszám szerint rendezzük csökkenő sorrendbe, azon belül pedig ábécébe az osztályfőnök neve alapján!

Megoldás

Induljunk ki a már tesztelt lekérdezésből.

```
1.  SELECT
2.      osztaly, COUNT(NEV) AS fő
3.  FROM
4.      tanulo NATURAL JOIN osztaly
5.  GROUP BY
6.      OSZTALY;
```

Módosítsuk a feladatnak megfelelően.

```
1.  SELECT
2.      OSZTALY, OSZT_FONOK,    COUNT(1) AS fő
3.  FROM
4.      iskola.tanulo NATURAL JOIN iskola.osztaly
5.  GROUP BY
6.      OSZTALY, OSZT_FONOK;
```

Az érezhető, hogy itt másképpen kell a feltételt megadnunk, mint az eddigi feltételeket, hiszen itt az csoportosítás során létrejött információra vonatkozik a feltétel. Ekkor kell használnunk a HAVING-ot.

Általános alak

```
1.  SELECT oszlop(ok)
2.  FROM tablaNev
3.  WHERE feltel(ek)
4.  GROUP BY oszlop(ok)
5.  HAVING feltetel(ek)
6.  ORDER BY oszlop(ok);
```

Megoldás

```
1.  SELECT
2.      OSZTALY, OSZT_FONOK,    COUNT(1) AS fő
3.  FROM
4.      iskola.tanulo NATURAL JOIN iskola.osztaly
5.  GROUP BY
6.      OSZTALY, OSZT_FONOK
7.  HAVING
8.      COUNT(1)>25;
```

Már csak a rendezést kell kialakítani a feladatnak megfelelően.

```
1.  SELECT
2.      OSZTALY, OSZT_FONOK,    COUNT(1) AS fő
3.  FROM
4.      iskola.tanulo NATURAL JOIN iskola.osztaly
5.  GROUP BY
6.      OSZTALY, OSZT_FONOK
7.  HAVING
8.      COUNT(1)>25
9.   ORDER BY 3 DESC, OSZT_FONOK ;
```

A rendezés esetén az rövidítésképpen a 3. oszlopra hivatkoztunk.

Eredmény

OSZTALY	OSZT_FONOK	fő
4/A	Kemény Irma	35
2/D	Tomka Béla	32
1/D	Hévízi Alfonz	31
2/E	Joós Károlyné	30
3/D	Hegedűs Ferenc	29
1/C	Kovács Jenőné	26

Megkötések

SQL-ben a megkötések egy adattábla oszlopaira, illetve adataira vonatkoznak.

Általános alak

```
1.  CREATE TABLE tablaNev (
2.      oszlop1 adattipus constraint,
3.      oszlop2 adattipus constraint,
4.      oszlop3 adattipus constraint,
5.      ....
6.  );
```

A következő megkötéseket használhatjuk.

- NOT NULL - Biztosítja, hogy egy oszlop ne legyen NULL érték.
- UNIQUE - Biztosítja, hogy az oszlop minden értéke más legyen.
- PRIMARY KEY – Az előző kettő kombinációja. A táblázat minden sorát egyedileg azonosítja.
- FOREIGN KEY - Egyedülállóan azonosítja a sorokat / rekordokat egy másik táblázatban.
- CHECK - Biztosítja, hogy az oszlop minden értéke megfelel egy adott feltételnek.
- DEFAULT - Egy oszlop alapértelmezett értékének beállítása, ha nincs érték megadva.

A megszorítások lehetnek oszlopszintűek, illetve táblaszintűek. A táblaszintű beállításakor általában több oszlopot érint a megkötés. A táblaszíntű feltételeket a mezők felsorolása után tehetjük meg, illetve itt is megadhatjuk utólag is az ALTER TABLE segítségével.

Példa - NOT NULL

```
1.  CREATE TABLE tanulok (
2.      ID INT NOT NULL,
3.      Nev VARCHAR(255) NOT NULL,
4.      SzuletesiHely VARCHAR(255) NOT NULL,
5.      Kor INT
6.  );
```

Lekérdezhetjük a létrehozott szerkezetet.

```
1.  DESC tanulok;
```

	Field	Type	Null	Key	Default	Extra
1	ID	int(11)	NO		[NULL]	
2	Nev	varchar(255)	NO		[NULL]	
3	SzuletesiHely	varchar(255)	NO		[NULL]	
4	Kor	int(11)	YES		[NULL]	

Utólag is módosíthatunk a megkötéseken.

```
2.  ALTER TABLE tanulok
3.  MODIFY Kor INT NOT NULL;
```

Ekkor újra ellenőrizve a szerkezetet.

	Field	Type	Null	Key	Default	Extra
1	ID	int(11)	NO		[NULL]	
2	Nev	varchar(255)	NO		[NULL]	
3	SzuletesiHely	varchar(255)	NO		[NULL]	
4	Kor	int(11)	NO		[NULL]	

Példa - UNIQUE

```
1.  CREATE TABLE tanulok2 (
2.          ID INT NOT NULL UNIQUE,
3.          Nev VARCHAR(255) NOT NULL,
4.          SzuletesiHely VARCHAR(255) NOT NULL,
5.          Kor INT
```

```
6.  );
```

Utólag

```
1.  ALTER TABLE tanulok
2.  MODIFY UNIQUE(ID);
```

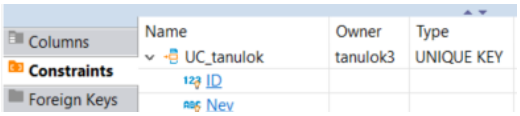
Táblaszinten

```
1.  CREATE TABLE tanulok3 (
2.          ID INT NOT NULL,
3.          Nev VARCHAR(255) NOT NULL,
4.          SzuletesiHely VARCHAR(255) NOT NULL,
5.          Kor INT,
6.      CONSTRAINT UC_tanulok UNIQUE (ID,Nev)
7.  );
```

A szerkezet ekkor.

	Field	Type	Null	Key	Default	Extra
1	ID	int(11)	NO	PRI	[NULL]	
2	Nev	varchar(255)	NO	PRI	[NULL]	
3	SzuletesiHely	varchar(255)	NO		[NULL]	
4	Kor	int(11)	YES		[NULL]	

A táblaszerkezete ellenőrizve a DBeaver segítségével.



Példa - PRIMARY KEY

```
1.  CREATE TABLE tanulok4 (
2.          ID INT PRIMARY KEY,
3.          Nev VARCHAR(255) NOT NULL,
4.          SzuletesiHely VARCHAR(255) NOT NULL,
5.          Kor INT
6.  );
```

Utólag

```
1.  ALTER TABLE tanulo4
2.  ADD PRIMARY KEY (ID);
```

Táblaszinten

```
1.  CREATE TABLE tanulok4 (
2.          ID INT NOT NULL,
3.          Nev VARCHAR(255) NOT NULL,
4.          SzuletesiHely VARCHAR(255) NOT NULL,
5.          Kor INT,
6.          CONSTRAINT PK_tanulo PRIMARY KEY (ID,Nev)
7.  );
```

Ellenőrizve

	Field	Type	Null	Key	Default	Extra
1	ID	int(11)	NO	PRI	[NULL]	
2	Nev	varchar(255)	NO	PRI	[NULL]	
3	SzuletesiHely	varchar(255)	NO		[NULL]	
4	Kor	int(11)	YES		[NULL]	

Programból

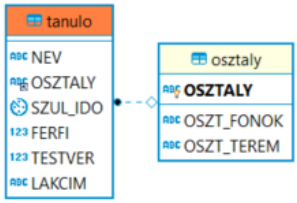
Példa - FOREIGN KEY

Külső kulcsot az InnoDB tárolási módon keresztül támogatja a MySQL.

Tekintsük a kapcsolatot az iskolai adatbásunkra tekintve.

```
1.  CREATE TABLE tanulo (
2.    NEV VARCHAR(35) ,
3.    OSZTALY VARCHAR(3) ,
4.    SZUL_IDO DATETIME ,
5.    FERFI TINYINT(1) ,
6.    TESTVER DOUBLE ,
7.    LAKCIM VARCHAR(50) ,
8.    CONSTRAINT tan_oszt_fk FOREIGN KEY (OSZTALY) REFERENCES osztaly(OSZTALY)
9.  ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Ekkor a E-R Diagram



Példa - CHECK

Középiskolai tanulók esetén ellenőrizhetjük, hogy az alsó határnak (14 év) megfelel-e a bevitt adat.

```
1.  CREATE TABLE tanulok5 (
2.          ID INT PRIMARY KEY,
3.          Nev VARCHAR(255) NOT NULL,
4.          SzuletesiHely VARCHAR(255) NOT NULL,
5.          Kor INT CHECK (kor >= 14)
6.  );
```

Utólag is létrehozhatjuk.

```
1.  ALTER TABLE tanulok5
2.  ADD CHECK (kor >= 14);
```

Megadhatunk több oszlopra vonatkozó feltételt is (azaz táblaszinten).

```
1.  CREATE TABLE tanulok6 (
2.          ID INT PRIMARY KEY,
3.          Nev VARCHAR(255) NOT NULL,
```

```
4.          SzuletesiHely VARCHAR(255) NOT NULL,
5.          Kor INT,
6.      CONSTRAINT CHK_tanulo CHECK (kor>=18 AND SzuletesiHely='Székesfehérvár')
7.  );
```

DEFAULT

Készítsünk olyan táblát, amelynek oszlop, oszlopai automatikus alapértéket vesznek fel, ha nincs megadva az adatbevitel esetén.

```
1.  CREATE TABLE tanulok7 (
2.          ID INT PRIMARY KEY,
3.          Nev VARCHAR(255) NOT NULL,
4.          SzuletesiHely VARCHAR(255) DEFAULT "Székesfehérvár",
5.          Kor INT,
6.          Beiratkozas TIMESTAMP DEFAULT CURRENT_TIMESTAMP
7.  );
```

Mint látható, megadhatunk konkrét értéket is, illetve használhatjuk a függvényeket is.

Ellenőrizve

```
1.  INSERT INTO tanulok7(ID, Nev) VALUES (1111,"Kiss Péter");
2.  SELECT * FROM tanulok7;
```

	ID	Nev	SzuletesiHely	Kor	Beiratkozas
1	1111	Kiss Péter	Székesfehérvár	[NULL]	2019-04-19 20:39:16

Utólag is megadhatjuk a DEFAULT megkötést, illetve törölhetjük is.

```
1.  ALTER TABLE tanulok7
2.  ALTER SzuletesiHely SET DEFAULT 'Székesfehérvár';
3.
4.  ALTER TABLE tanulok7
5.  ALTER SzuletesiHely DROP DEFAULT;
```

Allekérdezések / egymásba ágyazott lekérdezések

SQL lekérdezéseink eredménye szintén egy tábla, ami akár állhat 1 sorból és 1 oszlopból. Mivel az utasításokban is táblázatokat, értékeket használhatunk, így akár használhatunk egy „másik” SQL utasítás eredményeként keletkezett eredményhalmazt is. Ebben az esetben allekérdezésről (subquery) beszélünk. Az allekérdezéseket zárójelek közé kell írnunk.

Példa

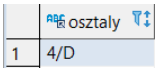
Írassuk ki Minarik Orsolyának az osztálytársait!

Megoldás

Elsőként írassuk ki, hogy melyik osztályba jár Minarik Orsolya!

```
1.  SELECT osztaly
2.  FROM tanulo
3.  WHERE nev = "Minarik Orsolya";
```

Eredmény



	osztaly
1	4/D

Ennek ismeretében készítsük el a lekérdezést az osztálytársakra.

```
1.  SELECT  nev
2.  FROM tanulo
3.  WHERE osztaly = "4/D";
```

Természetesen a szemmel kikeresés nem a jó megoldás, de elvezet bennünket a megoldáshoz, hiszen a „4/D” helyére bemásoljuk a lekérdezést és megkapjuk a megoldást.

```
1.  SELECT  nev
2.  FROM tanulo
3.  WHERE osztaly = (
4.          SELECT osztaly
5.          FROM tanulo
6.          WHERE
7.          nev = "Minarik Orsolya"
8.          )
```

Már csak annyi marad hátra, hogy az osztálytársak között ne jelenítsük meg Minarik Orsolyát!

```
1.  SELECT  nev
2.  FROM tanulo
3.  WHERE osztaly = (
4.          SELECT osztaly
5.          FROM tanulo
6.          WHERE
7.          nev = "Minarik Orsolya"
8.          )
9.      AND nev != "Minarik Orsolya"
10. ORDER BY nev;
```

Eredmény

	nev
1	Bacsa Brúnó
2	Balogh Alida
3	Baranyai Tódor
4	Busa Ipoly
5	Darida Mózes
6	Dócs Ádám
7	Gyémánt Nárcisz
8	Gyivicsán Pál
9	Kecskeméti Gedeon
10	Kiss Róbert
11	Kőris Péter
12	Lőrincz Félix
13	Monori Domonkos
14	Nyíri Fanni
15	Ormos Tódor
16	Pápai Hugó
17	Paszt Mária
18	Polgár Kinga
19	Rajnai Bánk
20	Satyinszlai Amália
21	Szekeres Helánia
22	Turú Jusztin
23	Velcsov Anita

Abban az esetben ha a belső lekérdezésünknek több eredménye is van, akkor az IN operátort kell használnunk.

EXISTS

Az EXISTS operátort arra használhatjuk, hogy teszteljük az allekérdezés eredményhalmazában lévő rekordok létezését.

Általános alak

```
1. SELECT oszlop(ok)
2. FROM tablaNev
3. WHERE EXISTS
4. (SELECT oszlopNev FROM tablaNev WHERE feltétel);
```

Példa

Tekintsük megint az iskola adatbázisunkat.

```
1. SELECT  COUNT(NEV)
2. FROM    tanulo
3. WHERE   EXISTS (
4.         SELECT nev
5.         FROM    tanulo
6.         WHERE   osztaly LIKE "1%"
7. );
```

Ekkor a lekérdezés 497-et ad eredményként, hiszen van elsős tanuló, Azonban, ha

```
1. SELECT  COUNT(NEV)
2. FROM    tanulo
3. WHERE   EXISTS (
```

```
4.          SELECT nev
5.          FROM    tanulo
6.          WHERE   osztaly LIKE "10%"
7.  );
```

lekérdezést teszteljük, akkor 0-t kapunk eredményként, hiszen nincs 10-es tanuló az adatbázisban.

Any, All

Az ANY és ALL operátorokat WHERE vagy HAVING záradékkal használhatjuk. Az ANY operátor akkor tér vissza igazra, ha az allekérdezés értékeinek valamelyike megfelel a feltételnek. Az ALL operátor akkor tér vissza, ha az allekérdezés összes értéke megfelel a feltételnek.

Általános alak - ANY

```
1.  SELECT oszlop(ok)
2.  FROM tablaNev
3.  WHERE oszlopNev operator ANY
4.  (SELECT oszlopNev FROM tablaNev WHERE feltetel);
```

Általános alak - ALL

```
5.  SELECT oszlop(ok)
6.  FROM tablaNev
7.  WHERE oszlopNev operator ALL
8.  (SELECT oszlopNev FROM tablaNev WHERE feltetel);
```

Operátor lehet: =, <>, !=, >, >=, <, <=.

Példa – ALL

Írassuk ki azokat, akiknek több testvére van, mint a 4/D tanulók közül bárkinek!

```
1.  SELECT NEV, TESTVER
2.  FROM TANULO
3.  WHERE TESTVER > ALL(
4.          SELECT TESTVER
5.          FROM TANULO
6.          WHERE OSZTALY="4/D"
7.  );
```

Eredmény

	NEV	TESTVER
1	Győri Elemér	4
2	Kolarovszki Sándor	6
3	Barra Adrienn	4
4	Tavasi Tekla	4

Példa – ANY

Írassuk ki azon tanuló adatait, akiknek az osztályfőnökének neve T-vel kezdődik!

```
1.  SELECT NEV, OSZTALY
2.  FROM TANULO
```

```
3.  WHERE OSZTALY = ANY(
4.          SELECT osztaly
5.          FROM osztaly
6.          WHERE OSZT_FONOK like "T%"
7.  );
```

Eredmény

	NEV	OSZTALY
1	Gyenes Boglárka	2/B
2	Mahler Jeromos	2/B
3	Papdi Kata	2/D
4	Gyovai Csenge	2/D
5	Guba Levente	2/D
6	Orudzsev Lukrécia	2/B
7	Pölös Konrád	2/D