ADATBÁZISOK I. ELŐADÁS ÉS GYAKORLAT JEGYZET

Szerkesztette: *Balogh Tamás*

2013. március 31.

Ha hibát találsz, kérlek jelezd a *info@baloghtamas.hu* e-mail címen!

Adatok:

Név: Brányi László **E-mail:** branyi@akármi.hu/.com

Webolal http://www.branyi.inf.elte.hu/branyi/~ora**Követelmények:**2 ZH, mindkettőt lehet

1. Gyakorlat

Relációs adatmodell

Reláció

Reláció a $D_1 \times D_2 \times \cdots \times D_n$ direkt-szorzat bármely részhalmazát. Jelöljük R-rel! A gyakorlatban kétdimenziós táblákal jelöljük a relációkat. Egy konkrét táblázatot a *reláció előfordulásának/példányának* nevezünk.

Sorok

Sor: A reláció egy eleme a táblázat egy sora. A reláció olyan táblázatnak tekinthető, melynek nem lehetnek azonos sorai. A sorrendjük mindegy.

Attribútumok

Attribútum: Az értéktartományok elemeit felvevő jellemzőket attribútumoknak nevezzük. Az attribútumok a relációk első sorában, ún. fejrészében találhatók. Ezek tehát a relációban szereplő oszlopoknak a nevei, melyek általában megadják az abban az oszlopban lévő adatok jelentését. A reláción belül ezeknek a neveknek egyedieknek kell lenniük, de más relációk tartalmazhatnak azonos nevű oszlopokat.

Sémák

Séma: A reláció neve és a reláció attribútumainak a halmaza együttesen alkotja a reláció sémáját. A reláció sémáját a reláció nevével és az attribútumainak zárójelek közötti felsorolásával adjuk meg. Pl.: R(A,B,C).

Komponensek

Komponens: Sor egy elemét komponensnek nevezzük. A reláció minden egyes fejlécben lévő attribútumához tartozik a sorban egy komponens.

Kulcsok

Kulcs: Az attribútumok egy halmaza egy kulcsot alkot egy relációra nézve, ha a reláció bármely elıfordulásában nincs két olyan sor, amelyek a kulcs összes attribútumának értékein megegyeznének.

Relációs algebra alapműveletei

Unió

$R \cup S$ azon sorok halmaza amelyek vagy R-ben vagy Sben vannak. Egy elem az eredményben csak egyszer szerepel.
R, S relációk sémájának ugyanazokat az attribútumhalmazokat kell tartalmaznia.
R, s relációk oszlopait rendezni kell úgy, hogy az R reláció i-edik attribútumának a neve megegyezzen az S reláció i-edik attribútumával, vagyis az attribútumok sorrendje azonos legyen mindkét reláció esetén.

Különbség

$R - S$ azon sorok halmaza amelyek R-benne vannak, de S-ben nincsenek. ($R - S \neq S - R$).
R, S relációk sémájának ugyanazokat az attribútumhalmazokat kell tartalmaznia.
R, S relációk oszlopait rendezni kell úgy, hogy az R reláció i-edik attribútumának a neve megegyezzen

az S reláció i-edik attribútumával, vagyis az attribútumok sorrendje azonos legyen mindkét reláció esetén.

Vetítés

Vetítés, másnéven projekció. Adott relációt vetít le az alsó indexben szereplő attribútumokra. (attribútumok számát csökkentik). Pl.: $\Pi_{\mathrm{A}}(R)$.
Ha az attribútumok értékei megegyeznek, akkor ezek az eredményrelációban csak egyszer fordulnak elő, hiszen a relációt sorok halmazaként értelmeztük.

2. Gyakorlat

Szelekció/szűrés/kiválasztás

$\sigma_{\mathrm{felt}}(R)$, pl.: $\sigma_{A=5}(R)$
Szelekcióban **NEM** lehet más relációs algebrai kifejezéseket rakni(kivéve $\wedge, \vee, \neg$). Balról jobbra zárójelezzük.
Kommutatív: $\sigma_{\mathrm{felt}_1}(\sigma_{\mathrm{felt}_2}(R))) = \sigma_{\mathrm{felt}_2}(\sigma_{\mathrm{felt}_1}(R)) = \sigma_{\mathrm{felt}_1 \wedge \mathrm{felt}_2}(R)$

Natural Join

E művelet használatával R és S relációknak csak azokat a sorait párosítjuk össze, amelyek értékei megegyeznek az R és az S sémájának összes közös attribútumán.
Legyen $R(A_1, \ldots, A_k, B_1, \ldots B_n), S(B_1, \ldots, B_n C_1 \ldots C_m)$ két séma.
$R \bowtie S$ típusa $(A_1, \ldots A_k, B_1, \ldots B_n, C_1, \ldots, C_m)$ vagyis a két attribútumhalmaz uniója.
Relációs algebrában nem változik az oszlopok sorrendje.
A természetes összekapcsolás kifejezhető más műveletek segítségével:$R \bowtie S = \Pi_L(\sigma_C(R \times S))$
c feltétel: $R.A_1 = S.A_1 \wedge ...R.A_n = S.A_n \quad L$: R séma + (S séma - R séma)
Ha R, S sémái megegyeznek, akkor $R \bowtie S = R \cap S$.
Ha R, S sémáiban nincs közös attribútum, akkor $R \bowtie S = R \times S$.

Átnevezés

Szükség lehet egy adott relációnak vagy a reláció attribútumainak átnevezésére: $\rho_{S(B_1,\ldots,B_k)}(R(A_1, \ldots, A_k))$.
Lehetőség van csak a relációt átnevezni: $\rho_S(R)$.

Relációs algebra hat alapművelete

- $R \cup S$ - unió
- $R - S$ - különbség
- $\Pi_{\mathrm{Lista}}(R)$ - vetítés
- $\sigma_{\mathrm{felt}}(R)$ - kiválasztás
- $R \bowtie S$ - natural join
- $\rho_{S(B_1,\ldots,B_k)}(R(A_1, \ldots, A_k))$ - átnevezés

Ebből a minimális készletből bármelyiket elhagyva az a többivel nem fejezhető ki.

További műveletek

Metszet

$R \cap S$ azon sorok halmaza amelyek R-ben is, és Sben is benne vannak. Egy elem az eredményben csak egyszer szerepel.
Kifejehezető a különbség segítségével: $R \cap S = R - (R - S)$.
R, S relációk sémájának ugyanazokat az attribútumhalmazokat kell tartalmaznia.
R, s relációk oszlopait rendezni kell úgy, hogy az R reláció i-edik attribútumának a neve megegyezzen

az S reláció i-edik attribútumával, vagyis az attribútumok sorrendje azonos legyen mindkét reláció esetén.

Descartes-szorzat

"Minden mindennel"
Descartes-szorzat esetén nem fontos az attribútumok egyenlősége. A két vagy több reláció azonos nevű attribútumait meg kell külölönböztetni egymástól: R=(A,B) S=(B,C,D)
R × S = A, R.B, S.B, C, D

Théta-összekapcsolás/Theta Join

$R \bowtie_c S = \sigma_c(R \times S)$
Ahol c egyszerű aritmetikai összehasonlítás, melyre $c = A_j \circ B_i$ alakú, ahol $\circ \in \{>, \geq, <, \leq, =, \neq\}$.
Észrevétel: $R \bowtie_c S = \sigma_c(R \times S)$.

Félig-összekapcsolás

...

Külső összekapcsolás/Outer join

A természetes összekapcsolás olyan kiterjesztése, amely figyelembe veszi a (pár nélküli) „lógó sorokat". A NULL használata kivezet a relációs algebrából, ezért az SQL-nél és a kiterjesztett relációs algebránál fogunk majd a külsı összekapcsolásokra késıbb visszatérni.

Osztás

$R \div S$
$R(A_1, \ldots, A_n, B_1, \ldots, B_m), S(B_1, \ldots, B_m)$ relációk, azaz S minden attribútuma benne van R attribútumhalmazában. Az $R \div S$ megadja azon $A_1, \ldots, A_n$ attribútumú t sorok halmazát, amelyekre igaz, hogy az S reláció minden s sorára a ts sor benne van az R relációban. Ez kifejezhető relációs algebrában.
Tehát az osztás egy speciális lekérdezésre adja meg a választ: keressük az R reláció azon sorait, amelyek mellett az S reláció mindegyik sora előfordul.
Az osztás műveletnek van egy fontos előfeltétele, mégpedig az, hogy csak olyan relációkra alkalmazható, amelyekre igaz az, hogy a második reláció S mindegyik attribútuma szerepel az első R relációban.
Az osztás művelete kifejezhető más relációs algebrai műveletekkel:

$$R \div S = \Pi_L(R) - \Pi_L((\Pi_L(R) \times (S) - R)$$

ahol L egy olyan attribútumlista, amelynek elemei az R reláció azon attribútumai, amelyek nincsenek benne az S relációban.
Levezetése:

- Képezzük az összes lehetséges sort!
$$\Pi_L(R) \times S$$
- Mely sorok nincsenek benne?
$$\Pi_L(R) \times S - R$$
- Ebben olyan sorok vannak, amelyek nem jók, ennek vesszük L-re a vetületét - megkapjuk a rossz sorokat.
$$\Pi_L(\Pi_L(R) \times S - R)$$
- A jó eredményre jutáshoz az összesből kivonjuk a rosszakat.
$$\Pi_L(R) - \Pi_L(\Pi_L(R) \times S - R)$$

Feladat

Táblázat: SZ(N,GY) szereti: név, gyümölcs

N	GY
MM	málna
MM	méz
Füles	körte
Malacka	méz
Malacka	málna
Malacka	körte
Kanga	banán
Tigris	méz

1, **Kérdés:** Melyek azok a gyümölcsök, amelyeket 'Micimackó' szeret?
Válasz: $\Pi_{GY}(\sigma_{N='MM'}(SZ))$

2, **Kérdés:** Melyek azok a gyümölcsök, amelyeket 'Micimackó' *nem* szeret? (de valaki más igen)
Válasz: $\Pi_{GY}(SZ) - \Pi_{GY}(\sigma_{N='MM'}(SZ))$

3, **Kérdés:** Melyek azok a gyümölcsök, amelyeket valaki szeret és nem csak egyedül Micimackó?
Válasz: $\Pi_{GY}(\sigma_{N\neq'MM'}(SZ))$ De relációs algebrában nem szeretjük a nem egyenlőt, így átírjuk
Másik megoldás: $\Pi_{GY}(SZ - \sigma_{N='MM'}(SZ))$

4, **Kérdés:** Kik azok akik *legalább* azokat a gyümölcsöket szeretik, mint Micimackó?
Válasz: $SZ \div \Pi_{GY}(\sigma_{N='MM'}(SZ))$
Másik megoldás: $X := \Pi_N(SZ) - \Pi_N(\Pi_N(SZ) \times \Pi_{GY}(\sigma_{N='MM'}(SZ)) - SZ)$

5, **Kérdés:** Kik azok, aki *legfeljebb* azokat a gyümölcsöket szeretik, mint Micimackó?
Válasz: $Y := \Pi_N(SZ) - \Pi_N(SZ - (\Pi_N(SZ) \times \Pi_{GY}(\sigma_{N='MM'}(SZ))))$

6, **Kérdés:** Kik azok, akik *pontosan* azokat a gyümölcsöket szereti, mint Micimackó?
Válasz: $X \cap Y$

7, **Kérdés:** Kik szeretik az almát?
Válasz: $\Pi_N(\sigma_{GY='alma'}(SZ))$

8, **Kérdés:** Kik *nem* szeretik az almát (de valami mást igen)?
Válasz: $\Pi_N(SZ) - \Pi_N(\sigma_{GY='alma'}(SZ)$

9, **Kérdés:** Kik azok, akik szeretnek legalább egy almán kívüli gyümölcsöt?
Válasz: $\Pi_N(SZ) - \Pi_N(\sigma_{GY='alma'}(SZ))$

10, **Kérdés:** Kik szeretik az almát és a diót is?
Válasz: $\Pi_N(\sigma_{\text{GY} =\text{'alma'}}(SZ)) \cap \Pi_N(\sigma_{\text{GY} = \text{'dió'}}(SZ))$

11, **Kérdés:** Kik szeretik vagy az almát vagy a diót?
Válasz: $\Pi_N(\sigma_{\text{GY} =\text{'alma'}}(SZ)) \cup \Pi_N(\sigma_{\text{GY} = \text{'dió'}}(SZ))$

12, **Kérdés:** Kik szeretik az almát, de a diót nem?
Válasz: $\Pi_N(\sigma_{\text{GY} =\text{'alma'}}(SZ)) - \Pi_N(\sigma_{\text{GY} = \text{'dió'}}(SZ))$

Amit a relációs algebrában nem lehet

- Számolni. Pl.: (2a = b).
- Eredmény sorrendjét változtatni. Pl.: abc sorrendbe.
- Összesíteni
- Módosítani
- Rekurzív(családfa)

3. Gyakorlat

Feladat

Táblázat: SZ(N,GY) szereti: név, gyümölcs

N	GY
MM	málna
MM	méz
Füles	körte
Malacka	méz
Malacka	málna
Malacka	körte
Kanga	banán
Tigris	méz

1, **Kérdés:** Kik szeretnek legalább kétféle gyümölcsöt?
Válasz: $Y := \Pi_{SZ1.N}(\sigma_{SZ1.GY \neq SZ2.GY \wedge SZ1.N=SZ2.N}(SZ1 \times SZ2)$

2, **Kérdés:** Kik szeretnek legalább 3 féle gyümölcsöt?
Válasz: $X := \Pi_{SZ1.N}(\sigma_{SZ1.N=SZ2.N \wedge SZ2.N=SZ3.N \wedge SZ1.GY \neq SZ2.GY \wedge SZ2.GY \neq SZ3.GY \wedge SZ1.GY \neq SZ3.GY}(SZ1 \times SZ2 \times SZ3))$

3, **Kérdés:** Kik szeretnek legfeljebb kétféle gyümölcsöt?
Válasz: $\Pi_N(SZ) - X$

4, **Kérdés:** Kik szeretnek pontosan kétféle gyümölcsöt?
Válasz: $(\Pi_N(SZ) - X) \cap Y$

5, **Kérdés:** Kik mit nem szeretnek?
Válasz: $\Pi_N(SZ) \times \Pi_{GY}(SZ) - SZ$

6, **Kérdés:** Kik nem szeretnek minden gyümölcsöt?
Válasz: $Z := \Pi_N(\Pi_N(SZ) \times \Pi_{GY}(SZ) - SZ)$

7, **Kérdés:** Kik azok, akik minden gyümölcsöt szeretnek??
Válasz: $\Pi_N(SZ) - Z$

SQL alapok

R → `select * from R;`
$\Pi_{\mathrm{X}}(R)$ → `select x from R;`
$\sigma_{\mathrm{felt}}(R)$ → `select * from R where felt;`
$\Pi_{\mathrm{X}}(\sigma_{\mathrm{felt}}(R))$ → `select x from R where felt;`
$R \cup S$ → `select * from R union select * from S;`
$R \cap S$ → `select * from R intersect select * from S;`
$R - S$ → `select * from R minus select * from S;`
$R \times S$ → `select * from R,S;`
$R \bowtie S$ → `select * from R natural join S;`
$R \bowtie_{\mathrm{felt}} S$ → `select * from R,S where felt;`

Hajós feladat

1, **Kérdés:** Melyek azok a hajók, amelyeket 1921 előtt avattak fel?
Relációs algebra: $\Pi_{\text{hajónév}}(\sigma_{\text{felavatva}<1921}(\text{hajók})$
SQL: `select hajónév from hajók where felavatva < 1921;`

2, **Kérdés:** Adjuk meg azokat a hajóosztályokat a gyártó országok nevével együtt, amelyeknek az ágyúi legalább 16-os kaliberűek!
Relációs algebra: $\Pi_{\text{osztály, ország}}(\sigma_{\text{kaliber}} > 16(\text{hajóosztályok})$
SQL: `select osztály, ország from hajóosztályok where kaliber >= 16;`

3, **Kérdés:** Adjuk meg az adatbázisban szereplő összes hadihajó nevét! (Ne feledjük, hogy a Hajók relációban nem feltétlenül szerepel az összes hajó!)
Relációs algebra: $\Pi_{\text{hajónév}}(\text{hajók}) \cup \Pi_{\text{hajónév}}(\text{kimenetelek})$
SQL: `select hajónév from hajók union select hajónév from kimenetelek;`

4, **Kérdés:** Adjuk meg a Denmark Strait-csatában elsüllyedt hajók nevét!
Relációs algebra:
$\Pi_{\text{hajónév}}(\sigma_{\text{csatanév}} = \text{'Denmark Strait'} \wedge \text{eredmény} = \text{'elsüllyedt'}(\text{Kimenetelek})$
SQL: `select hajónév from kimenetelek where csatanév = 'Denmark Strait'and eredmény = 'elsüllyedt';`

5, **Kérdés:** Melyek azok az országok, amelyeknek csatahajóik is és cirkálóhajóik is voltak?
Relációs algebra:
$\Pi_{\text{ország}}(\sigma_{\text{típus}} = \text{'bb'}(\text{hajóosztályok})) \cap \Pi_{\text{ország}}(\sigma_{\text{típus}} = \text{'bc'}(\text{hajóosztályok}))$
SQL: `select ország from hajóosztályok where típus = 'bb'intersect select ország from hajóosztályok where típus = 'bc';`

6, **Kérdés:** Melyik hajó melyik országban készült?
Relációs algebra: $\Pi_{\text{hajónév, ország}}(\text{hajók} \bowtie \text{hajóosztályok})$
SQL: `select hajónév, ország from hajók natural join hajóosztályok;`

7, **Kérdés:** Adjuk meg a Guadalcanal csatában részt vett hajók nevét, vízkiszorítását és ágyúinak a számát!
Relációs algebra: $\Pi_{\text{hajónév, vizkiszorítás, ágyúkszáma}}(\sigma_{\text{csatanév}} = \text{'Guadalcanal'}(\text{kimenetelek} \bowtie \text{hajók} \bowtie \text{hajóosztályok}))$
SQL: `select hajónév, vízkiszorítás, ágyúkszáma from kimenetelek natural join hajók natural join hajóosztályok where csatanév = 'Guadalcanal';`

8, **Kérdés:** Soroljuk fel a biztosan 1943 előtt épült hajókat!
Relációs algebra: $\Pi_{\text{hajónév}}(\sigma_{\text{felavatva}} < 1943(\text{hajók})) \cup \Pi_{\text{hajónév}}(\sigma_{\text{dátum}} < \text{'1/1/43'}(\text{Csaták} \bowtie \text{Kimenetelek})$
SQL: `select hajónév from hajók where felavatva < 1943 union select hajónév from csatákak natural join kimenetelek where dátum < '1/1/43';`

4. Gyakorlat

Hajós feladat folytatás

9, **Kérdés:** Melyik csatában volt mindenféle eredmény?
Relációs algebra: $\Pi_{\text{csatanév, eredmény}}(\text{Kimenetelek}) \div \Pi_{\text{eredmény}}(\text{Kimenetelek})$
Osztás nincs SQL-ben, ezért át kell alakítani:
Másik megoldás relációs algebrában:
$\Pi_{\text{csatanév}}(K1) - \Pi_{\text{csatanév}}(\Pi_{\text{csatanév}}(K1) \times K2 - K1)$
SQL:

10, **Kérdés:** Melyik években avattak legalább 3 hajót?
Relációs algebra:
$\Pi_{\text{h1.felavatva}}$
$(\sigma_{h1.felavatva=h2.felavatva \wedge h2.felavatva=h3.felavatva \wedge h1.hajk \neq h2.hajk \wedge h1.hajk \neq h3.hajk \wedge h2.hajk \neq h3.hajk}$
$(h1 \times h2 \times h3)$
SQL:
```
select distinct h1.felavatva from hajók h1, hajók h2, hajók h3 where
h1.felavatva = h2.felavatva and h2.felavatva = h3.felavatva and
h1.hajónév != h2.hajónév and h1.hajónév != h3.hajónév and
h2.hajónév != h3.hajónév;
```

11, **Kérdés:** Az 1921-es washingtoni egyezmény betiltotta a 35 000 tonnánál súlyosabb hajókat. Adjuk meg azokat a hajókat, amelyek megszegték az egyezményt!
Relációs algebra: $\Pi_{\text{hajónév}}(\sigma_{\text{felavatva} > 1921 \wedge \text{vízkiszorítás} > 35000}(\text{hajók} \bowtie \text{hajóosztályok}))$
SQL:
```
select hajónév from hajók natural join hajóosztályok where felavatva > 1921 and
 vízkiszorítás > 35000;
```

12, **Kérdés:** Adjuk meg azokat a hajókat, amelyek "újjáéledtek", azaz egyszer már megsérültek egy csatában, de egy későbbi csatában újra harcoltak!
Relációs algebra:
SQL:

13, **Kérdés:** Adjuk meg azokat az osztályokat, amelyekbe csak egy hajó tartozik!
Relációs algebra:
SQL:

14, **Kérdés:** Évenkénti bontásban hány hajót avattak?
Relációs algebra:
SQL:

15, **Kérdés:** Mely hajóosztályból mikor avatták az utolsó hajót?
Relációs algebra:
SQL:

SQL

Dual-tábla

A dual táblát használva tudunk olyan lekérdezéseket végezni, amelyekhez nem szükséges tábla. Példa:

```
select 'hello' from dual;  --Attribútum név 'HELLO' lesz.

select 1+1 from dual;  -- A mező neve 1+1 lesz.

select 1+1 eredmény from dual; -- a mező neve 'EREDMÉNY' lesz.

select 1+1 select from dual;  -- HIBA! A fordító azt hiszi, hogy a select
   új parancs.

select 1+1 "select" from dual; -- A mező neve select lesz.

select 1+1 "SELECT" from dual;  -- A mező neve SELECT lesz.

select 'hello' köszönés, 1+1 összeg, 2*3 szorzat from dual; -- 3 oszlopos
   táblázat.
```

Distinct

A relációs algebrával ellentétben az SQL eredménye multihalmaz. A duplikátumokat a **`distinct`** segítségével szűrhetjük ki.

```
select gy from sz;  -- A tábla összes sorának gyümölcse felsorolása kerül.
select distinct gy from sz; -- Minden gyümölcs pontosan egyszer szerepel.
```

Rownum

Néha szükség lehet egy lekérdezés sorainak explicit megszámozására. Ezt a `rownum` pszeudomező segítségével tehetjük meg.

```
select rownum, Hajónév from Hajók;  -- ROWNUM nevű attribútumot ad hozzá,
   amely a  sorok sorszámait tartalmazza.
select rownum "sorszám", Hajónév from Hajók;  -- A ROWNUM attríbútumot á
   tnevezi sorszám-ra.
```

Összefűzés

Két attribútumot és mezőit össze lehet fűzni. Ehhez a `||` műveletet használhatjuk.

```
select hajónév||hajónév from hajók; -- Összefűzi a két attribútumot.
select hajónév||hajónév egyesített from hajók; --Összefűzi a két attribú
   tumot, és a nevét megváltoztatja EGYESÍTETT-re.
```

Műveletek, in, not

A legkérdezésekben használhatóak a megszokott műveletek ($==, != , <, \leq, >, \geq$), továbbá rendezett n-esek esetén használható az `in`, illetve ennek tagadása a `not in` kulcsszó. A `not` kulcsszó bármely logikai kifejezés tagadására használható. Példák:

```
select 'igaz' eredmény from dual where 2>1; -- Igazat ír ki.
select 'igaz' eredmény from dual where 2>3; -- Semmit.
select 'igaz' eredmény from dual where not 2>3; --Igazat ír ki.
select 'igaz' eredmény from dual where 1 in (1,2,3);
select 'igaz' eredmény from Hajók where 'Kongo' in Hajónév;  -- Kiírja,
   hogy igaz, ha Kongo benne van a hajónév oszlopban.

select 'igaz' eredmény from dual where 'Füles' in ('Malacka', 'Füles');
select 'igaz' eredmény from dual where (1,2) in ((1,2), (3,4), (5,6));
```

Like

Szöveg típusú mezők esetén a `like` kulcsszóval mintaillesztés szerűen szűrhetünk. a `'_'` karakter pontosan egy, a `'%'` bármennyi karakterre illeszkedik. Példák:

```
select * from Hajók where Hajónév like 'M%';  -- M betűvel kezdődő hajó
   nevek kiírása.
select * from Hajók where Hajónév like '%i%';  -- Minden hajűt írjunk ki,
   amiben van i betű.
select * from Hajók where Hajónév like '_%';  -- Írjuk ki azokat a hajókat
   , aminek a 2. betűje i.
select * from Hajók where Hajónév like '%M%' or Hajónév like '%m%';  -- M
   vagy m betűt tartalmazó hajók kiírása
```

Between

Ha azt szeretnénk megtudni, hogy egy mező értéke benne van-e egy adott intervallumban, akkor a `between` kulcsszót használhatjuk. Példák:

```
select * from emp where sal between 1000 and 2000;  -- Írjuk ki azokat az
   employeekat akinek a fizetése 1000 és 2000 között van.
select * from emp where ename between 'J' and 'S';  -- J és S betű közötti
    emberek.
```

Exists

Annak eldöntésére, hogy egy allekérdezés adott-e vissza legalább egy sort az `exists` kulcsszót használjuk, amely hamis értéket ad vissza, ha nem tért vissza sorral az allekérdezés, igazat különben. Példák:

```
select 'igaz' eredmény from dual where exists (select * from sz where gy =
   'körte'); --igazat ír ki, ha létezik sz-ben körte nevű gyümölcs
select 'igaz' eredmény from dual where not exists (select * from sz where
   gy ='dinnye'); -- igazat ír ki, ha nem létezik az sz-ben dinnye nevű gy
   ümölcs
```

*-hoz adás

Alapvetően a `select *`-hoz nem lehet plusz attribútumokat adni, de lehet olyan helyzet, amikor nekünk a teljes táblához kell egy sorszámozást kapcsolnunk. A következőképp oldhatjuk meg:

```
select rownum sorszám, * from Hajók;  --HIBA, * mellett nem lehet semmi

select rownum sorszám, Hajók.* from Hajók;  -- Így már hozzáadja a
táblázathoz
```

5. Gyakorlat

All, any

Az `all`, `any` kulcsszavakkal összehasonlíthatunk egy mezőt egy allekérdezést vagy egy rendezett n-es minden elemével. Az `any` akkor ad igazat, ha legalább egy elemre igaz az összehasonlítás, mig a `all` pedig, ha az összes elemre. Példák:

```
select * from emp
  where sal > all (select sal from emp where job = 'SALESMAN');
select * from emp
  where sal > any (select sal from emp where ename='FORD');
```

NULL

Az SQL-ben egy mező felvehet `NULL` értéket. Fontos megjegyezni, hogy a `NULL` $\neq$ az üeres szöveggel vagy a 0-val, és a `NULL = NULL` és `NULL != NULL` is hamisat ad vissza. Az `is NULL` kulcsszó segítségével dönthetjük el, hogy egy mező értéke `NULL`-e. Példák:

```
select 'igaz' from dual where null = null; --Nem tudja eldönteni a gép a v
   álaszt.

select 'igaz' from dual where null != null; -- Itt sem.

select * from emp where comm = null; -- Nem tudja megjeleníteni.

select * from emp where comm != null; --Ezt sem.

select *from emp where comm is null; -- Ezért kell az is null. Megjeleníti
    azokat, akiknek nincs jutalékuk.

select *from emp where comm is not null; --Így azokat, akiknek van jutalé
   ka.
```

Coalesce

Mivel bármely `NULL`-ra végzett művelet `NULL`-t eredményez, ezért szükség lehet a `NULL` érték helyére egy létező értéket írni. Erre szolgál a `coalesce` függvény, amely visszaadja az első nem `NULL` értéket a paraméterei közül.
A `coalesce` ANSCII szabványban létező függvény. Oracle SQL-ben hivatkozhatunk erre `nvl` függvény névvel is.
Több mezőt is fel lehet sorolni, amit megvizsgál a gép. Pl.: vezetéknév, keresztnév, becenév. Amíg nem találunk valamit, megy tovább, majd ha találtunk, akkor azt fogja használni.

```
select ename név, sal fizetes, comm jutalék, sal+comm "össz fiz" from emp;
    --Mivel az egyik érték NULL, az összeadás eredménye is NULL lesz.
select ename név, sal fizetes, comm jutalék, sal+ coalesce (comm, 0) "össz
    fiz" from emp; -- coalesce segítségével 0-ra állítjuk.
```

Csoportfüggvények

Az SQL tartalmaz olyan csoportfüggvényeket, amelyek a lekérdezés minden során végrehajtódnak, majd egy értékkel térnek vissza.
5 alap csoportosítófüggvény: `sum`, `count`, `avg`, `max`, `min`
Ezek a függvények a null-t nem veszik figyelembe, kivéve a `count`.
Példák:

```
select sum(sal), count(sal), sum(sal)/count(sal), avg(comm) from emp;
```

Group by

Adatok csoportosítására a **group by** kulcsszót használjuk. A **where** kulcsszóval a csoportosítás elött szűrhetünk, míg a **having** kulcsszóval a létrejött csoportokat szűrhetjük. **Group by**-nál a **NULL** értéke egy.

```
select job from emp group by job; -- összeszedi milyen job-ok vannak
select job, sal from emp group by job; -- ez hiba, mivel egy job-hoz több
   sal társul
select job, sum(sal) from emp group by job; -- így már jó, összegzi a sal-
   t job

select job, sum(sal), count(sal), count(*) from emp
where sal > 1000
group by job;
```

Order by

Az adatok rendezéséhez **order by** kulcsszót használjuk. Rendezhetünk csökkenő sorrendben(**desc**), és növekvő sorrendben(**asc**) is. Az alapértelmezett a növekvő sorrend.
Oszlopszámot is megadhatunk. Pl.: **order by 1 asc** → Az első oszlop szerint fog rendezni.
Ha szükségünk van arra, hogy tobb oszlop szerint rendezzen, megadhatjuk.Pl: **order by 1,3 asc** → Elsődlegesen 1. oszlop szerint, és ha azon belül vannak egyezések akkor a 3. oszlop szerint rendez.

```
select job, deptno from emp
  group by job, deptno
  order by job, deptno desc;
```

subsubsection*Théta-összekapcsolás

Théta-összekapcsolás SQL-ben a következőképp írható le: **<tábla 1> join <tábla 2> on <feltétel>**. Példa:

```
select * from ivók join látogat on név = ivó;
```

Megjegyzések

A **select-from-where** állítások multihalmaz szemantikát használnak, de a halmazműveletek **union**, **intersect**, **minus** eredményei halmazok.

Forrás

- ELTE IK programtervezői informatikus szak 2013 tavaszi féléves Adatbázisok I. előadás és gyakorlat alapján írt órai jegyzetem.
- Bókay Csongor: Adatbázisok I., ELTE IK, 2013