

**KENDE - KOTSIS - NAGY**

**Adatbázis-kezelés  
az ORACLE-rendszerben**

**ELMÉLET**

**PANEM**

E felsőoktatási tankönyv a Művelődési és Köznevelési Minisztérium támogatásával, a Felsőoktatási Pályázatok irodája által lebonyolított felsőoktatási tankönyvtámogatási program keretében jelent meg.

Copyright © Kende Mária (III.Rész, IV.Rész)

Copyright © Kotsis Domokos (I.Rész)

Copyright © Nagy István (II.Rész, III.Rész)

A kiadásért felel a Panem Könyvkiadó Kft. ügyvezetője, 2002.

Panem Kft.  
1385 Budapest, Pf 809  
Hungary

ISBN 963 545 xxx x

Lektor: Juhász István  
Műszaki szerkesztő: Beck Zsuzsa

A Panem könyvek megrendelhetők az (1) 340-1515 hívószámú telefonon, illetve a 1385 Budapest, Pf. 809 levélcímen.

[panem@panem.hu](mailto:panem@panem.hu)

[www.panem.hu](http://www.panem.hu)

A könyv a Kiadó és a Szerzők gondos munkájának eredménye. Ennek ellenére előfordulhatnak benne hibák, melyeknek következményeiért sem a Szerzők, sem a Kiadó nem vállalnak felelősséget.

Minden jog fenntartva. Jelen könyvet, illetve annak részeit tilos reprodukálni, adatrögzítő rendszerben tárolni, bármilyen formában vagy eszközzel – elektronikus úton vagy más módon – közölni a kiadók engedélye nélkül.

# Tartalom vázlat

## I. Az információ és feldolgozása (Kotsis Domokos)

1. Az információ-feldolgozás kezdetei
2. A folyamat-szemléletű információ-feldolgozás
3. Az adatbázis-szemléletű információ-feldolgozás
4. Az információ-feldolgozás jelene és jövője

## II. A relációs adatmodell (Nagy István)

5. Relációalgebra
6. A relációs adatmodell szerkezete
7. A relációs adatbázis tervezése

## III. Adatbázis-kezelés az ORACLE környezetben (Kende Mária, Nagy István)

8. Az SQL\*Plus környezet és az SQL nyelv alapjai
9. Adatbázis-kezelés SQL nyelven
10. A PL/SQL nyelv

## IV. Adatbázis-kezelés Delphiben (Kende Mária)

11. A Delphi, mint fejlesztési és beágyazó környezet
12. Lokális adatbázisok használata
13. Oracle adatbázisok használata

Függelék, Tárgymutató, CD melléklet

# Tartalomjegyzék

Előszó .....	12
AZ INFORMÁCIÓ ÉS FELDOLGOZÁSA .....	13
Az információ-feldolgozás kezdetei .....	14
Az első lépések .....	14
Az információk összegyűjtése, válogatása .....	14
Kódolás, rögzítés .....	14
Felhasználás .....	15
Járulékos problémák .....	15
Az adatok helyessége .....	15
Titkosság .....	15
Adatátvitel .....	16
Információ-feldolgozás számítógéppel .....	16
A folyamat-szemléletű információ-feldolgozás .....	17
Optimális szerkezetű állományok .....	17
Legfontosabb állomány struktúrák .....	19
Szekvenciális állomány struktúrák .....	19
Fizikai szekvenciális állomány struktúrák .....	19
Bináris, logaritmikus keresés .....	19
Peterson-féle keresés .....	20
Logikai szekvenciális állomány struktúrák .....	20
Hierarchikus állomány struktúrák .....	23
A hierarchikus állományokat kezelő eljárások .....	23
Belső mutatós eljárások .....	23
Külső mutatós eljárások .....	24
Kiemelten fontos hierarchikus struktúrák .....	24
Bináris fák .....	24
B-fák .....	25
Hálós állomány struktúrák .....	26
A hálós állományokat kezelő eljárások .....	26
Belső mutatós eljárások .....	26
Külső mutatós eljárások .....	26
Nem konzekutív állomány struktúrák .....	26
Indexelt állomány struktúrák .....	26
A sűrű indexelés .....	26
A ritka indexelés .....	27
Az index-szekvenciális szervezés .....	27
A direkt állományszervezés .....	29
A leképezések .....	29
A szinonimok kezelése .....	30
A szinonimkezelő módszerek összehasonlítása .....	30
Az adatbázis szemléletű információ-feldolgozás .....	34
Adatbázisok tervezése .....	34

A modellek és kapcsolataik .....	34
Az adatbázis felügyelő és alapvető feladatai .....	35
Konkurrens folyamatok .....	35
A felhasználó eszközei .....	36
Az operációs rendszer által nyújtott támogatás .....	37
Az alkalmazott rendszerek sajátosságai .....	38
A hierarchikus modell .....	38
A hálós modell .....	39
Az információ-feldolgozás jelene és jövője .....	41
Relációs adatmodell és használata .....	41
A relációs algebra .....	42
A relációs algebra alapl műveletei .....	42
A következmény műveletek .....	42
A relációs kalkulus .....	44
Lekérdezés relációs rendszerekben .....	45
QBE .....	45
SQL .....	45
A negyedik generációs (4GL) nyelvek .....	46
Kliens-szerver architektúrák .....	46
Osztott adatbázisok kezelése .....	47
Központosított protokollok .....	48
A központi zárellenőrzés .....	48
A token módszer .....	48
Az elsődleges példány módszer .....	48
Osztott protokollok .....	49
Az időbélyeg módszer .....	49
Az adatok bizalmas kezelése .....	49
Rejtjelezés .....	49
konvencionális kódolás .....	50
nyilvános (rejtő) kulcsú kódolás .....	51
Kulcs gondozás .....	51
Kulcsgenerálás .....	51
Kulcskiosztás .....	51
Kulcs tárolás .....	52
Felhasználó- és partner-azonosítás, hozzáférés-védelem .....	52
A felhasználó-azonosítás .....	52
A partner-azonosítás .....	53
Üzenethitelesítés, digitális kézjegy .....	53
A hozzáférés-védelem .....	53
Nested Relational Model .....	54
Objektum-orientált adatbázis-kezelés .....	55
Zártság .....	55
Öröklődés és többértésűség .....	55
Deduktív adatbázisok .....	56
Szakértői rendszerek .....	56
Adatbányászat .....	58

Az információk integrálása.....	58
Adatbázisok összekapcsolása .....	59
Adattárház .....	59
Közvetítés .....	60
A RELÁCIÓS ADATMODELL.....	61
Relációalgebra .....	62
Algebrai alapfogalmak .....	62
Jelölések .....	62
A halmazalgebra alapfogalmai .....	63
Részhalmaz, halmaz bővítése, halmazegyenlőség, hatványhalmaz.....	63
Halmazok metszete.....	64
Halmazvetítés (relatív részhalmaz).....	64
Halmazok különbsége .....	64
Halmazok egyesítése, uniója .....	64
Halmazok pontozott egyesítése, minősített név.....	65
Halmazok szimmetrikus differenciája .....	65
Multihalmaz-műveletek.....	65
Multihalmaz, monohalmaz, redukálás .....	65
Halmaz eleme .....	66
Multiplikátor, multiplicitás .....	66
Valódi multihalmaz .....	66
Részhalmaz, halmazegyenlőség .....	67
Halmazok metszete.....	67
Halmazok különbsége .....	67
Halmazok multiuniója .....	67
Halmazalgebrai azonosságok .....	68
Multiplikált halmaz .....	68
Redukáló műveletek .....	68
Redukált részhalmaz, redukált halmazegyenlőség .....	69
Halmazok redukált metszete.....	69
Halmazok redukált különbsége .....	69
Halmazok redukált egyesítése (uniója).....	70
Vektoralgebrai alapfogalmak .....	70
Vektor, komponens, index, monovektor, multivektor .....	70
Komponenshalmaz-függvény.....	71
A vektor fogalmának értelmezése .....	72
Normál vektor, vektor normál alakja.....	72
Ekvivalens vektorok .....	73
Alvektor.....	73
Közös gyök.....	74
Hasonló vektorok .....	75
Vektorok hasonlósági függvénye .....	75
Vektorvetítés (halmaz-relatív és vektor-relatív alvektor) .....	76
Monovektor-vetítés.....	77
Multivektor-vetítés .....	78
Vektorláncolás.....	80

Vektorok pontozott láncolása .....	81
Vektorkivonás .....	81
Egyesítő láncolás .....	82
Halmazrendszerek .....	83
Egyszerű és összetett elemek .....	83
Halmazrendszerek típusai .....	83
Hiperhalmaz .....	84
Hiperhalmaz metszete és egyesítése .....	84
Halmazvektor .....	84
Halmazvektor egyszerű vektorvetítése, alvektora .....	85
Halmazvektor-vetítés .....	85
Halmazvektor résztartománya .....	86
Vektorhalmaz .....	86
Komponenshalmaz-függvény kiterjesztése vektorhalmazra .....	87
Vektorhalmaz egyszerű halmazvetítése, részhalmaz .....	87
Vektorhalmaz-vetítés (bevezető gondolatok) .....	87
Halmazok Descartes-szorzata, rekord .....	88
Az egytényezős Descartes-szorzat .....	89
Néhány megjegyzés a Descartes-szorzás elvégzéséről .....	89
Halmazok hatványozása .....	91
Rekordvetítés (halmaz-relatív és vektor-relatív alrekord) .....	91
Hasonló rekordok (relatív azonos rekordok) .....	92
Vektorhalmaz (a Descartes-szorzathoz származtatva) .....	93
Vektorhalmaz-vetítés (halmaz-relatív és vektor-relatív vetítés) .....	94
Alvektorhalmaz .....	95
Hasonlósági vetítés .....	95
Hasonló vektorhalmazok .....	96
Transzvertált vetítés .....	97
Transzvertálás .....	97
Megjegyzések a transzvertáláshoz és a transzvertált vetítéshez .....	98
Vektorhalmaz parciális és teljes keretei .....	98
Vetítések és tulajdonságaik (összefoglaló) .....	99
Halmazvetítés .....	99
Vektorvetítés .....	100
Halmazvektor-vetítés .....	100
Rekordvetítés .....	100
Vektorhalmaz-vetítés .....	101
Hasonlósági vetítés .....	101
Transzvertált vetítés, transzvertálás .....	101
Hasonlóságok és tulajdonságaik (összefoglaló) .....	102
Vektor-ekvivalencia .....	102
Vektorhasonlóság .....	102
Rekordhasonlóság .....	103
Vektorhalmaz-hasonlóság .....	103
A relációalgebra alapfogalmai .....	104
Relációséma, attribútum .....	104

Relációséma keretei (értéktartományai) .....	104
A relációséma értelmezése .....	105
Reláció .....	106
Reláció keretei .....	106
A reláció értelmezése .....	107
További származtatott fogalmak .....	108
Rekord, mint a reláció eleme .....	108
Relációban álló attribútumértékek .....	108
Alséma, alreláció (attribútumhalmazra és attribútumvektorra relatív) .....	108
Reláció, mint tábla .....	109
Altábla .....	110
Részséma, részreláció, résztábla .....	110
Relációs adatbázismodell (adatbázisséma), adatbázis .....	110
Leképezések és függvények .....	112
Egyszerű leképezés .....	112
Egyszerű leképezés vetületei .....	113
Elem képhalmaza (elem-leképezés) .....	113
Egyszerű leképezés inverze .....	114
A leképezés és a leképezés-vetületek értelmezése .....	114
Egyszerű függvény .....	114
Bijektív függvény .....	115
Összetett leképezések származtatása az egyszerű leképezésekből .....	116
Vektorleképezés .....	116
A vektorleképezés vetületei .....	117
Az unáris, a bináris, az $n$ -változós és az egyrétegű vektorleképezés .....	118
Reláció leképezése .....	119
Elem képhalmaza (elem vektorleképezése) .....	120
Leképezés-vetületek, mint speciális részhalmazok .....	122
Vektorfüggvény .....	122
Hipervektor-leképezés .....	123
A hipervektor-leképezés vetületei .....	123
Elem képhalmaza (elem hipervektor-leképezése) .....	124
Hipervektorfüggvény .....	125
Halmazleképezés .....	125
A halmazleképezés vetületei .....	126
Egyszerű halmazleképezés .....	126
Elem képhalmaza (elem halmazleképezése) .....	126
Halmazfüggvény .....	126
Vektorhalmaz-leképezés .....	127
A vektorhalmaz-leképezés vetületei .....	127
Elem képhalmaza (elem vektorhalmaz-leképezése) .....	128
Vektorhalmazfüggvény .....	128
Műveletek és struktúrák .....	129
Művelet .....	129
Néhány megjegyzés műveletek definíciójáról .....	129
Egyszerű művelet (unáris és bináris műveletek), infix jelölés .....	130

Összetett műveletek .....	131
Struktúra .....	131
Egyszerű struktúra .....	131
Struktúrák tulajdonságai .....	132
Asszociativitás .....	132
Kommutativitás .....	132
Invertálhatóság .....	132
Egységelem .....	133
Összetett struktúrák .....	134
Osztályozás .....	135
Alapfogalmak .....	135
Ekvivalencia-reláció .....	135
Halmazfelosztás, particionálás .....	136
Osztálykritérium .....	136
Néhány megjegyzés a halmazfelosztás és az ekvivalencia-reláció kapcsolatáról .....	137
Gyakorlati osztályozások .....	138
Halmazfelosztás osztálykritérium szerint .....	138
Sémafelosztás és reláció-felosztás osztálykritérium szerint .....	139
Sémafelosztás és reláció-felosztás attribútumhalmaz szerint .....	140
Elemi-sémafelosztás, egységfelosztás .....	141
Relációalgebrai műveletek .....	142
Szerkezzettartó műveletek .....	143
Reláció átnevezése, alias név, másodlagos attribútumnév .....	143
Reláció redukálása .....	144
Relációk metszete, különbsége és egyesítése .....	144
Szelekció (kiválasztás, szűrés) .....	145
Szerkezzetmódosító műveletek .....	145
Projekció (vetítés) .....	145
Dekompozíció .....	146
Descartes-szorzás (szorzás) .....	146
Relációk osztása (osztás) .....	147
Természetes összekapcsolás (Natural Join) .....	148
Általános összekapcsolás (Théta összekapcsolás, Join) .....	148
Relációműveletek kapcsolata .....	149
Osztályozó műveletek .....	150
Bevezető példa .....	151
Az osztályozó műveletek általános alakja .....	155
A bevezető példa folytatása .....	156
Osztályozó műveletek kompozíciója .....	157
Egyszerű lekérdezések .....	157
Egy összetett mintapélda .....	161
Feladatok .....	164
A relációs adatmodell szerkezete .....	166
A funkcionális-függőség .....	166
Alapfogalmak .....	167
Hatványfüggőség, függőség, kompatibilitás .....	167

Függőség magja, képe, egyszerű és összetett függőség.....	168
Zérusfüggőség, egységfüggőség, triviális függőség .....	168
Funkcionális-függőség és relációsémája .....	168
Funkcionális-függőség és relációséma kompatibilitása.....	169
Irreducibilis relációséma .....	169
Funkcionális-függőség vektorfüggvénye, „függvényjellege” .....	169
Funkcionális-függőség és vektorfüggvényének szemléltetése .....	170
A funkcionális függőség és a sémafelosztás kapcsolata .....	171
Függőségi-felosztás (belső-felosztás).....	171
A függőségi-felosztás szemléltetése .....	171
A függőségi-felosztás tulajdonságai .....	172
A funkcionális-függőség gyakorlati meghatározása.....	175
A logikai-struktúra .....	177
Alapfogalmak .....	177
Logikai-struktúra, kompatibilitás, részstruktúra, triviális struktúra.....	177
A logikai-struktúra és a relációséma kapcsolata.....	178
Logikai-struktúra relációsémája .....	178
Relációséma struktúrája, kompatibilitás .....	178
Struktúra-kompatibilitás, séma-kompatibilitás, ekvivalencia .....	179
Kompatibilitások és halmazműveletek kapcsolata .....	180
Logikai-struktúra hatása relációsémára .....	181
A logikai-struktúrák hatásának vizsgálata, ekvivalencia, ortogonalitás .....	181
Néhány szó a logikai-struktúrákról.....	182
Néhány algebrai megjegyzés, ekvivalencia .....	183
A logikai-struktúra és a sémafelosztás kapcsolata.....	184
Ortogonalis relációsémák .....	184
Ortogonalis-felosztás (külső-felosztás) .....	184
Az ortogonalis-felosztás tulajdonságai .....	185
Függőség-algebra.....	188
Alapfogalmak .....	188
Kulcs (kulcshalmaz, elsődleges kulcs, idegen kulcs) .....	188
Szuperkulcs .....	189
A logikai-struktúrák bővítése .....	189
Relációséma burokstruktúrája .....	189
Logikai-struktúra buroksémája, burokséma-halmaza .....	190
A burokséma és az ortogonalis felosztás kapcsolata .....	191
Logikai-struktúra burokstruktúrája, ekvivalens bővítése.....	191
Az Armstrong-szabályok.....	192
Az Armstrong-bővítés .....	194
Az Armstrong-szabályok következményei .....	194
Kulcs meghatározása az Armstrong-szabályok segítségével.....	194
Függőségcsalád .....	195
Függőség-generátor .....	197
Logikai-struktúrák ekvivalenciája .....	197
Irredundáns függőség-generátor .....	198
Minimális függőség-generátor.....	198

A függőség-generátorok néhány tulajdonsága.....	198
A relációalgebra és a függőségalgebra kapcsolata .....	199
A relációalgebra fogalomhierarchiája: .....	199
A függőségalgebra fogalomhierarchiája:.....	200
Függőségalgebrai műveletek .....	200
Függőségek egyesítése .....	201
Függőségek szorzata.....	201
Függőség felosztása.....	201
A relációs adatbázis tervezése .....	203
Alapfogalmak .....	203
Elsődleges és másodlagos attribútumok .....	203
Speciális függőségek .....	203
Teljes és tranzitív függőség .....	203
Belső-függőség.....	204
Relációséma dekompozíciója attribútumvektor-halmaz szerint .....	204
Relációséma dekompozíciója funkcionális függőség szerint.....	205
Relációséma veszteségmentes dekompozíciója.....	205
Relációséma függőségőrző dekompozíciója.....	207
Normalizálás.....	208
Normálformák .....	210
Az ősmódel (alapmódel) .....	210
Nulladik normálforma, 0NF .....	212
Első normálforma, 1NF .....	212
Második normálforma, 2NF .....	214
Harmadik normálforma, 3NF .....	215
Boyce-Codd normálforma, BCNF.....	216
Adatbázis-anomáliák .....	218
Módosítási anomália.....	219
Törlési anomália .....	219
Bővítési anomália .....	220

# Előszó

Először jelenik meg magyar nyelven olyan felsőoktatási adatbázis-tankönyv, mely a bemutatott példák, és a tárgyalás tematikájában is ahhoz a korszerű és világszerte elterjedt adatbázis-kezelő rendszerhez kötődik, melyet az ORACLE cég neve fémjelez.

A könyv célja, hogy az adatbázis-kezelés elméletének bemutatásán túl a gyakorlatban is jól hasznosítható ismereteket adjon az Oracle adatbázis-kezelő rendszer használatához, valamint e rendszer Delphi fejlesztő környezetből való eléréséhez.

A könyv négy fő részből, és a nagyszámú mintapéldát tartalmazó CD mellékletből áll.

Az első rész általános elméleti bevezetést tartalmaz, mely kitér a folyamatszemplétű, valamint az adatbázis-szemléletű információfeldolgozás alapvető elveire, módszereire, továbbá a modern információfeldolgozás olyan kurrens területeire, mint az osztott adatbázisok kezelése, a rejtjelezés, az objektum-orientált adatbázis-kezelés, valamint az adatbázis-kezelés társterületeire; a szakértői rendszerekre, az adattárházakra, és az adatbányászatra.

A második rész alapos algebrai tárgyalás keretében a relációs adatmodell logikai szerkezetével foglalkozik. Fő területei a relációalgebra, a függőségek tulajdonságai, és bevezető jelleggel a relációs adatmodell tervezése. E rész célja az adatbázis-kezelés gyakorlatában alkalmazott módszerek elméleti indoklása és megalapozása.

A harmadik rész gyakorlati példákon keresztül az Oracle rendszerben való lekérdezéseket, programozást és az alapvető rendszer-adminisztrátori teendőket mutatja be. Részletesen ismerteti az SQL, az SQL\*Plus és a PL/SQL nyelveket. Ennek során bemutatja a DDL, a DML és a DCL eszközöket; az adatbázis létrehozását, módosítását, lekérdezését, a tranzakciók kezelését, a jogosultságokat és a kivételkezelést. A PL/SQL bemutatása során eljut az összetett alprogramok, a paraméteres kurzorok tárgyalásáig is.

A negyedik rész alapvető célja, hogy bemutassa beágyazó környezetként a Delphi fejlesztő rendszert, valamint az ebben történő adatbázis-kezelést. Ennek során kitér a Delphi rendszerből való, különböző típusú adatbázisokkal való kapcsolattartás módjaira, az SQL utasítások és PL/SQL alprogramok Delphi alkalmazásban való futtatásának módjára.

A könyv elkészítésében a szerzők elsősorban azoknak mondanak köszönetet, akik javasla-  
taikkal, bíráló megjegyzéseikkel segítették, hogy igényes munka születhessen. Köszönet  
illeti *Békéssy Andrást, Boschán Évát, Szelezsán Jánost és Vajda Istvánt*, akik észrevételeik-  
kel, kritikáikkal serkentették az alkotó munkát, és mindenek előtt a könyv lektorát, *Juhász  
Istvánt*, aki a könyv elkészítésének egész folyamatában biztos támogatást jelentett. Szeret-  
nénk megköszönni a Panem kiadónak és különösen *Sághi Mártának* a türelmes és segítő  
noszogatását. Köszönet *Czinkóczi Lászlónak* és *Nagy Kálmánnak*, akik az Oracle rendszer-  
rel való ismerkedésben és a napi problémák megoldásában egyaránt baráti segítséget jelen-  
tettek. Hasonlóképpen köszönjük *Nyéki József* úrnak a Delphi 6 Enterprise használatával  
kapcsolatos szakmai tanácsait, és a IV. rész bírálatával nyújtott támogatását.

Budapest, 2002. január 9.

A szerzők

# I. RÉSZ

## AZ INFORMÁCIÓ ÉS FEL- DOLGOZÁSA

E rész az információ-feldolgozás célját, eszközeit tekinti át. Mivel az információ gyűjtése és feldolgozása azóta folyik, amióta ember él a földön, így a tárgyalás sorrendjében a történetiséget tartottuk szem előtt. A folyamat-szemléletű információ-feldolgozás például a legrégebbi módszerek egyike – és bár jelentősége csökkent az idők folyamán – ma is élő, használt módszer. A jóval későbbi, s egy időben lényegében egyeduralkodó hierarchikus adatbázisok gyakorlatilag eltűntek, s átadták helyüket azoknak a relációs adatbázisoknak, melyek alapja egy, a hierarchikus adatbázisokkal egyidős elmélet, s melyek gyakorlati térhódítása az elmélet ismertté válása után húsz évvel következett be. Bemutatásunk utolsó részében a szakértői rendszerekkel, mint a jövő lehetséges rendszereivel is foglalkozunk annak ellenére, hogy az első ilyen rendszerek egy emberöltővel ezelőtt készültek, s hogy ma a gyakorlatban ilyen rendszereket alig készítenek.

## 1.FEJEZET

# Az információ-feldolgozás kezdetei

A bevezetőben említettük, hogy az információ feldolgozása egykorú az emberiséggel. Ez az állítás meglepőnek tűnhet mindenkinek, aki eddig információ-feldolgozáson csak annak elektronikus eszközökkel végzett formáját értette. Ha azonban alaposabban megvizsgáljuk a tevékenységet, kiderül, hogy egy ókori írnok, amikor agyag- vagy viasztáblája, esetleg papi-rusztekerese fölé hajolt, hasonló cél érdekében munkálkodva lényegében ugyanolyan lépéseket hajtott végre, amilyeneket egy modern informatikus. Kövessük ezeket a lépéseket.

## Az első lépések

### Az információk összegyűjtése, válogatása

Az információ-feldolgozás első lépése mindig a tények gyűjtése. Természetesen valamennyi általunk ismert – vagy ismerni vélt – tényt nem vagyunk képesek figyelembe venni. Minden információs rendszer készítője tehát a tényeknek csak egy részhalmazával foglalkozhat. Így ki kell választania a világnak egy olyan – jelentősen leegyszerűsített – modelljét, mely kezelhető számú ténnyel leírható, ugyanakkor jó közelítést ad az általa vizsgált szempontból.

Egy egyiptomi írnok számára esetleg a Nílus áradásának és a gabonatermésnek adatai jelenthették egy konkrét esetben a rögzítendő világot, egy középkori sapkakészítőnek a világ az emberek fejméretéből és a divatos sapkákhöz szükséges anyagmennyiségből állhatott.

Az újabb kor technikája több adat kezelését tette lehetővé, de a cél és az alapelvek alig változtak. Az 1890-es USA-beli népszámlálás erre jó példa. Ez a – Hollerith által vezetett – munka sok újdonságot hozott az adatrögzítés és tárolás terén, ám a lakosság adatainak itt is csak egy részhalmazát lehetett feldolgozni. Ez akkor is igaz, ha a technikai újdonságoknak köszönhetően ez a részhalmaz óriásinak tűnt minden korábbi feldolgozáshoz képest.

### Kódolás, rögzítés

A begyűjtött információt a későbbi felhasználás érdekében rögzíteni kell. A rögzítéshez valamilyen kódolás szükséges. Ezekután a rögzített anyag csak a kódot ismerők számára jelent információt, a kívülállók semmit nem értenek belőle.

Ha egy egyiptomi írnok rögzíteni akarta a Nílus áradásával kapcsolatos adatokat, azt a saját nyelvén megfogalmazva, az általa ismert írásjelekkel rögzítette papiruszra. Ezek az adatok sumér kollégája számára (hacsak az illető nem ismerte az egyiptomi nyelvet és írást) ugyanúgy nem jelentettek semmit, mint azoknak az európai tudósoknak, akik Champollion előtt kísérleteztek annak értelmezésével.

Logikáját illetően nem különbözik alapvetően ettől az a feldolgozás, melyet Hollerith csapata végzett, mindössze a papiruszra írás helyett egy másik adathordozón: a lyukkártyán (ez egy kemény papírlap, melynek mérete megegyezik az 1890-ben használt egy dolláros bankjegy méretével), egy másik kódolási módszert: a számokat és betűket lyukkombinációkkal ábrázoló lyukasztást választották (a Hollerith-féle lyukkártyán 80 oszlopban egy-egy karakternyi adat ábrázolható lyukkombinációkkal). Azt pedig, hogy az elektronikus információ-feldolgozás is beleillik ebbe a sorba, jól bizonyítja az, hogy az elektronikus számítógépek harmadik generációja mellett még használták a Hollerith-féle lyukkártyákat.

## Felhasználás

A rögzített információ tehát elérhető a kódolását ismerők számára. Ahogy gyarapszik azonban a rögzített adatok mennyisége, úgy válik egyre nehezebbé az azok között való eligazodás. Az információnak csak akkor vehetjük hasznát, ha szükség esetén megtaláljuk azt.

E probléma megoldására szinte az írásbeliség kezdete óta készültek a rögzített anyagokról különböző kivonatok és katalógusok, melyek megkönnyítették a szükséges információk gyors megtalálását.

A technika fejlődése lehetővé, az adatok számának szaporodása pedig szükségessé tette, hogy a felhasználó rendelkezésre álló információ tömegből ne csak egy-egy ott található tényt keressünk meg, hanem a tények között válogatni, azokat rendezni és csoportosítani, sőt közöttük összefüggéseket felismerni is képes legyen.

A már említett Hollerith-féle géppark lehetővé tette a lyukkártyák mechanikus úton való válogatását, rendezését, stb. Figyeljünk fel arra, hogy a kódolás megválasztása igen erősen befolyásolta a feldolgozás lehetőségeit: a kártyákon lévő lyukak elhelyezkedése volt a mechanikus válogató, rendező berendezések működésének alapja: nevezetesen az, hogy vékony fémrudacskákat sikerült-e az egyes kártyák bizonyos pontjain átdugni, vagy nem.

## Járulékos problémák

### AZ ADATOK HELYESSÉGE

A kezdeti idők „informatikusai” nagyon kevés gondot fordítottak adataik helyességének, pontosságának ellenőrzésére. Még azt is csak egyes vallási tárgyú iratok esetén tartották fontosnak, hogy a másolatok hűen kövessék az eredetit (igaz, itt viszont még a nyilvánvaló hibákat sem javították ki).

### TITKOSSÁG

Arra viszont igen korán rádőbrentek az írástudók, hogy az információ hatalom, s azt féltékenyen őrizni kell. Ez akkor is igaz, ha a történelem folyamán hosszú ideig maga az írás-olvasás is olyan tudomány volt, amivel csak kevesen rendelkeztek, s így az információ bizonyos fokig eleve védett volt.

Az első biztonsági eljárások között megtaláljuk a „fizikai védelem” ötét (az iratokat elzárták biztos helyre) éppúgy, mint a kódolást. Ez utóbbira példaként később, a biztonsággal foglalkozva, majd megemlítjük Julius Caesar kódolási módszerét.

### **ADATÁTVITEL**

Az információ továbbítása is igen régi feladat, sajnos a lovas futárok esetén bevált módszerek nem alkalmazhatók az elektronikus adatátvitel esetén, így azok taglalásától eltekintünk.

## **Információ-feldolgozás számítógéppel**

A mai értelemben vett információ-feldolgozásról – az előző pontban mondtak ellenére – azóta beszélhetünk, amióta számítógépekkel rendelkezünk e munka elvégzésére. Ez akkor is igaz, ha tudjuk, hogy a már említett Hollerith-féle kártya szabvány még az 1980-as években is használatban volt, az 1970-es években pedig még mechanikus Hollerith gépekkel is lehetett találkozni Magyarországon.

Ugyanakkor az első generációs számítógépekre nem csak az elektroncsövek, hanem a kis tárolókapacitás mellett az is jellemző volt, hogy azokat csak erre specializálódott szakemberek tudták használni. Nem csoda, hogy ezeket a gépeket elsősorban numerikus műveletekre alkalmazták. (Erre a célra is fejlesztették őket ki.) A második generációs gépek a fordítóprogramok révén már szélesebb felhasználói kör számára voltak elérhetőek, s a már valamivel nagyobb kapacitású háttértárolóik lehetővé is tették az első információ-feldolgozó rendszerek megjelenését. Mindez fokozottan igaz a harmadik számítógépes generációra (operációs rendszerek megjelenése). Nem véletlen, hogy azt a mesterséget, melyet eleinte számítástechnikának neveztek, ma (legalábbis nálunk) informatikának hívják: napjainkban a számítógépek fő felhasználási területe az információ-feldolgozás. Mára az információ feldolgozása igazi iparággá vált.

Az információ feldolgozása különféle szempontból lehetséges. A korai időkben két gondolkodásmód hódított. A hagyományokhoz leginkább kötődő gondolkodásmód az úgynevezett folyamat-szemléletű, a későbbi technikai lehetőségek terméke pedig az adatbázis-szemléletű információ-feldolgozás.

## 2. FEJEZET

# A folyamat-szemléletű információ-feldolgozás

A folyamat-szemléletű információ-feldolgozás látásmódjának lényege az, hogy mindenekelőtt rögzítjük a feldolgozás célját (ezért beszélnek gyakran célorientált feldolgozásmódról is), majd a feldolgozásban szereplő egyes *egyedekkel* (entities) kapcsolatos adatokat egy-egy feljegyzésben, *rekordban* (record) írjuk le (az ilyen rekordok egy készletét *állománynak*, azaz file-nak nevezzük), s a feldolgozás során az egyes állományok rekordjainak rendezésével, válogatásával igyekszünk a kitűzött célt elérni. Az állományok csak adatokat tartalmaznak: kapcsolatukat a folyamat írja le. Nem nehéz felismerni ebben a gondolkodásmódban (különösen a hozzá tartozó angol terminológiában) a cédulakatalógusok feldolgozásában szerzett tapasztalatokat.

A tervezés sorrendje tehát:

cél  $\rightarrow$  folyamat  $\rightarrow$  állományok.

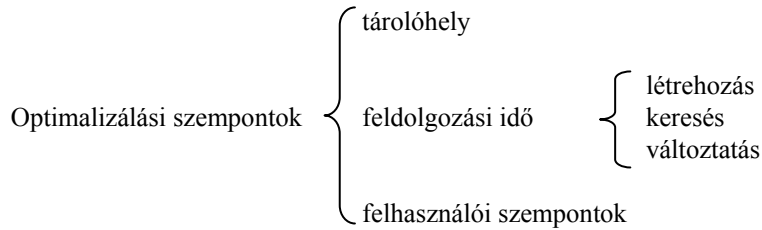
Az ilyen módon megtervezett adatállományok nem igényelnek túl nagy tárolókapacitást, hiszen csak a valóban felhasználandó adatokat kell tárolnunk. Másrészt az adatállományok feldolgozása igen gyors lehet, hiszen azok szerkezete optimálisan illeszkedhet az adott feladathoz. (Tisztázandó persze, hogy mit tekinthetünk optimális illeszkedésnek.) A leírt tulajdonságok indokolják, hogy miért használták az első információ-feldolgozó rendszerek ezt a formát, s hogy miért használják még napjainkban is kisebb méretű, alkalmi feladatok esetén.

Ugyanakkor a folyamat-szemléletű információ-feldolgozás minden célhoz új folyamatot és állományokat igényel, függetlenül az esetleges közös adatok előfordulásától. Az így keletkező *redundancia* rossz tároló-kihasználást eredményez, és – ami még nagyobb baj – felidéri az *inkonzisztencia* veszélyét, vagyis azt, hogy egy-egy több helyen előforduló adat egyes példányainak különbözik a tartalma. Hátránya az effajta feldolgozásnak az is, hogy csak az eredeti kérdésre ad választ, a rendszernek *ad hoc* jellegű kérdések nem tehetők fel.

## Optimális szerkezetű állományok

A programozással foglalkozók jól tudják, hogy egy olyan egyszerű kérdésre: mi a jó program ismérve, vagy két azonos funkciójú program közül melyik a jobb, nem lehet egyszerűen válaszolni. El kell döntenünk, milyen szempontból akarjuk a programokat összehasonlítani.

Ha az információ-feldolgozó rendszereket vizsgáljuk, a helyzet ugyanilyen bonyolult. Információs rendszerek esetén az alábbi optimalizálási szempontokat vehetjük figyelembe:



Könnyen belátható, hogy a felsorolt szempontok egyidejűleg nem elégíthetők ki maradéktalanul, bármelyiket választjuk is ki az optimalizálás alapjául, az így adódó megoldás a többi szempont szerint messze lesz az optimálistól. Így például a tárkihasználás szempontjából ideális tömörített állományok feldolgozása nyilvánvaló többletidőt igényel, a könnyen kezelhető felhasználói felületek mind tár, mind idő tekintetében igen sokat követelnek.

De ellentmondanak egymásnak az időre optimalizálás „szempontjai” is: nyilvánvalóan például, hogy a keresést jelentősen meggyorsítja, ha egy állomány strukturált (mondjuk a keresési kulcs szerint rendezett), ugyanekkor a rendezettség megteremtése a létrehozáskor, megtartása új rekordok beszúrásakor komoly időigénnyel járhat.

Az elmondottak ellenére általában kijelenthetjük, hogy a modern információs rendszerekben kiemelt szerepe van a *keresésnek*. Egyfelől a nagy hálózatok korában az időtényező többnyire szorítóbb, mint az elosztott információk helyigénye, másfelől az időre való optimalizáláson belül elmondhatjuk, hogy egy-egy változtatást (adatbeszúrás, törlés) megelőz az illető rekord keresése (van-e ilyen, s ha igen, hol). Ha mindehhez hozzávesszük, hogy egy-egy állományban igen sokszor kell keresnünk, nem meglepő a keresés kiemelt szerepe.

A keresés időigényét két tényező szabja meg, az egyik az átlagosan szükséges lépésszám (hány keresési lépést kell végrehajtanunk átlagosan egy-egy keresés alkalmával), a másik az egyes lépések időigénye.

Ha egy  $N$  elemű állományban az  $i$ -edik rekordot  $p_i$  valószínűséggel keressük, és  $S_i$  lépés után találjuk meg (vagy derítjük ki, hogy a keresett rekord nincs az állományban), a keresés időigénye:

$$T_N = \sum_{i=1}^N p_i \sum_{j=1}^{S_i} (T_{i,j}^A + T_{i,j}^B + T_{i,j}^C).$$

Itt  $T_{i,j}^A$  az  $i$ -edik rekord keresésekor a  $j$ -edik lépéshez szükséges helymeghatározó algoritmus ideje,  $T_{i,j}^B$  az ennek a rekordnak a tárból való behozásához szükséges idő, végül  $T_{i,j}^C$  annak az összehasonlításnak az ideje, mely eldönti, hogy az éppen vizsgált rekord-e a keresett, vagy sem.

Ha feltesszük, hogy a rekordokat azonos gyakorisággal keressük, és a behozási idők helyett egy átlagértékkel számolunk, akkor egy keresés időigénye:

$$T_N = S_N (T^A + T^B + T^C)$$

A képletben  $T^A$  az átlagos algoritmusidőt jelenti,  $T^B$  az átlagos behozási idő,  $T^C$  az átlagos összehasonlítási idő, végül  $S_N$  az átlagosan szükséges lépésszám. (Az irodalomban megszokott módon  $S_N$ -nel a sikeres, míg  $S'_N$ -vel a sikertelen esetek átlagos lépésszámát jelöljük.) Belátható, hogy központi tár esetén elesik a  $T^B$  tag, míg az általánosan használt mozgófejes

lemezeknél  $T^A$  és  $T^C$  elhanyagolható  $T^B$ -hez képest. Másrészt míg a  $T^C$  értéket nem tudjuk befolyásolni,  $T^A$ ,  $T^B$  és  $S_N$  értéke a választott algoritmustól függ, melyek körét az állomány szervezési módja, szerkezete határozza meg.

Például ha egy adott nevű személy létező rekordját akarjuk egy rendezetlen állományból kikeresni, az úgynevezett lineáris keresést alkalmazhatjuk, azaz sorban meg kell vizsgálnunk a rekordokat, hogy azonosak-e az általunk keresettel. Ez esetben átlagosan a rekordok felét kell megvizsgálnunk (azaz  $S_N \approx N/2$ ). Ha azonban az illető személy rekordja nem található meg az állományban, ennek megállapításához már az egész állományt végig kell keresnünk (vagyis  $S'_N \approx N$ ). Ha az állomány a névre rendezett, az utóbbi esetben is elég átlagosan a rekordok felét megvizsgálni ( $S'_N \approx N/2$ ). Így a struktúra megváltoztatása önmagában gyorsított az eljárás. Más kérdés, hogy az ilyen módon strukturált állományokban már sokkal hatékonyabb keresési algoritmusok használatára is van remény.

## Legfontosabb állomány struktúrák

A folyamat-szemléletű információ-feldolgozás során a leggyakrabban használt állomány struktúrákat a következőképpen csoportosíthatjuk:

- *Szekvenciális állomány struktúrák*,  
melyeknél logikailag a rekordok mindegyikének egy megelőzője és egy rákövetkezője van (az első és az utolsó elem kivételével).
- *Hierarchikus állomány struktúrák*,  
melyeknél logikailag a rekordok mindegyikének (az első elem kivételével) egy megelőzője, de esetleg több rákövetkezője van.
- *Hálós állomány struktúrák*,  
melyeknél logikailag a rekordok mindegyikének több megelőzője és több rákövetkezője lehet.
- *Nem konzekutív állomány struktúrák*,  
ahol nem a rákövetkezés ill. megelőzés adja az állomány struktúráját, hanem más logikai szempont szerint lehet a rekordokat ill. a rekordok egy csoportját kiválasztani.

## Szekvenciális állomány struktúrák

### Fizikai szekvenciális állomány struktúrák

*Fizikai szekvenciális struktúráról* akkor beszélhetünk, ha a rekordok logikai és fizikai sorrendje megegyezik. E szervezési forma legfontosabb előnye, hogy a természetesen itt is alkalmazható lineáris keresés mellett annál jóval hatékonyabb keresési eljárások alkalmazását is lehetővé teszi.

### **BINÁRIS, LOGARITMIKUS KERESÉS**

E közzismert keresési eljárás lényege, hogy a vizsgálandó állomány, vagy állományrész középső rekordjának kulcsát hasonlítjuk össze a keresett rekordéval, s így eldönthetjük, hogy

- megtaláltuk a keresett rekordot,

- a keresett rekord a vizsgált állomány jobb felében van,
- a keresett rekord a vizsgált állomány bal felében van,
- ha az állomány tovább nem felezhető, a keresett rekord nincs meg benne.

Ezután a keresést már csak az előző lépésben vizsgált állomány felében kell folytatnunk (ha egyáltalán szükséges).

Belátható hogy bináris keresés esetén az átlagosan szükséges lépésszám mind sikeres, mind sikertelen esetben:

$$S_N \approx \log_2 N.$$

### PETERSON-FÉLE KERESÉS

Az előző módszer továbbgondolásából adódik ez a módszer. A gondolatmenet lényege az, hogy az állományt az egyes lépésekben ne két, a rekordok számára nézve egyenlő részre osszuk, hanem két olyan részre, melyekben a keresett rekord előfordulása egyenlően valószínű. (Például egy nevekre rendezett nyilvántartásban Ács Ába rekordját nyilván az állomány elején, Zöld Zoltánét a végén célszerű keresni.) Az ilyen osztáshoz azonban ismerni kellene a kulcsok eloszlását. Feltehetjük például, hogy azok eloszlása egyenletes. Ekkor egyszerű arányosítással meghatározható a keresett rekord legvalószínűbb helye (és egyúttal a kívánt osztáspont): ha  $K_E$  az első és  $K_U$  az utolsó kulcs, továbbá  $A_E$  az első és  $A_U$  az utolsó cím, a  $K$  kulcsú rekord becsült  $A$  helye:

$$A = A_E + \frac{A_U - A_E}{K_U - K_E} K.$$

Bebizonyítható, hogy ebben az esetben az átlagosan szükséges lépésszám:

$$S_N \approx 1/2 \log_2 N.$$

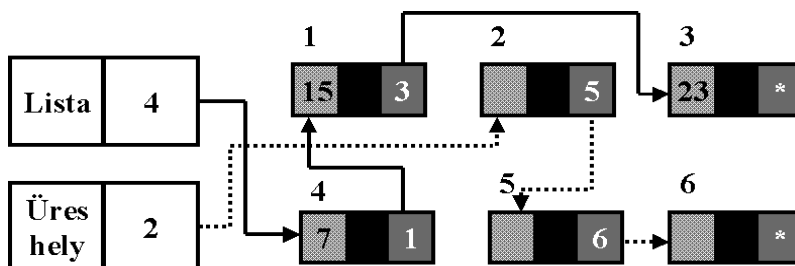
Mindebből nem következik azonban, hogy a Peterson-féle keresés mindig hatékonyabb a binárisnál. Ennek egyik oka az, hogy kulcsaink eloszlása messze eshet az egyenletestől, s ilyenkor a módszer hatékonysága jelentősen romlik. A másik ok, hogy a Peterson módszer minden lépése arányosítást, azaz „igazi” osztást igényel. A bináris keresésnél erre nincs szükség, az itt használt felezés a rendkívül gyors léptetéssel megvalósítható. Mivel központi tárbeli használatnál az egyes lépések időigényét az algoritmusidő határozza meg, hiába igényel a Peterson-féle keresés kevesebb lépést, ha azok jóval lassabban hajthatók végre.

Bármilyen gyorsak a fizikai szekvenciális állomány struktúrájánál használható keresési eljárások, elhamarkodott kijelentés lenne azt állítani, hogy az ilyen állományok feldolgozása minden  $N > 4$  esetén gyorsabb, mint a struktúrával nem rendelkezőké. Az állomány kezelésekor ugyanis a keresésre fordított időn kívül figyelembe kell venni azt az időt is, melyet a karbantartásra, azaz a rendezettség megtartására kell fordítanunk. Mindezt összevetve azt mondhatjuk, hogy fizikai szekvenciális struktúra használata elsősorban a statikus állományoknál javasolt, azaz ott, ahol egy-egy – struktúrát érintő – változtatásra (pl. beszúrás) sok keresés esik.

### Logikai szekvenciális állomány struktúrák

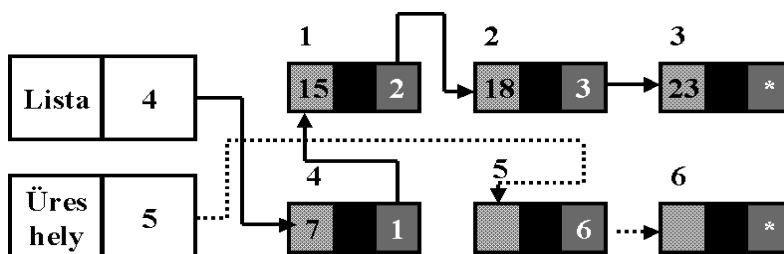
*Logikai szekvenciális struktúráról* akkor beszélünk, ha a rekordok logikai és fizikai sorrendje nem egyezik meg. A logikai sorrendet ez esetben legtöbbször a rekordokban elhelyezett mutatók (pointerek) adják meg. (Meg lehetne adni a logikai sorrendet valamilyen külső

táblázat segítségével is, ezt azonban a szakirodalom általában az indexelt szervezés részeként tárgyalja.) A mutatórendszerek lehetnek egy és kétirányúak, ill. gyűrűsek. Az állomány és a sokszor ugyancsak speciális listaként kezelt üres helyek kezdetét egy úgynevezett indító tábla tartalmazhatja (1. ábra).



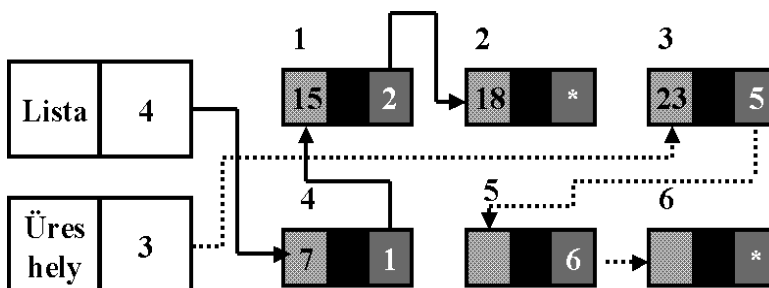
1. ábra Logikai szekvenciális szervezés

E szervezési mód legfőbb előnye, hogy alkalmazásakor az állomány logikai struktúrájának megváltoztatása igen könnyű, nem igényli a rekordok mozgatását, csupán a mutatók módosítása szükséges (2. és 3. ábrák). További jelentős előny, hogy fizikailag ugyanaz az állomány több mutatórendszer alkalmazásával többféle logikai szekvenciális rendbe is szervezhető. (E logikai szekvenciáknak nem kell minden esetben valamennyi rekordot tartalmazniuk.)



Új elem: 18

2. ábra Beszúrás logikai szekvenciális állományba



Törlendő: 23

## 3. ábra Törlés logikai szekvenciális állományból

A felsorolt előnyök mellett a logikai szekvenciális szervezésnek komoly hátrányos tulajdonságai is vannak. Elsősorban azt kell megemlítenünk, hogy e szervezési forma mellett nem állnak rendelkezésünkre olyan hatékony keresési algoritmusok, mint a bináris, vagy a Peterson-féle. Nem kisebb probléma, hogy mozgófejes lemez használata esetén – ha a rekordok véletlenszerűen helyezkednek el az állományban – a keresés során szükséges pozicionálások száma a feldolgozást nagyon lelassítja. (Egy  $C$  cilindert elfoglaló állomány esetén két rekord elérése között átlagosan  $C/3$  cilindernyi pozicionálásra van szükség.)

A fenti hátrányok csökkentésére használhatóak az úgynevezett „csaknem fizikai szekvenciális” struktúrák. E struktúrák lényege, hogy az állományt (mely logikai szekvenciális, így mutatókkal rendelkezik) a létrehozásakor a mágneslemezen fizikailag is szekvenciálisan helyezük el. Beszúrásakor nem szükséges a fizikai szekvencialitást megtartani, az új rekordot a – minden cylinderen külön-külön fenntartott – úgynevezett cylinder túlcsoportulási területre helyezük, és a logikai rendbe a mutatók segítségével illesztjük be.

Látható, hogy így elkerülhetjük a sok pozicionálásból adódó idővesztést. Ezen túlmenően azonban e szervezési formának az is előnye, hogy mellette alkalmazhatjuk a *kupacos* keresési módszert. E keresési módszer lényege, hogy a rekordokat kupacokra osztjuk, majd először e kupacok első elemeit hasonlítjuk össze a keresett rekorddal. Így vagy megtaláljuk a keresett rekordot, vagy legalább meghatározhatjuk, hogy melyik kupacban van. Ezután ezt a kupacot kell végigkeresnünk. Ha az egyes kupacok kezdőelemei az elsődleges (nem a túlcsoportulási) területen vannak, a keresés elég gyors lehet. Természetesen kulcskérdés a kupacok méretének megválasztása. Ha az állományban  $N$  rekord van, és azt úgy osztjuk fel kupacokra, hogy egy-egy kupacba  $G$  rekord kerüljön (esetleg az utolsót kivéve), azaz összesen közelítőleg  $N/G$  kupac legyen, akkor a szükséges lépésszám:

$$\sum_{k=1}^{\frac{N}{G}} k \left( \frac{1}{\frac{N}{G}} \right) \approx \frac{\frac{N}{G} + 1}{2},$$

hiszen a csoportok között, és a kiválasztott csoporton belül is egy-egy lineáris keresést hajthatunk végre (a kiválasztott csoporton belül eggyel kevesebb elemmel számolhatunk, mivel az első elemet már megvizsgáltuk). Így

$$S_N \approx \frac{\frac{N}{G} + 1}{2} + \frac{G - 1}{2} = \frac{1}{2} \frac{N}{G} + \frac{G}{2}.$$

Ez az érték azonban nem csak  $N$ -től, hanem  $G$ -től is függ. Annak megvizsgálására, hogy milyen  $G$  érték mellett lesz minimális, deriváljuk a képletet  $G$  szerint, és keressük a derivált 0 helyét. Ekkor az

$$\frac{N}{G^2} = 1$$

értékhez jutunk, amiből egyrészt

$$G = \sqrt{N},$$

másrészt (az eredeti képletbe visszahelyettesítés után)

$$S_N \approx \sqrt{N}$$

adódik. Természetesen az egyes kupacok mérete általában nem választható pontosan a megadott képlet szerint (semmi sem garantálja ugyanis, hogy  $N$  négyzetszám legyen), a függvények folytonossága miatt azonban az ehhez közeli kupacméret ehhez közeli lépésszámot eredményez.

Az eddigiekben mindig feltételeztük, hogy az egyes rekordokat azonos gyakorisággal keressük vissza. (Ezt a soron következő bekezdés kivételével a továbbiakban is feltesszük majd.) A logikai szekvenciális szervezés azonban lehetőséget ad jelentősen különböző keresési gyakoriságú rekordok olyan szekvenciába szervezésére, melynél a sorrendet a keresési gyakoriság csökkenő sorrendje szabja meg. Ilyen szervezés fizikai szekvenciális struktúrában is lehetséges, a logikai szekvenciális struktúra esetén azonban alkalmazhatóak az úgynevezett önrendező algoritmusok is, melyek a gyakoriság ismerete nélkül, ill. változó gyakoriság mellett is lehetővé teszik a gyakoriság szerinti szervezést. (A legegyszerűbb ilyen algoritmus minden rekordot annak keresése után az első helyre csatol. Ha nem is érhető így el pontos gyakoriság szerinti rendezés, azt ez az egyszerű eljárás is biztosítja, hogy a gyakran használt rekordok mindig az állomány elején legyenek találhatóak.)

## Hierarchikus állomány struktúrák

### A hierarchikus állományokat kezelő eljárások

A hierarchikus (faszerű) struktúrák számítógépes ábrázolására egyaránt elterjedtek mind a belső mutatós (pointeres), mind a külső mutatós (táblázatos) eljárások.

#### **BELSŐ MUTATÓS ELJÁRÁSOK**

- *Visszavezetés szekvenciális struktúrákra.*  
E módszereknél a fának egy megadott módon való bejárását rögzítjük. A legjellegzetesebb ilyen módszer az úgynevezett left-list módszer. Sajnos az eljárás információvesztéssel jár: az eredeti fastruktúra nem állítható vissza belőle egyértelműen.
- *Több mutató használata.*  
A rekordokban lehetséges annyi mutatót elhelyezni, ahány rákövetkezője (gyermeke) van az illető rekordnak. Az ábrázolás teljesen egyértelmű, hátrányos tulajdonsága viszont, hogy – mivel a rákövetkezők száma tág határok között változhat – vagy igen rossz tároló kihasználással, vagy változó hosszúságú rekordok használatával kell számolnunk.
- *Külön kapcsolórekordok használata.*  
Látszólag bonyolítja a helyzetet, ha az információt hordozó rekordokon kívül egy újabb rekordtípust vezetünk be, mely csak a kapcsolatok leírására szolgál. Ugyanakkor ezzel a megoldással elérhető, hogy egy-egy rekordban (a kapcsolórekordokat is beleértve) legfeljebb két mutató legyen, sőt az is, hogy ezek egyike „oldalra”, másika „lefelé” mutasson, ami a struktúrát igen áttekinthetővé teszi.
- *Gyűrűk alkalmazása.*  
Szokás a fában „függőleges” és „vízszintes” gyűrűket szervezni, melyekkel ugyan csak megoldható, hogy egy-egy rekord ne tartalmazzon kettőnél több mutatót.

## KÜLSŐ MUTATÓS ELJÁRÁSOK

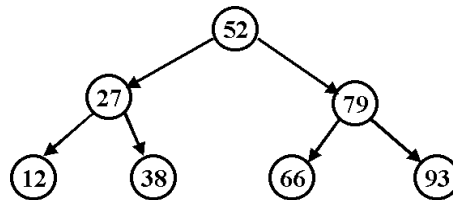
- *Táblázatok használata.*  
E táblázatokban az egyes rekordokhoz tartozó bejegyzések adják meg a rákövetkezőket (esetleg a megelőzőt is). Mivel a rákövetkezők száma itt is erősen változhat, a táblázatok ábrázolásánál ugyanazokkal a problémákkal kell szembenéznünk, mint a több mutatós rendszer alkalmazásánál.
- *Bináris mátrixok használata.*  
Bináris mátrixokban ábrázolhatjuk azt, hogy egyes rekordok között van-e kapcsolat. A bináris jellegen kívül az teszi a – nagyméretű mátrixokat használó – módszert használhatóvá, hogy hierarchikus állományoknál elég a mátrix felét tárolni (a másik félmátrix csak 0 elemeket tartalmaz).

## Kiemelten fontos hierarchikus struktúrák

Az alábbiakban két, különösen fontos hierarchikus struktúrát külön is ismertetünk. Jelentőségük elsősorban az indexelt struktúráknál (lásd később) lesz.

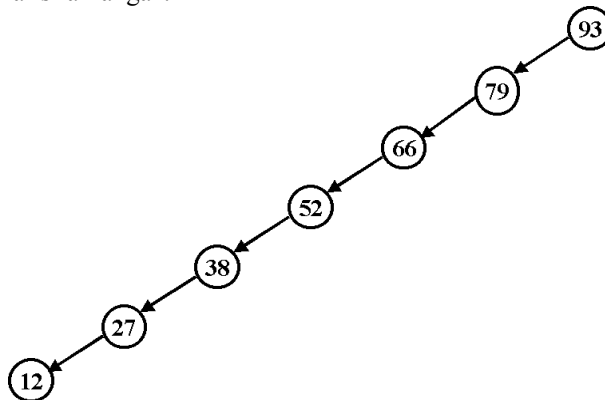
### BINÁRIS FÁK

Azokat a fákat, melyekben minden elemnek legfeljebb két gyermeke van, bináris fának nevezzük. Bizonyos feltételek teljesülése esetén az ilyen fákból a keresés igen gyors lehet. Nézzük a 4. ábrán látható példát:



4. ábra Bináris, teljesen kiegyensúlyozott fa

Könnyen belátható, hogy egy ilyen felépítésű fában legfeljebb  $\log_2 N$  lépésben célhoz ér minden keresés (lényegében egy bináris keresésről van szó). Ez azonban csak az „elégé szimmetrikus” bináris fákra igaz.



## 5. ábra Nem kiegyensúlyozott bináris fa

Vizsgáljuk meg az előző példával azonos elemeket tartalmazó, ugyancsak bináris fát (5. ábra). Ez a fa csak egyféleképpen járható végig, s könnyen látható, hogy itt a lineáris keresésre jellemző  $N/2$  lépésre lesz szükség.

Sajnos, a bináris fák „szimmetrikus” voltát (amit a szakirodalom kiegyensúlyozottnak, vagy teljesen kiegyensúlyozottnak hív) egy-egy beszúrás után rendkívül bonyolult biztosítani. (A kiegyensúlyozottság pontos definíciója és a bináris fák kezelésére szóló algoritmusok leírása megtalálható [Wi82]-ben.)

**B-FÁK**

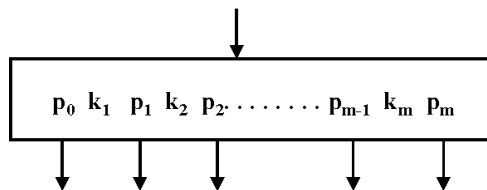
Ezt az állomány struktúrát R. Bayer dolgozta ki 1970-ben (ezért nevezik a struktúrát B, azaz Bayer fának, ami nem tévesztendő össze a bináris fákkal). Az állomány úgynevezett lapokból áll, egy-egy lapon rekordok és mutatók helyezkednek el. Az állomány jellemzői közé tartozik egy  $n$  szám, melyet a fa rendjének nevezünk.

A fa egy-egy lapján mutatók ( $p_i$ ) és rendezett kulcsok ( $k_i$ ) helyezkednek el (a kulcsokhoz természetesen tartozhatnak adattételek, de a struktúra szempontjából azoknak nincs jelentőségük). A lap első, ill. utolsó mutatója egy-egy olyan lapra mutat, melyen az adott lapnál kisebb, ill. nagyobb kulcsok találhatók, a többi  $p_i$  mutató esetében a hivatkozott lapon (és annak utódain) a kulcsok  $k_i$  és  $k_{i+1}$  közé esnek. A levéllapokon a mutatók értéke NIL (nem mutatnak sehová, azaz egy levéllap  $k_i$  és  $k_{i+1}$  kulcsai közé eső kulcs nincs az állományban).

Egy B-fa a következő kritériumoknak kell eleget tennie:

1. Minden lap legfeljebb  $2n$  tételt tartalmaz.
2. Minden lapon – a gyökér kivételével – legalább  $n$  tétel van.
3. Ha egy lapon  $m$  tétel van, az vagy levéllap, vagy  $m+1$  utódja van.
4. Minden levéllap ugyanazon a szinten van.

A kereséshez szükséges átlagos lépésszám elsősorban a laphivatkozásoktól függ. Belátható, hogy ezek száma  $N$  elemnél egy  $n$  rendű fában:  $\log_n N$

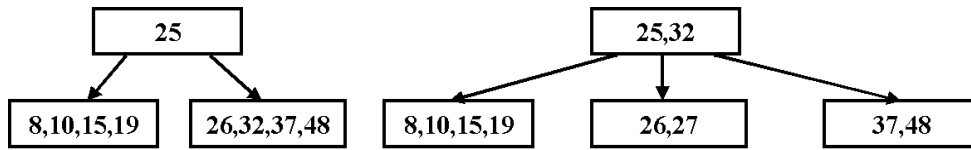


6. ábra Egy B-fa egy lapja

A B-fák használatának legnagyobb előnye, hogy a fent leírt struktúra az állomány megváltoztatásakor is viszonylag könnyen megtartható. Ha egy rekordot akarunk beszúrni az állományba, megtehetjük a megfelelő levéllapon. Ha ezek után a lap elemeinek száma több lenne, mint  $2n$ , a lapról kiemelve és eggyel magasabb szintre emelve a középső elemet, a maradékot két, egyenként  $n$  elemű lapra oszthatjuk. Ha a kiemelt középső elem sem „fér el” az új lapon, az eljárást megismételhetjük, akár egészen a gyökérlapig, ha pedig a gyökérlap is több mint  $2n$  elemet tartalmazna, egy új – esetleg egy elemű – gyökérlapig. Mindezt a 7. ábrán láthatjuk.

Ha törölni akarunk egy elemet, az eljárás az előbb leírtak megfordítása lehet, ennek részleteit az érdeklődők a [Wi82] könyvben találhatják meg.

A B-fákat használhatjuk adattárolásra is, elsősorban nem túl sok, és rövid rekord feldolgozásakor (ellenkező esetben a lapok mérete, ill. a szintek száma olyan nagy lenne, hogy megakadályozná a hatékony feldolgozást). Gyakoribb, hogy B-fa, ill. annak egy változata, a B+-fa formájában úgynevezett index állományokat tárolunk (lásd később).



7. ábra A 27 kulcsú elem beszúrása

## Hálós állomány struktúrák

### A hálós állományokat kezelő eljárások

#### BELSŐ MUTATÓS ELJÁRÁSOK

- *Visszavezetés a hierarchikus struktúrákra.*  
Megkíséreljük a hálós struktúrákat hierarchikus struktúrákra visszavezetni. Ez esetben úgy kerülhetjük el az egyes elemek többszörös tárolását, hogy a többszöröződő rekordok helyett – egy kivételével – úgynevezett virtuális elemeket alkalmazunk (azaz olyan elemeket, melyek adatokat nem, csak azok helyére való utalást tartalmaznak).
- *Több mutatós eljárások.*  
Ezek az eljárások ugyanúgy használhatók, mint a hierarchikus esetben, az előnyök és a hátrányok is ugyanazok lesznek. Itt azonban nem alkalmazhatóak azok az eljárások (külön kapcsolórekordok, gyűrűk), melyek kihasználják a szintek létezését: a „vízszintes” és a „függőleges” fogalmát.

#### KÜLSŐ MUTATÓS ELJÁRÁSOK

- *Táblázatok használata.*  
A táblázatok alkalmazását illetően semmi lényeges különbség sincs a hálós és a hierarchikus eset között.
- *Bináris mátrixok használata.*  
A bináris mátrixok is ugyanúgy használhatók, mint a hierarchikus állományoknál, itt azonban általában nem elegendő a fél mátrix tárolása.

## Nem konzekutív állomány struktúrák

### Indexelt állomány struktúrák

#### A SŰRŰ INDEXELÉS

E szervezési formánál az adatállomány minden tételéhez tartozik az index állományban egy index bejegyzés: egy kulcs – cím pár.

Ekkor több szempont alapján több index is szervezhető, az adatállományhoz új rekordok könnyen fűzhetők (az állomány végére), az index bejegyzés ismeretében a keresés gyors. Ugyanakkor nagyméretű index állományokban kell keresni, és azokat karban is kell tartani.

Alapfeladat tehát az index állomány szerkezetének helyes megválasztása. Sok szempontból a korábban már ismertetett úgynevezett bináris fák (olyan fák, melyek minden elemének legfeljebb két gyermeke van) használata tűnik előnyösnek. Bizonyos struktúrájú bináris fák esetében a keresés igen gyors benne: közelítőleg  $\log_2 N$  lépésre van szükség átlagosan, azonban a szükséges struktúra megtartása új elemek beszúrásánál igen időigényes.

A gyakorlatban index állományok készítésére elsősorban az úgynevezett B-fákat, pontosabban azoknak egy változatát, az úgynevezett B+-fákat használják. (Ilyen struktúrát alkalmaz – sok más rendszerrel együtt – az Oracle rendszer is.) A B+-fa egy olyan B-fa, melyben azok a pointerok, melyek adatokra (adatok címére) mutatnak, s amelyeket keresünk, valamennyien a levéllapokon helyezkednek el, így a levéllapok és a többi lap struktúrája különbözik. Az ilyen állományok könnyen karbantarthatóak, és a bennük való keresés elég gyors. (A B-fákról is szoltunk már az előzőekben, a B+-fákról olvashatunk pl. [EN94]-ben.)

### **A RITKA INDEXELÉS**

E szervezési formánál nem minden rekordhoz, csak bizonyos jelzőpontokhoz tartozik bejegyzés. Természetesen az eligazodáshoz ez esetben valamilyen rendezettség szükséges. Az indexek általában több szintűek, az alsó szintű indexekben (sok bejegyzés) való eligazodást újabb ritka index segítheti. Jól példázza a ritka indexelést a szokásos telefonkönyv.

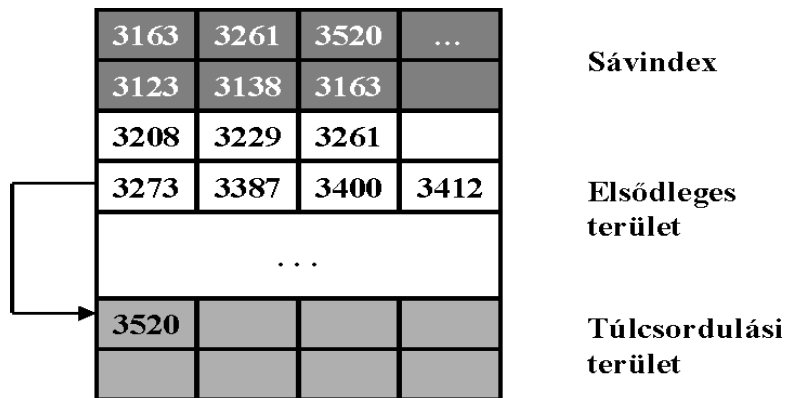
Ilyen szervezés esetén igen gyors a keresés, és az állományba aránylag könnyen lehet új rekordokat beszúrni, ám ilyen index rendszer csak egy szempont alapján szervezhető.

### **AZ INDEX-SZEKVENCIÁLIS SZERVEZÉS**

A ritka indexelés egyik legnépszerűbb megvalósítása az index-szekvenciális szervezés. Több szintű indexet tartalmaz, s mivel kifejezetten mágneslemezre tervezték, így a legalsó index a sávindex. (A sávokat technikailag amúgy sem lehetséges másképpen, mint szekvenciálisan feldolgozni.)

Az index szintek tehát (felülről lefelé):

- fő index,
- cylinder index,
- sáv index.

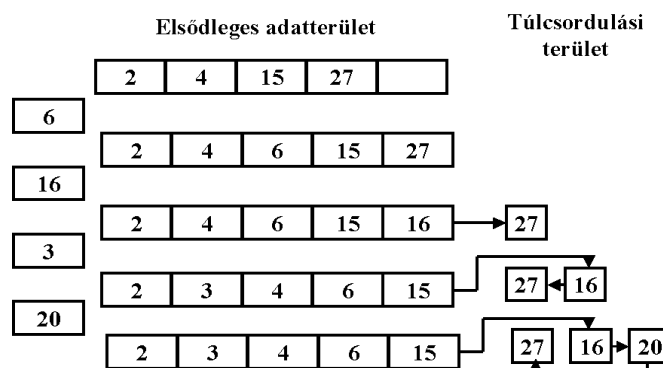


8. ábra Index-szekvenciális állomány felépítése

Minden egyes cilindert három részre osztanak, hogy az azon levő területeket fejmozgás nélkül lehessen elérni. A lemez felosztása:

- index terület,
- elsődleges adatterület,
- túlsordulási terület.

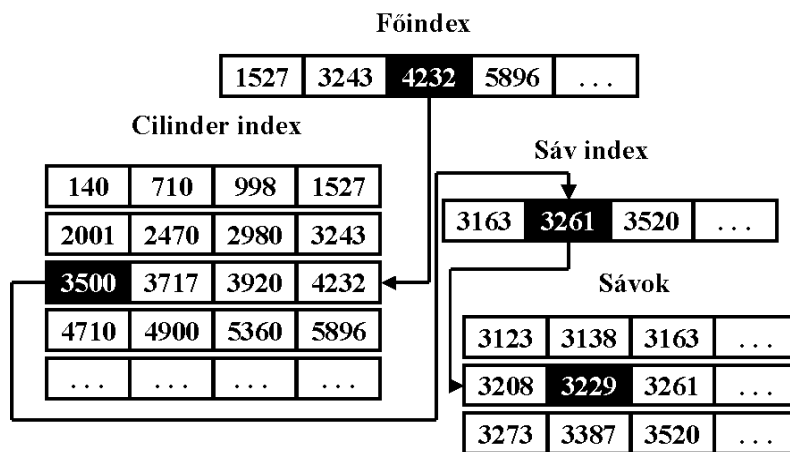
Mindezt a 8. ábra mutatja:



9. ábra Az elsődleges és a túlsordulási terület használata

Az állomány létrehozásakor a rekordok az elsődleges adatterületre kerülnek, melyet csak részben töltenek fel. A feltöltés rendje fizikai szekvenciális. Ekkor rögzítik az indexeket. Ha az elsődleges adatterület betelik, a rekordok a túlsordulási területre kerülnek, mutatókkal ellátva, azaz logikai szekvenciális rendben (9. ábra).

Az index-szekvenciális szervezés legfőbb előnye a hatékony keresési lehetőség. Az adatok keresése az index rendszeren keresztül történik (10. ábra).



10. ábra Keresés index-szekvenciális állományban

Természetesen az index-szekvenciális állományokat időről időre újra kell szerveznünk, azaz valamennyi rekordot az elsődleges területre kell tölteni. Ellenkező esetben a feldolgozás nagyon lelassulhat.

### A direkt állományszervezés

A direkt szervezésnél a rekordok kulcsai és címei között egy leképezés hozza létre a kapcsolatot. Természetesen kíváncsi lenné, hogy ez a leképezés egy-egyértelmű legyen, vagyis, hogy a leképezés ne rendeljen azonos címeket különböző kulcsokhoz. (Látjuk majd, hogy ez a követelmény nem mindig teljesíthető.)

### A LEKÉPEZÉSEK

A fenti követelményeknek eleget tevő leképezést *közvetlen leképezésnek* nevezzük. Erre példa egy olyan elképzelt állomány, mely a teljes magyar lakosság adatait tartalmazza, s melynél a kulcsként használt személyi szám egyben a rekord állományon belüli sorszáma. (Bináris számítógépben ábrázolt kulcsokat minden további nélkül tekinthetünk numerikus értékeknek, s az sem okozhat problémát, hogy címnek az állományon belüli sorszámot tekintjük.) A közvetlen leképezések sajnos többnyire katasztrofálisan rosszul gazdálkodnak a tárterülettel, s így gyakorlatilag használhatatlanok. (Ez azonban nem vonatkozik a sűrűn indexelt állományokra, melyek tekinthetők ilyen szervezésűnek.)

A jobb tárkihasználás érdekében követelményeinkből engedni kényszerülünk. Az úgynevezett *hashing* algoritmusok olyan leképezések, melyek – lehetőleg nem túl sokszor – megengedik, hogy különböző kulcsokhoz ugyanaz a cím tartozzék (ezek az úgynevezett *szinonimok*). Cserébe jobb tárkihasználásra számíthatunk. A tapasztalat szerint 80-85%-os tárkihasználás mellett 20%-nál nem több szinonim elfogadható érték.

A leggyakoribb hashing algoritmusok:

- a *csonkítás*,  
melynél elhagyjuk a kulcs olyan jegyeit, melyek nagy valószínűséggel csak kevés ér-

téket vesznek fel (pl. a már említett személyi számnál a hónapok tizes jegyei, stb.), és a maradék számot tekintjük az állományon belüli sorszámnak,

- *a maradék módszer,*  
melynél a kulcsot egy  $m$  modulussal osztjuk, s a sorszám az osztás maradéka lesz. (Célszerű modulusként prímszámot választani.) E módszer előnyös tulajdonsága, hogy az eredetileg egyenletes elosztást nem rontja el.

## A SZINONIMOK KEZELÉSE

Hashing algoritmusok esetén külön kell gondoskodnunk a szinonimok kezeléséről.

- *A külső láncolás*  
E módszernél egy külön területen tároljuk a szinonimokat, melyeket mutatólánc köt össze az eredeti (a hashing algoritmus által adott) címmel. Mivel előre nem lehet a szinonimok számát tudni, a külön szinonimterület fenntartása akadály a jó tárkihasználásnak.
- *A belső láncolás*  
Itt is mutatórendszer köti össze a szinonimot eredeti címével, azonban azt az elsődleges terület egy üres (és lehetőleg az eredeti címhez közel eső) helyére tesszük. A tárterület kihasználása jó lesz, azonban lehetséges, hogy a szinonim által elfoglalt helyre később „igényt tart” annak „jogos” tulajdonosa, mely rekordot így kénytelenek leszünk „műszinonim”-ként kezelni. Ennek kezelése az állomány feldolgozásakor több lépést igényel.
- *A Peterson módszer (nyílt módszer)*  
A rekordok ugyanott helyezkednek el, mint a belső láncolásnál, azonban nem használunk mutatókat. Az állomány struktúrája egyszerű, feldolgozása azonban több lépést igényel, mintha a belső láncolást alkalmazzuk.
- *Többszörös hashing*  
Ha a kiszemelt hashing algoritmus szinonimra vezet, alkalmazhatunk újabb és újabb ilyen algoritmusokat, míg csak a probléma el nem háruul. (A gyakorlatban két hashing algoritmust használnak, az esetlegesen még maradó szinonimokat az előző módszerek valamelyikével kezelik.)
- *Bugyrok (bucket-ek) alkalmazása*  
Lehetséges az egyes címekhez nem egy, hanem több rekordnyi helyet rendelni. Így kevesebb lesz a szinonimok kezelésére fordítandó idő, ugyanakkor minden keresésnél az érintett bugyrok (pontosabban az azokban szereplő rekordoknak átlagosan a felét) meg kell vizsgálni. A módszer használata akkor indokolt, ha a tároló fizikai tulajdonságai (és a rekordok rövidege) amúgy is csak több rekord egyidejű ki-, ill. bevitelét teszik lehetővé.

## A SZINONIMKEZELŐ MÓDSZEREK ÖSSZEHASONLÍTÁSA

Az egyes módszereknél átlagosan szükséges lépésszámokra egy úgynevezett kitöltési tényező (az állományban ténylegesen található  $N$  és a maximálisan elhelyezhető rekordok  $M$  számainak hányadosa:  $A = N/M$ ) függvényében elég jó matematikai becslés adható, bár ezek a képletek még akkor sem könnyen hasonlíthatóak össze, ha – kihasználva, hogy  $A \leq 1$  – azokat Taylor sorba fejtjük. Tekintsük meg néhány esetben a megadható képleteket.

Belső láncolásnál:

$$S_N = 1 + \frac{1}{8A} (e^{2A} - 1 - 2A) + \frac{1}{4} A = 1 + \frac{1}{2} A + \sum_{i=2}^{\infty} \frac{2^{i-2} A^i}{(i+1)!},$$

$$S'_N = 1 + \frac{1}{4} (e^{2A} - 1 - 2A) = 1 + \frac{1}{4} \sum_{i=2}^{\infty} \frac{(2A)^i}{i!}.$$

Nyílt módszernél:

$$S_N = \frac{1}{2} \left( 1 + \frac{1}{1-A} \right) = 1 + \frac{1}{2} A + \frac{1}{2} A^2 + \dots = 1 + \frac{1}{2} \sum_{i=1}^{\infty} A^i,$$

$$S'_N = \frac{1}{2} \left[ 1 + \left( \frac{1}{1-A} \right)^2 \right] = 1 + A + \frac{3}{2} A^2 + \dots = 1 + \sum_{i=1}^{\infty} \frac{i+1}{2} A^i.$$

Többszörös hashing esetén:

$$S_N = -A^{-1} \ln(1-A) = 1 + \frac{A}{2} + \frac{A^2}{3} + \dots = \sum_{i=0}^{\infty} \frac{A^i}{i+1},$$

$$S'_N = (1-A)^{-1} = 1 + A + A^2 + \dots = \sum_{i=0}^{\infty} A^i.$$

Nem elegendő azonban pusztán a lépésszámokat venni figyelembe, hiszen az egyes lépések időigénye is különböző. E témakörnek jelentős matematikai irodalma van (pl. [Kn88]), itt csak igen durva heurisztikus okoskodásra van módunk.

Központi tárban való használat esetén a többszörös hashing alkalmazásakor előnytelen a jelentős algoritmusidő (egy hashing algoritmus lefuttatása jóval hosszabb, mint egy cím átírása, vagy egy regiszter növelése, amit a láncolás illetve a nyílt módszer igényel). A nyílt módszer több lépést igényel, mint a láncolások, s algoritmusideje azokéval összemérhető. Jóllehet a belső láncolás (a „műszinonimok” miatt) kicsivel több lépést igényelhet a külsőnél, nagyobb súllyal esik latba a jó tárkihasználás. Így általában a belső láncolást tudjuk javasolni.

Mozgófejes lemeznél előnytelenek a sok fejmozgást igénylő módszerek: ilyen a többszörös hashing és a külső láncolás. Bár a belső láncolás elméletileg kevesebb lépést igényel a nyílt módszernél, ha a szinonim rekord ugyanazon a sávon van, mint az eredeti helye, a gyakorlatban ez az előny nem érvényesül. Ugyanakkor, ha a szinonim az eredeti helyet közvetlenül követi (és ez igen gyakori eset), a nyílt módszer gyorsabb is lehet (nem kell a mutatókat feldolgozni). Végeredményben tehát ilyen esetekben többnyire a nyílt módszer javasolható.

### Példák

Képzeljünk el egy mindössze tíz rekordból álló állományt, melyben a rekordok kulcsai rendre 46, 24, 29, 36, 17, 75, 74, 14, 19. Mivel 10 rekordról van szó, válasszuk a leképezés modulusául a 11-et (prím szám, és kevéssel nagyobb a rekordok számánál, így megfelelő tárkihasználást biztosít). A szinonimok kezelésére alkalmazhatjuk akár belső láncolást, akár a nyílt módszert (1. táblázat).

Kulcs	Számított Cím	Tényleges Cím	Mutató	Lépésszám belső lánc.	Lépésszám nyílt
46	2	2	3	1	1

24	2	3	4	2	2
29	7	7	-	1	1
36	3	4	5	2	2
30	8	8	10	1	1
17	6	6	-	1	1
75	9	9	-	1	1
74	8	10	0	2	3
14	3	5	-	3	3
19	8	0	-	3	4
Összesen				17	19

1. táblázat *Direkt szervezés maradék módszerrel, belső láncolással, ill. nyílt módszerrel*

Figyeljük meg, a 74 és a 19 kulcsú rekordok nem közvetlenül a kiszámított utáni címen lesznek találhatóak, így ezek megtalálásához a nyílt módszerrel több lépésre lesz szükség.

Észrevehetjük azt is, hogy a 36 kulcsú rekordot a 24 kulcsú szinonimjaként kell kezel-nünk, jöllehet nem az, de a szinonim kezelő módszer logikája miatt a 24 kulcsú elfoglalja a 36 kulcsú helyét. Ezt az előnytelen helyzetet részben elháríthatjuk az úgynevezett két me-netben való töltéssel, vagyis azzal, ha első menetben csak azokat a rekordokat töltjük be, amelyek „saját helyükre” kerülnek, s a többit csak a második menetben. (Természetesen ez a módszer csak az állomány feltöltésekor használható, később már nem.) Próbálkozzunk e módszer alkalmazásával (2. táblázat)!

Kulcs	Számított cím	Tényleges Cím	Mutató	Lépésszám belső lánc.	Lépésszám nyílt
46	2	2	4	1	1
24	2	4	-	2	3
29	7	7	-	1	1
36	3	3	5	1	1
30	8	8	10	1	1
17	6	6	-	1	1
75	9	9	-	1	1
74	8	10	0	2	3
14	3	5	-	2	3
19	8	0	-	3	4
Összesen				15	19

2. táblázat *A két menetben való töltés*

Láthatóan a módszer a belső láncolásnál sikeresnek bizonyult, a nyílt módszer esetében azonban nem járt eredménnyel. Ha végiggondoljuk, ebben nincs semmi meglepő, a nyílt módszer használatakor nem lehet a szinonimlánchoz nem tartozó rekordokat „átugrani”, így a kétszeres betöltésnél „amit nyerünk a réven, elveszítjük a vámon”.

Végezetül helyezzük el ugyanezeket a rekordokat többszörös hashing használatával is. A második hashing algoritmus is legyen maradék módszer, azonban ez esetben a modulust válasszuk 7-nek (3. táblázat).

Kulcs	Számított cím I.	Számított cím II.	Tényleges cím	Lépés- szám
46	2	-	2	1
24	2	3	3	2
29	7	-	7	1
36	3	1	1	2
30	8	-	8	1
17	6	-	6	1
75	9	-	9	1
74	8	4	4	2
14	3	0	0	2
19	8	3	5	3, ill. 4
Összesen				16, ill. 17

3. táblázat *Példa kétszeres hashing-ra*

A 19 kulcsú rekord a második hashing alkalmazása után is szinonim lesz. Elhelyezését megoldhatjuk a belső láncolás, vagy a nyílt módszer használatával (ezért tartalmaz a táblázat itt két értéket).

## 3. FEJEZET

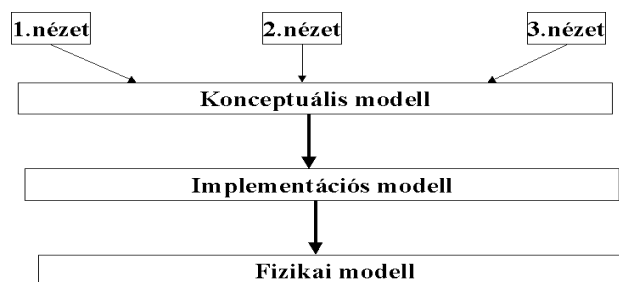
# Az adatbázis szemléletű információ-feldolgozás

Egyfelől a tárolókapacitások robbanásszerű növekedése, másfelől a felhasználói körnek a rendszertámogatások által lehetővé tett bővülése indokolta az új információ-feldolgozási forma megjelenését az 1960-as években. E szemlélet lényege, hogy adatorientált, azaz a modellálni kívánt mini világ rendelkezésére álló összes adatot és a közöttük fennálló összes kapcsolatot egy integrált adatbázisba gyűjtjük, és valamennyi felhasználó valamennyi kérdéséhez ezt az adatbázist (vagy ennek egy részét) használhatja.

## Adatbázisok tervezése

### A modellek és kapcsolataik

Szemben a folyamat-szemléletű információ-feldolgozással, az adatbázisokat használják *ad hoc* jellegű kérdések megválaszolására. Rögzíteni kell tehát annak a *mini világnak* határait, melyen belül e kérdések feltehetőek. Azt a konzisztens világot, melyet a teljes adatbázis-kezelő rendszer leír, *konceptuális modellnek* (magasszintű modell) nevezik. Ezt a modellt lehet leképezni az *implementációs modellre* (reprezentációs modell), melyben a használt konkrét adatbázis-kezelő rendszernek megfelelő struktúrában kerül sor a rekordok, mezők, stb. leírására. Az implementációs modellt lehet leképezni a *fizikai modellre* (alacsony szintű modell), mely már az alkalmazott számítógép tulajdonságainak is megfelel.



11. ábra Egy adatbázis struktúrája

A felhasználók nagy részének nincs szüksége a teljes adatbázisra, sokszor egy-egy felhasználónak nincs is joga az adatbázisban mindenhez hozzáférni. Egy-egy *felhasználói nézet* tehát csak az adatbázis egy részét láthatja (11. ábra). Ezen belül is szabályozandó, hogy a látott adatokkal ki mit tehet (más egy adatot megnézni, vagy például megváltoztatni).

## Az adatbázis felügyelő és alapvető feladatai

Szükséges tehát, hogy legyen valaki, aki a felhasználókénál nagyobb hatáskörrel rendelkezve, ezeket a jogokat kiossza és a munkát felügyelje. Ez a személy, vagy csoport az *adatbázis felügyelő* (adatbázis adminisztrátor). Az ő joga és kötelessége az adatbázist megtervezni, elkészíteni a fent említett modelleket leíró *sémát*, valamint az egyes nézetek által látott adatbázis-rész logikai struktúráit leíró *alsémákat*. Ugyancsak az adatbázis felügyelő feladata az adatbázis működtetése során a felhasználókkal való kapcsolattartás, az adatbázis logikai és fizikai struktúrájának a felhasználói igényeknek megfelelő módosítása, valamint az adatbázis tartalmának rendszeres mentése.

A séma és alséma leírások (másképpen a meta adatok megadása) az úgynevezett *DDL* (Data Definition Language – adatleíró nyelv) nyelven készülnek, ezt a nyelvet többnyire csak az adatbázis felügyelet használhatja.

A tervezéskor az adatbázis felügyelőnek figyelembe kell vennie az adott információ-feldolgozás előzményeit, és biztosítani kell a későbbi változtatások lehetőségét. Ezt segíti elő a *logikai és fizikai adatfüggetlenség*. A logikai adatfüggetlenség azt jelenti, hogy az adatok logikai struktúrájában (konceptuális, implementációs modell) történő változtatás ne érintse az ezen változtatást nem igénylő felhasználók munkáját. A fizikai adatfüggetlenség pedig azt jelenti, hogy a fizikai modell változtatása ne kényszerítse ki az implementációs modell változtatását. A két követelmény együttes teljesítésekor beszélhetünk *adatfüggetlenségről*.

Ugyancsak ügyelni kell a tervezéskor a biztonság (géphiba, természeti csapás, stb. elleni védelem), a titkosság (illetéktelen hozzáférések megakadályozása), továbbá a pontosság (lehetőleg ne kerüljön az adatbázisba hibás adat) követelményeire, és természetesen nem hagyható figyelmen kívül a válaszidő és a költségek kérdése sem.

Az adatbázis felügyelőnek nem csak tervezéskor vannak feladatai, végig felügyelnie kell az adatbázis működését is, eközben egyrészt rendszeresen mentenie kell annak tartalmát (és természetesen szükség esetén ezekből helyreállítani a megsérült adatbázist), másrészt kapcsolatot kell tartania a felhasználókkal, meghallgatva, s lehetőség szerint teljesítve azoknak az adatbázis szerkezetét illető kéréseit.

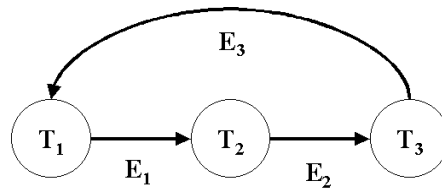
## Konkurrens folyamatok

Az egyes rekordok feldolgozásának lépései:

1. a rekord megkeresése,
2. a rekord beolvasása a felhasználó által elérhető munkaterületre,
3. az adatok feldolgozása, megváltoztatása,
4. a rekord visszaírása az adatbázisba.

Ha az adatbázist egyidejűleg többen is használják, nem zárható ki az sem, hogy egyszerre többen akarják ugyanazt a rekordot módosítani. Ekkor azonban bekövetkezhet az a – meg nem engedhető – helyzet, hogy a rekordot másodikként kiolvasó és visszaíró felhasználó tevékenysége miatt az első felhasználó által elvégzett változtatások hatása elvész. Ez ellen

úgy védekezhethetünk, hogy – változtató tranzakciók esetén – megtiltjuk, hogy az érintett adatokhoz a tranzakció időtartama alatt más is hozzáférhessen (ezt nevezzük az adatok *lezárásának*). Sajnos, ennek a módszernek is vannak nem kívánt „mellékhatásai”. A lezárható adatok meg nem osztható erőforrások egy olyan rendszerben, melyben több tranzakció vettélkedik az erőforrásokért. Ha két tranzakciónak ellenkező sorrendben van szüksége ugyanarra az adatra, mégpedig kizárólagos használati joggal, előállhat az úgynevezett *patthelyzet* (kölsönös kizárás, deadlock – holtpont), melyben a tranzakciók, illetve az azokat kiváltó folyamatok (processzek) kölcsönösen egymásra várnak. A patthelyzet megelőzéséhez egy-egy tranzakciót csak akkor lenne szabad elindítani, ha az összes általa igényelt erőforrást le tudjuk a tranzakció számára foglalni. Túl azon, hogy ez mindenképpen erősen lelassítaná a rendszer működését, azzal is számolnunk kell, hogy egy adott tranzakció pontos erőforrás-igénye sokszor csak futás közben derül ki. Be kell érünk tehát azzal, hogy felfedezzük, és feloldjuk a patthelyzeteket. A felfedezéshez segítséget nyújthatnak az úgynevezett *várakozási gráfok* (12. ábra). Jelölje  $T_i$  az  $i$ -edik tranzakciót, s legyenek ezek a gráf csúcspontjai. Legyen  $E_k$  a  $k$ -adik erőforrás, és mondjuk ki a következő szabályt: akkor vezet  $T_i$ -ből  $T_j$ -be az  $E_k$  él, ha az  $E_k$  erőforrásra várakozó sorban  $T_j$  megelőzi  $T_i$ -t. A patthelyzetet kereső algoritmusnak azt kell vizsgálnia, hogy van-e a várakozási gráfban kör.



12. ábra Várakozási gráf

A patthelyzet csak az abban részt vevő tranzakciók valamelyikének leállításával oldható fel. A leállított tranzakciót később újra kell futtatni. Ha azonban a leállítással egyidejűleg nem gondoskodunk a leállított tranzakció által eddig végzett változtatások visszavonásáról (*visszagörgetés*, rollback), előfordulhat, hogy a rekordok egy részén ugyanazt a változtatást többször is elvégezzük (pl. egy összeget megadott százalékkal megnövelünk). A visszagörgetés megoldására többféle technika is ismeretes. A legelterjedtebb megoldás az úgynevezett *journal* állományok használata, amikor a változtatásokat nem az eredeti adatokon, hanem azok egy kópiáján végzi a rendszer, s a visszamásolásra csak a teljes tranzakció befejezése után kerül sor. Egy másik sokat használt módszer az úgynevezett *különbségi* állományok használata, amikor az eredeti adatok egyáltalán nem változnak meg, csak a különbségi állományba kerül bejegyzésre a változtatás. (Hasonlóan egy könyv hibajavító jegyzékéhez.) E módszer egyszerű lekérdezéseknél feleslegessé teszi a patthelyzettel fenyegető lezárásokat, ezzel szemben egy-egy rekord kiolvasásánál meg kell vizsgálni a különbségi állományt is, hogy nem tartozik-e a rekordhoz bejegyzés. Ha a különbségi állomány túl nagy, újra kell szervezni az adatbázist, hogy a keresés ne lassuljon le túlságosan.

## A felhasználó eszközei

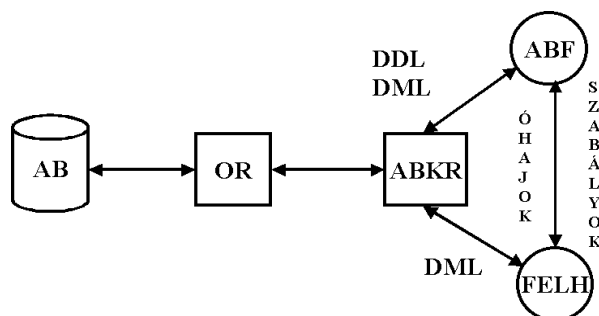
A felhasználók a lekérdezéseket, adatváltoztatásokat a **DML** (Data Manipulation Language – adatmanipuláló nyelv) nyelven eszközölhetik. Későbbi rendszerek további nyelvtípusokat is használnak (például a DCL, melyről a következő fejezetben lesz szó.)

E nyelveket több szempontból osztályozhatjuk. Egyfelől aszerint, hogy korábban ismert, vagy újonnan készített nyelvet használunk. Egy gazdanyelv (befogadó nyelv, beágyazó nyelv) azoknak a programozóknak áll rendelkezésére, akik az általuk korábban jól megismert magasszintű nyelven kívánnak dolgozni, ezt a nyelvet egészíti ki a rendszer az adatbázisok kezelésére alkalmas utasításokkal, eljárásokkal. Egy önálló nyelv teljesen kerek önálló rendszer, ami lehet programozási nyelv éppúgy, mint egy paraméterezéssel kezelhető felhasználói felület (erre példa a légitársaságok helyfoglalási rendszere).

Másik lehetőség az osztályozásra annak vizsgálata, hogy az alkalmazott nyelv *procedurális* (rekordokat feldolgozó), vagy *deklaratív* (halmazokat feldolgozó) jellegű. Az első adatbázisok egyértelműen csak a procedurális feldolgozást támogatták.

### Az operációs rendszer által nyújtott támogatás

Az adatbázis-kezelő rendszerek is kihasználják az operációs rendszerek adatkezelési támogatását, építenek az operációs rendszerek data management rutinjaira, így rendszerint a fizikai input/output lebonyolítását az operációs rendszerre bízzák (12. ábra).



13. ábra Az adatbázis, az operációs rendszer, az adatbázis-kezelő rendszer, a felhasználó és az adatbázis felügyelő kapcsolata

Vizsgáljuk meg hogy, egy egyszerű lekérdezés esetén milyen mozzanatokat kell az adatbázis-kezelő rendszernek, ill. az operációs rendszernek elvégeznie (13. ábra).

1. A felhasználó az adatbázis-kezelő rendszerhez fordul kérésével.
2. Az adatbázis-kezelő rendszer ellenőrzi a sémában a kérés jogosságát.
3. Az adatbázis-kezelő rendszer ellenőrzi a megfelelő alsémában a kérés jogosságát.
4. Az adatbázis-kezelő rendszer kéri az operációs rendszertől a fizikai I/O elvégzését.
5. Az operációs rendszer megkeresi a kért rekordot.
6. Az operációs rendszer beolvassa a kért rekordot a rendszerpufferbe.
7. Az operációs rendszer értesíti a történetekről az adatbázis-kezelő rendszert.
8. A kért rekord a rendszerpufferből átkerül a felhasználó munkaterületére.
9. Az adatbázis-kezelő rendszer értesíti a felhasználót, hogy a rekord a rendelkezésére áll.

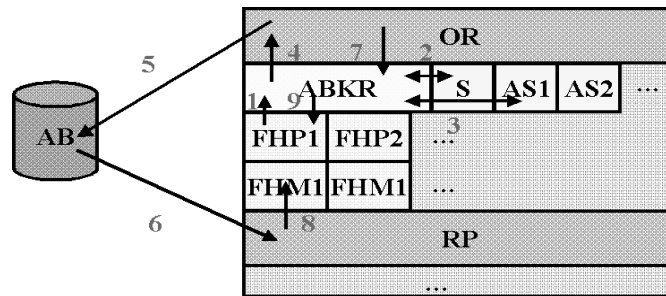
Világos, hogy egy olyan tranzakció, mely változtató, s így lezárásokat követel meg, a fenténél sokkal több és bonyolultabb lépést igényel.

## Az alkalmazott rendszerek sajátosságai

Az adatbázis kezelő rendszerek múltjához ez idő szerint két modell tartozik: a hierarchikus, és a hálós. Mindkét modell közös tulajdonsága, hogy *egyedeket* (entity) és *kapcsolatokat* (relationship) ábrázol (*ER modell*).

### A HIERARCHIKUS MODELL

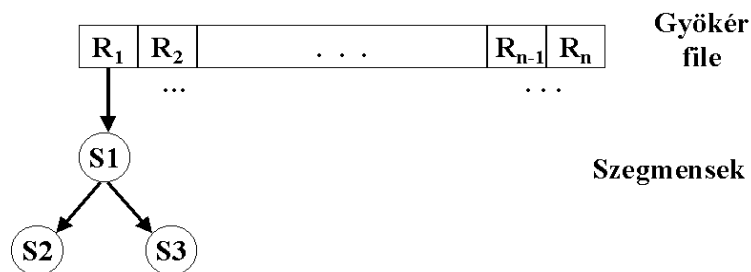
A legrégebbi modell a hierarchikus, mely ma azonban már nem használatos. Lényege, hogy az adatokat fákban tároljuk, ahol a fák egy-egy szögpontja a *szegmens* adatokat és a további szegmensekre utaló mutatókat tartalmaz. A fák gyökérelemei hagyományos állományokba vannak szervezve. Az egyes nézetek a számukra *érzékeny szegmenseket* látják. A csoport jellegzetes képviselője az IBM által készített IMS (14. ábra).



14. ábra Egy egyszerű lekérdezés folyamata

A modell a hálós struktúrájú feladatok leírására csak korlátozottan alkalmas, ilyen esetekben a lekérdezés hatékonysága erősen függ az adatbázis struktúrájától.

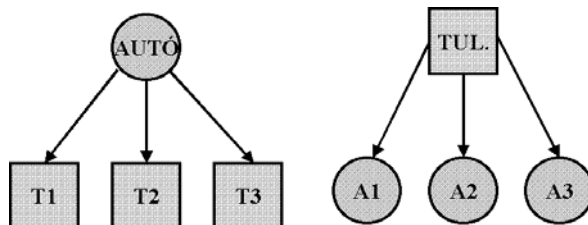
A 15. ábrán látható esetben például, ha autók adatainak ismeretében keressük azok tulajdonosainak adatait, az első struktúra hatékony megoldást ad, a második azonban csak a lineáris keresés lehetőségét nyújtja. Ha viszont a tulajdonosok adatai ismertek és az autókét keressük, a helyzet éppen fordított. A hatékonyan megválaszolható kérdések köre tehát a struktúra függvénye, s ez nem egyezik azokkal az elvekkel, melyeket az adatbázis szemlélet meghatározásánál kitűztünk.



15. ábra Hierarchikus állomány szerkezete

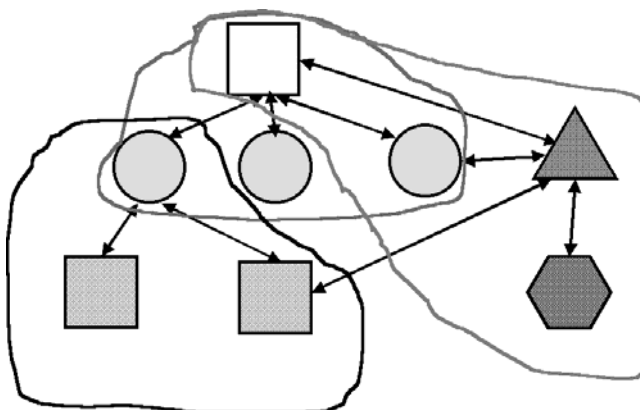
### A HÁLÓS MODELL

A hierarchikus modell ki nem elégítő volta miatt hamarosan szükségessé vált az új modell kidolgozása. Sok individuális próbálkozás után a CODASYL bizottság által létrehozott DBTG (Data Base Task Group: adatbázis munkacsoport) jelentései (1969-1971) hozták létre azt a terminológiai és metodológiai közös alapot, amin a hierarchikus rendszereket fejleszteni lehetett. Mintegy két évtizedig ez a jelentés volt a nagy adatbázis-kezelő rendszerek tervezésének alfája és omegája.



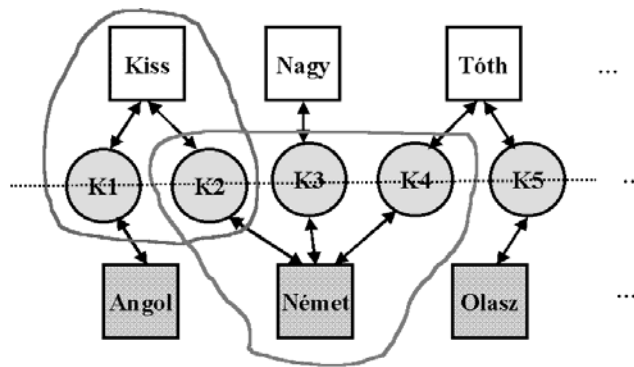
16. ábra Egy autókból és tulajdonosaikból álló állomány lehetséges kétféle felépítése

A DBTG jelentés terminológiai javaslataiból már használtunk néhányat: ilyen pl. a séma, az alséma, a DDL, a DML, stb. A legfontosabb újítás a *set* fogalmának bevezetése volt. A set egy rekordokból álló kétszintű fa, melynek gyökéreleme a *tulajdonos*, levelei a *tagok*. Természetesen egy-egy rekord lehet az egyik set-ben tulajdonos, egy másikban tag, s ugyanígy lehetséges, hogy egy rekord több set-nek is tagja legyen. Ha megengedjük olyan rekordok használatát is, melyek csak a kapcsolat leírására szolgálnak, a set-ek segítségével a legbonyolultabb hálós kapcsolatok is leírhatók (16. ábra).



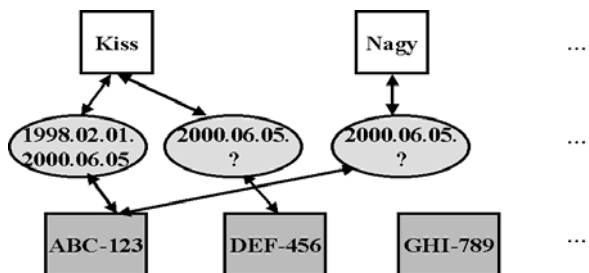
17. ábra Példa set-ekre

Egy korábban is ismert megoldás, az úgynevezett kapcsolórekordok használata, tulajdonképpen a set-ek megvalósítását adja. Képzeltük el, például, hogy embereket és nyelveket tartunk nyilván, valamint ezek közötti kapcsolatokat: pl. hogy melyik ember milyen nyelvet beszél. Ezt megtehetjük úgy, hogy minden kapcsolathoz egy-egy kapcsolórekordot rendelünk. Ilyen módon a hálós struktúra hierarchikus struktúrákra, set-ekre bontható (17. ábra).



18. ábra Set-ek a kapcsolórekordos megoldásnál

A kapcsolórekordok természetesen tartalmazhatnak adatokat is. Egy személyeket és autókat nyilvántartó rendszerben például a tulajdonlást leíró kapcsolórekordok tartalmazhatják a kapcsolat kezdetét és végét (18. ábra).



19. ábra Példa adatot is tartalmazó kapcsolórekordokra

A másik fontos új fogalom a *terület*, mely együtt kezelendő rekordok egy halmaza.

A hálós adatbázis-kezelő rendszerek egyik fontos reprezentánsa a Magyarországon is elterjedt IDMS (Integrated Data Management System), melyben a fentiekén kívül rendelkezni lehet a set-ek tagjainak rendezéséről, a rekordok valamilyen hashing algoritmus szerinti elhelyezéséről, inextáblák készítéséről, stb. is. E tulajdonságok a hálós adatbázis-kezelő rendszereknek nem kötelező, de gyakori tartozékai.

DML-ként a DBTG jelentés a COBOL nyelv gazda nyelvként való használatát javasolta (akkoriban szinte kizárólag kötegelt (batch) feldolgozást használtak, s az elterjedt magas-szintű nyelvek közül csak a COBOL volt alkalmas adatok manipulálására). A COBOL-t sok helyen hamar felváltotta a PL1, majd az interaktív feldolgozás térnyerésével más, ezt kihasználó felhasználói felületek is szerepet kaptak.

## 4. FEJEZET

# Az információ-feldolgozás jelene és jövője

## Relációs adatmodell és használata

Napjainkban az adatbázis-kezelő rendszerek gyakorlatilag kivétel nélkül a *relációs modell* képviselői. E modell ötlete Codd 1970-es cikkéből [Co70] származik, így lényegében egyidős a DBTG jelentéssel. Mégis csaknem húsz évet kellett várni arra, hogy ez a modell átvegye a vezető szerepet. Az ok: a számítógépek kapacitásának és sebességének kellett növekednie ahhoz, hogy a relációs modellt hatékonyan implementálni lehessen. Mindenesetre napjainkban szinte kizárólag e modell alapján készülnek adatbázis-kezelő rendszerek.

A *reláció* ebben az értelemben tulajdonképpen egy *táblázat*, a gyakorlati szakirodalom így is említi. A táblázat oszlopai a *tulajdonságok*, a sorai az *objektumok*. A hagyományos terminológiának megfelelően azonban gyakran nevezik a sorokat rekordnak, az egyes oszlopokhoz tartozó értékeket pedig mezőnek.

A táblázattal kapcsolatos alapvető feltételezések, hogy abban ne legyenek teljesen megegyező tartalmú sorok vagy oszlopok, továbbá a sorok és az oszlopok sorrendje ne hordozzon információt. Azt a mezőt, vagy mezőkészletet, mely a sort (a sor többi elemét) egyértelműen meghatározza, *kulcsnak* nevezzük.

Karbantartási anomáliák és információvesztések elkerülése végett a relációkat célszerű *normalizálni*. A normalizálás legfontosabb lépései az alábbi *normál formákon* át vezetnek.

- 1NF  
A relációt elsőrendben normalizáltnak nevezzük, ha annak mezői elemi értéket (nem relációt) tartalmaznak.
- 2NF  
A relációt másodrendben normalizáltnak nevezzük, ha elsőrendben normalizált, és amennyiben valamelyik mezőjének azonosításához egy összetett kulcs szükséges, nincs olyan mező, melynek azonosításához elegendő ennek egy része.
- 3NF  
A relációt harmadrendben normalizáltnak nevezzük, ha másodrendben normalizált, és a nem kulcs-jellemzői nem függenek egymástól.
- BCNF  
A relációt BOYCE-CODD értelemben normalizáltnak nevezzük, ha a fentiekén kívül

teljesül az is, hogy egyetlen nem kulcs-jellemző sem határozza meg egy összetett kulcs valamelyik összetevőjét (nincs kulcstörés).

Bizonyítható, hogy minden reláció felbontható ilyen, normalizált relációkra, ezt a műveletet nevezzük *dekompozíciónak*.

A relációk legfontosabb tulajdonsága, hogy azok egzakt matematikai eszközökkel kezelhetők. Ilyen eszközrendszer a *relációs algebra* és a *relációs kalkulus*.

## A relációs algebra

A relációs algebra a relációk feldolgozásának legfontosabb eszköze. A II. részt teljes egészében a relációs algebra tárgyalásával foglalkozik, ezért itt csak a legfontosabb tulajdonságait vázoljuk.

### A RELÁCIÓS ALGEBRA ALAPMŰVELETEI

- *Egyesítés:*  $R \cup S$   
Az  $R$  és az  $S$  reláció egyesítésén azt a relációt értjük, melyben szerepelnek mindazon sorok, melyek akár az  $R$ , akár az  $S$  relációban szerepelnek. Az  $R$  és az  $S$  reláció (és természetesen az eredmény reláció) oszlopszámának meg kell egyeznie.
- *Különbség:*  $R \setminus S$   
Az  $R$  és az  $S$  reláció különbségén azt a relációt értjük, melyben csak azok a sorok szerepelnek, melyek benne vannak az  $R$ , de nincsenek benne az  $S$  relációban.
- *Descartes-szorzat:*  $R \times S$   
A  $r$  oszlopszámú  $R$  és a  $s$  oszlopszámú  $S$  reláció Descartes-szorzatán azt a relációt értjük, melynek  $r + s$  oszlopszámú soraiban minden  $R$ -beli sor minden  $S$ -beli sor előtt előfordul.
- *Projekció:*  $\pi_{i_1, i_2, \dots, i_n}(R)$   
Az  $R$  reláció  $i_1, i_2, \dots, i_n$  oszlopaire való projekcióján azt a relációt értjük, mely az  $R$  relációból úgy keletkezik, hogy elhagyjuk az  $i_1, i_2, \dots, i_n$ -től különböző oszlopokat.
- *Szelekció:*  $\sigma_T(R)$   
Az  $R$  relációnak a  $T$  feltétel melletti szelekcióján azt a relációt értjük, mely az  $R$  relációból úgy származtatható, hogy abból csak a  $T$  feltételnek eleget tévő sorokat hagyjuk meg.  
A  $T$  feltétel az  
 $[i] \bullet [j]$  (a sor  $i$ -edik és  $j$ -edik eleme között fennáll  $\bullet$ )  
és az  
 $[i] \bullet c$  (a sor  $i$ -edik eleme és a  $c$  konstans között fennáll  $\bullet$ )  
elemi kifejezések logikai műveletekkel ( $\wedge, \vee, \neg$ ) való összekötéséből állhat. A  $\bullet$  tetszőleges összehasonlító operátort ( $<, >, \leq, \geq, =, \neq$ ) jelenthet.

### A KÖVETKEZMÉNY MŰVELETEK

A soron következő műveletek voltaképp nélkülözhetők, de a gyakorlatban jól lehet őket használni.

- *Metszet:*

$$R \cap S = R \setminus (R \setminus S)$$

- *Hányados:*

$$R \div S$$

Az  $R$  és az  $S$  reláció hányadosán azt a relációt értjük, melyre igaz, hogy  $(R \div S) \times S$  összes sora  $R$ -beli sor. (Feltesszük, hogy  $r > s$  és  $s \neq 0$ .)

#### Állítás

A hányados valóban következmény művelet.

#### Bizonyítás

Legyen ugyanis

$$T = \Pi_{1,2, \dots, r-s}(R), \text{ továbbá}$$

$$V = \Pi_{1,2, \dots, r-s}((T \times S) - R).$$

Ekkor nyilvánvaló, hogy

$$R \div S = T - V.$$

- *Általános összekapcsolás* (Theta kapcsolat, Join):

$$R \bowtie_T S = \sigma_T(R \times S).$$

- *Természetes összekapcsolás* (natural join):

$$R \bowtie S.$$

Hasonló az általános összekapcsoláshoz, de itt a szelekció feltétele az, hogy az azonos nevű oszlopokban azonos érték szerepeljen. Természetesen a szelekció után projekcióra is szükség lesz, mivel az azonos tartalmú oszlopok egyikét meg kell szüntetni.

### Példa a relációs algebra alkalmazására

Tekintsük az alábbi adatbázist:

Legyen három relációnk.

- Autók: jele  $A$   
Mezői: rendszám, gyártmány, típus, szín,...
- Emberek: jele  $E$   
Mezői: számszám, név, foglalkozás, cím,...
- Kapcsolat: jele  $K$   
Mezői: forgrsz, számszám.

#### Feladat:

Keressük ki a sárga opelek tulajdonosainak foglalkozását.

*Megoldás:*

- 1.) Kiválasztjuk az  $A$  táblázatból a sárga opelekre vonatkozó sorokat (szelekció):  
 $R1 = \sigma_{\text{szín}=\text{sárga} \wedge \text{típus}=\text{opel}}(A)$
- 2.) Eldobjuk a „rendszám” mezőn kívüli oszlopokat (nem kötelező, de célszerű a projekció, így nem hordozunk célunk szempontjából felesleges adatokat):  
 $R2 = \pi_{\text{rendszám}}(R1)$
- 3.) Összekapcsoljuk eddig nyert eredményünket a  $K$  táblázattal a „rendszám” és a  $K$ -beli „forgrsz” mezők egyezése alapján (általános összekapcsolás):  
 $R3 = R1 \bowtie_T K$ , ahol  $T = (\text{rendszám} = \text{forgrsz})$
- 4.) Eldobjuk a „számszám” mezőn kívüli oszlopokat (projekció):  
 $R4 = \pi_{\text{számszám}}(R3)$
- 5.) Összekapcsoljuk eddig nyert eredményünket az  $E$  táblázattal (a „számszám” mező mindkét táblázatban előfordul: természetes összekapcsolás):  
 $R5 = R3 \bowtie E$

6.) Eldobjuk a „foglalkozás” mezőn kívüli oszlopokat (nincs rájuk szükség, projekció):

$$R6 = \pi_{\text{foglalkozás}}(R5)$$

Természetesen mindezt egy formulába is foglalhatjuk:

$$\pi_{\text{foglalkozás}}(\pi_{\text{szemszám}}(\pi_{\text{rendszám}}(\sigma_{\text{szín}=\text{sárga} \wedge \text{típus}=\text{opel}}(A)) \bowtie_T K) \bowtie E),$$

ahol  $T = (\text{rendszám} = \text{forgrsz})$ .

## A relációs kalkulus

A relációk kezelésére használatos másik matematikai módszer a relációs kalkulus. Ennek során a

$$\{t \mid \Psi(t)\}$$

kifejezést vizsgáljuk, melynek jelentése: azokat a  $t$  sorokat tekintjük, melyek kielégítik a  $\Psi(t)$  formulát. A formula következő, úgynevezett atomokból épülhet fel:

- $R(s)$  azt jelenti, hogy az  $s$  sor része az  $R$  relációnak,
- $u[i] \bullet v[j]$  azt jelenti, hogy az  $u$  sor  $i$ -edik és a  $v$  sor  $j$ -edik eleme között fennáll a  $\bullet$  viszony,
- $u[i] \bullet c$  azt jelenti, hogy az  $u$  sor  $i$ -edik eleme és a  $c$  konstans között fennáll a  $\bullet$  viszony.

Az atomok önmagukban is formulák, de azokat a logikai műveletek jeleivel ( $\wedge$ ,  $\vee$ ,  $\neg$ ) összeköthetjük, ezenkívül a  $\exists$  jel (van olyan, hogy), a  $\forall$  jel (minden olyan) és a zárójelek is használhatóak.

A relációs kalkulus eszközeivel minden felírható, amit a relációs algebrával kifejezhetünk. Ez egyszerűen belátható az alpműveleteknek megfelelő kifejezések felírásával.

- *Egyesítés:*

$$\{t \mid R(t) \vee S(t)\},$$

- *Különbség:*

$$\{t \mid R(t) \wedge \neg S(t)\},$$

- *Direkt szorzat:*

$$\{t^{(r+s)} \mid (\exists u^{(r)})(\exists v^{(s)})(R(u) \wedge S(v) \wedge t[1]=u[1] \wedge \dots \wedge t[r]=u[r] \wedge t[r+1]=v[1] \wedge \dots \wedge t[r+s]=v[s]\},$$

- *Projekció:*

$$\{t^{(k)} \mid (\exists u)(R(u) \wedge t[1]=u[i_1] \wedge \dots \wedge t[k]=u[i_k]\},$$

- *Szelekció:*

$$\{t \mid R(t) \wedge F\}.$$

Ugyanakkor készíthetünk olyan kifejezéseket is, melyek a relációs algebrában nem fejezhetők ki (például  $\{t \mid \neg R(t)\}$ ). Könnyen belátható, hogy ennek a kifejezésnek nem sok értelme van. Az effajta kifejezések kizárásával létrehozhatjuk az úgynevezett *biztos kifejezések* körét. Ehhez szükség van a *domain* függvény bevezetésére. Egy  $\Psi$  formulához rendelt  $\text{domain}(\Psi)$  azoknak az elemeknek a készletét jelenti, melyek a  $\Psi$  formulában közvetlen,

vagy közvetett módon említésre kerülnek. Ahhoz, hogy egy kifejezés biztos legyen, az alábbi feltételeknek kell eleget tennie:

- Ha egy  $t$  sor kielégíti a  $\Psi$  formulát, minden komponense tagja  $\text{domain}(\Psi)$ -nek.
- $\Psi$  minden  $(\exists u)(\omega(u))$  formájú részkiefezésére, ahol  $u$  kielégíti  $\omega(u)$ -t,  $u$  minden komponense tagja  $\text{domain}(\omega)$ -nak.

A biztos kifejezések készletéről már belátható, hogy ekvivalens a relációs algebra kifejezés-készletével. Mindezek részletesebb kifejtése megtalálható pl. az [U182] könyvben.

## Lekérdezés relációs rendszerekben

A relációs rendszerekben a relációs algebra ill. a relációs kalkulus lehetővé teszi, hogy ne (csak) a hagyományos procedurális módon, hanem deklaratív úton (is) dolgozzuk fel az információkat. Az első lekérdező rendszerek pusztán a relációs algebra vagy kalkulus számítógépes megvalósítását tűzték ki célul. (Ilyen például az IBM által kifejlesztett *ISBL*.) Ez azonban nem elégítette ki a felhasználói felületekkel kapcsolatos igényeket.

### QBE

Viszonylag korai nyelv az ugyancsak IBM termék *QBE* (Query by Example), mely az úgynevezett *táblavázak* (skeleton) részbeni kitöltése után a táblázat fennmaradó részének kitöltésével ad választ a felhasználó kérdéseire. Ez a technika a legújabb rendszerekben is megtalálható. A QBE változó vektorok alkalmazásával tette lehetővé az egyes relációk összekapcsolását.

A relációs algebraánál ismertetett példa megoldása az alábbi módon történhet (1. ábra):

AUTÓK			
rends.	gym.	szín	...
<u>X</u>	OPEL	SÁRGA	

KAPCSOLAT		
frsz.	sz.sz.	...
<u>X</u>	<u>Y</u>	

EMBEREK			
sz.sz.	név	fogl.	...
<u>Y</u>	Kiss	tanár	
...	...	...	

1. ábra A QBE használata

### SQL

E könyv következő részeiben részletesen foglalkozunk az *SQL* (Structured Query Language) nyelvvel, melyet szinte minden korszerű adatbázis-kezelő rendszer „ért”.

Az SQL - némiképp eltérően a szokásoktól - négy nyelvet, illetve résznyelvet tartalmaz.

- DDL (Data Definition Language)  
Ezen lehet táblákat, nézettáblákat és indexeket létrehozni (CREATE), táblákat módosítani (ALTER), valamint táblákat, nézettáblákat és indexeket megszüntetni (DROP).

- DML (Data Manipulation Language)  
E résznyelv tartalmazza a rekordok beszúrásához (INSERT), módosításához (UPDATE) és törléséhez (DELETE) szükséges utasításokat.
- DQL (Data Query Language)  
Ez a résznyelv egyetlen utasításból áll (SELECT), mely azonban a legösszetettebb kérdések megfogalmazására is alkalmas.
- DCL (Data Control Language)  
E nyelv feladata kettős, egyrészt ide tartozik a jogosítványok kiosztása és visszavonása (GRANT, REVOKE), másrészt e nyelven lehet a tranzakciókat véglegesíteni, illetve visszagörgetni (COMMIT, ROLLBACK).

### A NEGYEDIK GENERÁCIÓS (4GL) NYELVEK

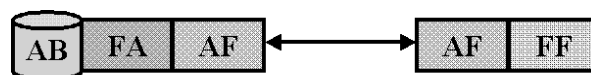
A modern adatbázis-kezelő rendszerek még az SQL-beli „programozást” is meg akarják takarítani a felhasználónak. Különbő paramétereizhető form, report, menu stb. generátorok teszik ezt lehetővé.

## Kliens-szerver architektúrák

A helyi hálózatok elterjedésével logikusan vált egyre általánosabbá az a megoldás, hogy a szerver gépen elhelyezett adatbázist a hálózatba kapcsolt gépekről használták. Ha azonban minden feldolgozást a kliens gépek végeznek, igen megnőhet a felesleges adatátvitel. (Gondoljunk csak egy egyszerű válogatásra, ha azt a helyi gép végzi, minden rekordot be kell olvasnia a hálózaton keresztül.) Logikusnak látszott tehát az ötlet, hogy a feladatok egy részét a szerver gépen kellene elvégezni. Az adatbázist feldolgozó programokat sokféleképpen lehet felosztani, az egyik – elterjedt – felosztás szerint az alábbi három részből beszélhetünk:

- *felhasználói felület* (user interface),  
mely szükségképpen a felhasználó (kliens) gépén található,
- *alkalmazás feldolgozása* (application processor, business logic),  
mely lehet a kliens és a szerver gépen is, de meg is osztható,
- *fizikai adatelérés* (data processing),  
melynek helye az adatbázissal együtt a szerver gép.

Hagyományos *kliens-szerver architektúráról* akkor beszélünk, ha a fentiek meg vannak osztva a kliens és a szerver között (2. ábra).

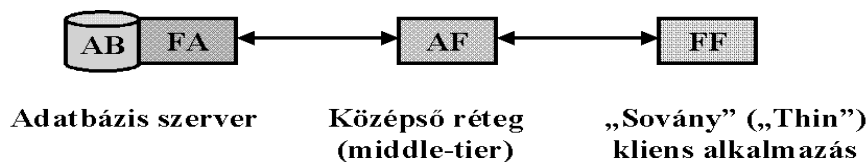


Adatbázis szerver

Kliens alkalmazás

2. ábra Kliens-szerver architektúra

Létezik olyan megoldás is, mely az említett három részt három gépre helyezi: a „sovány” („thin”) kliens gépen csak felhasználói felület van, a felhasználói alkalmazás a középső gépen fut, s a szerverre csak a fizikai adatelérés marad (3. ábra).



3. ábra Három rétegű kliens-szerver architektúra

Természetesen a gyakorlatban egy-egy adatbázis-kezelő rendszerben több kliens és igen gyakran több szerver is dolgozik.

## Osztott adatbázisok kezelése

A nagy hálózatok és az ezek következtében sok távoli lekérdezés a 80-as évekre erősen megnövelte a kommunikációs költségek részarányát az adatbázis-kezelés költségein belül. Ez a tény sugallta az ötletet: próbáljuk meg az adatokat a felhasználás közelében elhelyezni. Az eredmény: egy *fizikailag megosztott, de logikailag egységes adatbázis*, ezt nevezzük *osztott adatbázisnak*.

Az osztott jelleg a felhasználó számára nem látható (transzparens), ugyanakkor több jelentős előnnyel és hátránnyal jár.

*Előnyök:*

- A kommunikációs költségek már említett csökkenése.
- Mindenki a számára ismerős adatokat gondolja.
- Egy-egy csomópont kiesése esetén a többi adatai továbbra is elérhetőek.
- Lehetséges a moduláris tervezés, a rugalmas konfigurálás.
- Hosszabb idő alatt a rendszer gépei akár ki is cserélhetők.

*Hátrányok:*

- A rendszer bonyolultabb és sebezhetőbb lesz.
- Nem könnyű minden csomópontra egyformán jó személyzetet találni, másrészt, ha találunk, fenyeget a szuboptimalizáció veszélye.
- Mindig valamennyi gépnek működni kell.
- Többféle hardvert és szoftvert kell a rendszernek kezelnie és „összehoznia”.
- Bonyolult a jogosultságok ellenőrzése. (A jogosultságokat leíró táblázatokat hol tároljuk: egy csomópontban, vagy mindenütt?)

Külön problémát jelent, ha feladjuk a redundancia-mentesség elvét. Erre okot adhat az is, ha nem eldönthető egy-egy adatról, hogy hol használják legtöbbször, de biztonsági okokból is dönthetünk egy-egy adat többszörözése mellett. Ilyen esetekben biztosítanunk kell, hogy az egyes példányok tartalma azonos legyen, azaz a rendszer *konzisztens* maradjon.

Az első osztott rendszerek megvalósításához nagy segítséget nyújtott az úgynevezett *backend gépek* felhasználása. A „hőskorban” egy-egy ilyen rendszer megtervezése és elkészítése hatalmas erőfeszítéseket követelt, gondoljunk csak arra, hogy 15-20 éve még egyáltalán nem voltak a maihoz hasonlóan kidolgozva a számítógépek kapcsolatainak szabályai, hogy az operációs rendszereket egy-egy géptípushoz fejlesztették ki, stb.. Ilyen körülmények között egy osztott rendszerhez le kellett küzdeni mind a hardver, mind a szoftver inkompatibilitásából adódó úgynevezett transzlációs problémákat. A mai adatbázis-kezelő rendszerek ugyanakkor készen adják a megosztáshoz szükséges eszközöket.

Az adatok elhelyezésének megtervezése az adatbázis felügyelő feladata. (Mivel az osztott rendszereknél meg kell különböztetnünk központi és csomóponti adatbázis felügyeletet, így ez a központi adatbázis felügyelet feladata.) A már működő rendszerek automatikusan képesek a felhasználás gyakoriságát figyelve a kópiák számán és elhelyezkedésén változtatni.

Az adatbázis felügyelő számára több elemzési módszer áll rendelkezésre döntéseihez.

- *A forrás-nyelő elemzés*  
Elsősorban a felhasználás gyakoriságát kell vizsgálnunk. Nem mindegy azonban, hogy a felhasználó csomópont nyelő (azaz csak olvassa az adatot, s így a felhasználáskor elég a legközelebbi példányt megtalálnia), vagy forrás (mely az adatot létrehozta, vagy módosítja).
- *Az ABC elemzés*  
Az adatok kategóriákba sorolhatók „nélkülözhetetlenség” szerint. Ezekhez a kategóriákhoz különböző minimális kópiaszámot rendelhetünk.
- *Az érzékenység elemzés*  
A különböző csomópontok költség/teljesítmény aránya más és más lehet. Nyilvánvaló, hogy a teljesítményt „kevésbé érzékeny” csomópontoknál növelni, az „érzékenyebbeknél” csökkenteni kell.

Említettük már a konzisztencia megőrzésének fontosságát. Ez a *szinkronizációs protokollok* (szabályok) feladata. E protokollok vagy biztosítják az *erős konzisztenciát* (vagyis azt, hogy egy-egy adat kópiái egyszerre változzanak meg), vagy *gyenge konzisztencia* (amikor az adatbázis átmenetileg inkonzisztens marad) mellett gondoskodnak arról, hogy a konzisztencia gyenge volta ne okozhasson problémát. A konzisztencia mérőszáma a *koherencia*, mely erős konzisztenciánál az 1 érték, gyenge konzisztenciánál pedig 1-hez tart. Tehát a koherencia konvergenciája a konzisztencia feltétele.

A szinkronizációs protokollokat két részre oszthatjuk. A központosított protokollokban az egyik (nem feltétlenül mindig ugyanaz) csomópont szerepe kitüntetett, az osztott protokolloknál ilyen nincs.

## Központosított protokollok

### A KÖZPONTI ZÁRELLENŐRZÉS

Egy állandó központ végzi el a változtatásokhoz szükséges lezárásokat. Működése hasonlít az osztatlan adatbázisoknál megszokotthoz. A központ sérülésére érzékeny, csak statikus adatbázisoknál használható, erősen konzisztens módszer.

### A TOKEN MÓDSZER

Egy, a csomópontok között körbejáró token („zseton”) jelöli ki az alkalmi központot, mely a változtatásokhoz szükséges lezárásokat elvégezheti. Igen kedvelt, erősen konzisztens módszer.

### AZ ELSŐDLEGES PÉLDÁNY MÓDSZER

Egy adat kópiáit egy szekvenciába szervezzük, s a változtatások csak ennek mentében történhetnek. Mivel a tranzakciók nem előzhetik meg egymást, nem okoz gondot, hogy a mód-

szer gyengén konzisztens. Előnye, hogy az ideiglenesen működésképtelen csomópontok újra munkába állítása egyszerű.

## Osztott protokollok

### AZ IDŐBÉLYEG MÓDSZER

E módszernél a tranzakciók elindításuk időpontjából és a csomópont azonosítójából egy időbélyeget kapnak, mely meghatározza végrehajtásuk sorrendjét. A távolságból adódó félreértéseket rendszeres üres (dummy) üzenetek küldésével kerülük el. A módszer gyengén konzisztens.

## Az adatok bizalmas kezelése

A modern ipar, kereskedelem, államigazgatás, stb. (vagyis a társadalom) zavartalan működéséhez azt jól kiszolgáló informatikai rendszereket igényel. Alapvető fontosságú ezen rendszerek védelme a behatolás, a szabotázs ellen. Az USA-ban már a 80-as években kimutatták, hogy egy-egy számítógépes bűncselekmény nagyságrendekkel nagyobb kárt okoz, mint egy átlagos bankrablás. A téma iránt érdeklődőknek a [Né84] könyvet ajánlom figyelmébe.

A védelemnek alapvetően három formája van:

- *A fizikai védelem,*  
mely az adatokhoz való illetéktelen hozzáférést fizikailag próbálja megakadályozni (zárt, őrzött számítóközpont, stb.). A modern rendszerekben az adatok ilyenfajta védelme a műholdas adatátvitel elterjedése óta alig lehetséges, hisz az ilyen vonalak vételének megakadályozása fizikailag lehetetlen.
- *Az ügyviteli védelem,*  
mely a követendő biztonságtechnikai szabályokat, kötelező viselkedési módokat, továbbá a kötelező dokumentálás rendjét írja elő (elsősorban a felelősség kérdését szabályozza).
- *Az algoritmikus védelem,*  
olyan algoritmusok összessége, mely a fentieket hatékonyan elősegítő, kiszolgáló algoritmusokat tartalmaz. (Mi csak ezzel foglalkozunk.)

Az algoritmikus védelem is tovább osztható: a *rejtjelezés* és az *üzenethitelesítés* lehetővé teszi az adatok védtelen közegen való továbbítását, a *felhasználó azonosítás* a rendszert használó személyek egyértelmű azonosítására szolgál (ugyanez a feladata számítógépek kapcsolata esetén a *partner azonosításnak*), a *hozzáférés védelem* megakadályozza az egyébként jogos felhasználót abban, hogy hatáskörét túllépje, végül a *digitális kézjegy* az elküldött üzenetek letagadását akadályozza meg.

## Rejtjelezés

Régóta ismeretes, hogy a védhetetlen közegen ájtuttatandó  $x$  üzenetet valamilyen

$$y = E(x)$$

transzformációval érdemes torzítani, természetesen úgy, hogy az üzenet címzettje képes legyen az inverz

$$x = D(y)$$

transzformációra. Ezt az alábbi ábra szemlélteti:



A transzformációktól elvárjuk, hogy (legalábbis az üzenet „érvényességi idején” belül) kívülálló számára megfejthetetlenek legyenek, a címzett ugyanakkor gyorsan és könnyen megértse azokat.

Mivel nem könnyű nagyszámú megfelelő  $E - D$  párt előállítani, nem egy, hanem kétváltozós transzformációkat célszerű alkalmazni, ahol a másik változó az esetenként cserélhető kulcs. Így a transzformációk akár nyilvánosak is lehetnek, csak a kulcsok titkos kezelését kell megoldani. Ha az  $E$  transzformációnál alkalmazandó  $K_r$  rejtőkulcsból ennek  $D$  transzformációjánál alkalmazandó  $K_f$  fejtőkulcs párja könnyen meghatározható, *konvencionális kódolásról* beszélhetünk, ellenkező esetben a kódolás *nyilvános (rejtő) kulcsú*.



### KONVENCIONÁLIS KÓDOLÁS

A legelső konvencionális kódolási eljárás Julius Caesar nevéhez fűződik: üzeneteiben a latin *abc* minden betűje helyett a rákövetkező harmadikat használta.

Főleg katonai körökben alkalmazták a következő eljárásokat.

- **Helyettesítés**  
A nyílt szöveg minden betűjét megadott rend szerint egy másikkal helyettesíthetjük. (Pl. az *abc*-hez egy *abc* permutációt rendelünk.) A nyelvi sajátosságok (betűk gyakorisága) alapján megfejthető.
- **Periodikus helyettesítés**  
Az előzőhöz hasonló, de több helyettesítési szabályt használunk periodikusan cserélve (pl. több *abc* permutációt). Ez is megfejthető: azonos betűkombinációkból lehet a periódus hosszára következtetni, s ezután az egyszerű helyettesítésnél leírtakhoz hasonlóan okoskodni.
- **Kulcsfolyamos rejtés**  
A helyettesítést aperiodikussá tesszük. A helyettesítést egy szöveg vezérli, az *abc* hosszának megfelelő számú különböző *abc* permutációból álló mátrixban a rejtendő szöveg betűjének megfelelő oszlopban és a kulcsszöveg soron következő betűjének megfelelő sorban álló betű lesz a rejtett szöveg következő betűje.  
A nagyméretű mátrix tárolását elkerülendő, választhatjuk az  $i$ -edik sort az *abc*  $i$  lépéses eltolásának. A betűk reprezentálhatóak az *abc*-n belüli sorszámaikkal. Ekkor a kódolás (és a dekódolás) leegyszerűsödik a kódolandó (dekódolandó) és a kulcsszöveg megfelelő betűihez rendelt számértékek (mod *abc* hossz) összeadására. Sajnos ez a kódolás is feltörhető. (Feltörés után mind a kódolt szöveg, mind a kulcsfolyam rendelkezésünkre áll, így egy betűsorozatból két szöveg bontható ki!)

A kulcsfolyamos rejtéshez hasonló kódolási eljárás a számítástechnikában is alkal-

mazható: ha a kódolandó és a kulcsfolyam  $\text{mod } 2$  összeadása a kódolás, ugyanez lehet a dekódolás módja is. A kulcsfolyamot valamilyen véletlenszám generátor állítja elő, ekkor kódolási kulcsként csak ennek elindítási módját kell a partnereknek egyeztetni.

Sajnos az előbbieken vázolt módszer nagyon érzékeny a szinkronításra (egy bit kiesése az egész üzenetet értelmetlenné teszi). Ezért az üzeneteket többnyire blokkokba rendezik, s így küldik el.

- A *rejtjelöltvözés* azt jelenti, hogy több egyszerű transzformációt alkalmazunk egymás után. 1949-ben Shannon javasolta a *keverő transzformációk* bevezetését, mely váltakozva helyettesítések ( $s$ -réteg) és permutációk ( $p$ -réteg) egymásutánjából áll.

A Shannon-féle elvek továbbfejlesztése alapján készítette az IBM a 60-as évek második felében a *LUCIFER* berendezést (128 bites blokkok, 128 bites kulcs, 6-7 emberév fejlesztési munka). Ezt követte az egyetlen LSI áramkörrel elkészíthető *DES* (Data Encryption Standard, 64 bites blokk, 56 bites kulcs). Ez utóbbi 1977 óta USA szabvány. A DES-t számos támadás érte, de az idő a fejlesztőket igazolta. (Diffie, Hellman egy elméleti gépet is szerkesztett a DES-sel kódolt üzenetek feltörésére. A gép elméletileg fél nap alatt feltörte volna az üzeneteket, de óriási teljesítmény-felvétele miatt nem volt megvalósítható, így gondolat kísérlet maradt.)

### NYILVÁNOS (REJTŐ) KULCSÚ KÓDOLÁS

E módszerek alapja valamilyen nehezen invertálható úgynevezett *egyirányú* függvény.

Az eljárásnál ahhoz, hogy a rejtőkulcsból a fejtőkulcs meghatározható legyen, egy több-száz jegyű számot kell prímtényezőkre bontani. Míg egy szám prímtényezőiből való előállítás (összeszorozása) egyszerű és gyorsan megvalósítható munka (néhány száz jegyű számokkal  $1\mu\text{sec}$  alatt lehet műveleteket végezni), ugyanakkor a ma ismert leghatékonyabb eljárással egy 200 jegyű szám prímtényezőkre bontásához  $3.8 \cdot 10^9$  évre van szükség.

Egy másik ismert megoldás, a Merkle-Hellman módszer az úgynevezett lefedési feladaton alapszik.

### Kulcsgondozás

Nyilvánvaló, hogy ha a kulcsok megfelelő védelme nem biztosított, az egész kódolási rendszer értelmetlen. A kulcsok gondozása három feladatból áll.

#### KULCSGENERÁLÁS

Az a művelet, melynek eredményeképpen a kulcsok előállnak. A kulcsgenerálás egy valódi véletlenszám generátor segítségével végrehajtható.

#### KULCSKIOSZTÁS

Kulcskiosztásra az alábbi módszerek ismertek:

- *Alapkulcsok*  
A kulcsok kiosztása történhet egy *alapkulcs készleten* keresztül, melyet rendszeren kívüli eszközökkel juttatnak el a résztvevőkhöz. Az alapkulcsokat, melyek gyakran nyilvános kulcsú rendszerhez tartoznak, csak a kulcsok cseréjéhez használják.
- *Merkle „rejtvény” módszere*  
A hívó fél  $n$  db  $(K_i, I_i)$  párt küld partnerének gyengén kódolva. Az egyet kiválaszt, fel-

töri, és  $I_i$ -t visszaküldi. Ezzel a kommunikáció kulcsa meghatározott. A behatolónak átlagosan a párok felét kell ahhoz feltörnie, hogy megtudja,  $I_i$ -hez melyik  $K_i$  tartozik.

- *A „hatványozós” módszer*

A módszer alapja, hogy az  $i$  és  $j$  felhasználó kitalál egy-egy  $x_i$  ill.  $x_j$  számot.

Egymás között kicserélik az

$$Y_i = \alpha^{x_i} \bmod p$$

illetve az

$$Y_j = \alpha^{x_j} \bmod p$$

számokat ( $p$  prímszám,  $\alpha$  a  $p$  elemű véges test egy primitív eleme), a kommunikáció kulcsa pedig a

$$K = \alpha^{x_i x_j}$$

szám (vagy annak valamilyen függvénye) lehet. Ennek előállítása  $Y_j$  és  $x_i$  vagy  $Y_i$  és  $x_j$  ismeretében egy egyszerű hatványozást igényel, a behatolónak viszont a diszkrét logaritmusképzés a feladata.

## KULCSTÁROLÁS

Nehézségeket okozhat, ha a kulcsokat akár túl kevés, akár túl sok ember ismeri. (Az első esetben lehet, hogy amikor kellene, nem áll rendelkezésre a kulcs, a második esetben nagy a kiszivárgás veszélye.)

Megoldást jelentenek az úgynevezett  $(n, k)$  küszöbrendszerek. E rendszerek lényege, hogy a kulcsot  $n$  darab (nem feltétlenül diszjunkt) részre osztva, bármelyik  $k$  darab kulcsrészletből a kulcs előállítható, de nincs olyan  $k-1$  darab kulcsrészlet, amiből ez megtehető lenne. Ilyen küszöbrendszerek készítésére több matematikai módszer is létezik.

## Felhasználó- és partner-azonosítás, hozzáférés-védelem

### A FELHASZNÁLÓ-AZONOSÍTÁS

- *Jelszóvédelem*

Talán a legrégebbi felhasználó-azonosítási mód a jelszavak használata. Használhatóságát azonban csökkenti, hogy a túl hosszú és értelmetlen jelszavakat nehéz megjegyezni, ugyanakkor a rövid és értelmes szavak gyakran egyszerű próbálkozással kitalálhatóak. Ezért a jelszavakat célszerű gyakran változtatni. Ez többféle alapon történhet, a legbiztosabb valami előre rögzített algoritmust használni. A jelszavakat a számítógépes rendszer nem közvetlenül, hanem valamilyen egyirányú transzformáció alkalmazása után tárolja, így a tároló táblázatból sem lehet azokat megállapítani.

- *Fizikai azonosító használata*

A felhasználók azonosítására gyakran használnak valamilyen fizikai eszközt. A legelterjedtebbek a különféle azonosító kártyák. Ezek nehezen hamisíthatók, de könnyen el lehet őket lopni. Jelentősen fokozhatja a biztonságot a fizikai azonosító eszköz és a jelszó együttes használata.

- *Személyi jellemzők*

Nem lophatók el, és nem hamisíthatók a különféle személyi jellemzők (ujj- vagy tenyérnyomat, rezechártya érzet, stb.). E jellemzők számítógépes tárolása és kiértékelése még nem teljesen megoldott, a biztató eredmények azonban azt sugallják, hogy a jövő útja erre keresendő.

### A PARTNER-AZONOSÍTÁS

A számítógép-számítógép kapcsolatokban is szükség lehet azonosításra. E célra fenntart-hat minden számítógép pár egy-egy kulcsot, ez azonban egy  $n$  elemű hálózatnál  $n^2$  kulcsot jelent. Folyhatnak az információcserek valamilyen hitelesítő központon keresztül, ez azon-ban a kommunikáció költségét növeli jelentősen. Megoldást jelenthet, ha minden csomópont egy, csak a hitelesítő központ által ismert kulccsal tud e központhoz bejelentkezni, s üzenet továbbítási igényét bejelenteni. A központ jelöli ki a kommunikáció kulcsát, melyet a fenti kulccsal kódolva a hívónak visszaküld. Ugyancsak küld emellett egy csomagot, mely a hí-vandó fél kulcsával kódolva tartalmazza a kommunikáció kulcsát s a hívó megjelölését. Ha a hívó e csomagot a hívott félhez továbbítja, a párbeszéd kettejük között folytatódhat, s az azonosítás is kielégítő biztonsággal történt meg.

### ÜZENETHITELESÍTÉS, DIGITÁLIS KÉZJEGY

Az üzenetek hitelesítésének két feltétele van, egyrészt ellenőrizni kell, hogy egy-egy blokkban az és csak az érkezett-e a címzetthez, amit a feladó feladott, másrészt tudnunk kell azt is, hogy az egyes blokkok abban a sorrendben érkeztek-e meg, amilyenben feladták őket (ideértve azt is, hogy nem hiányzik-e közülük). Az első célhoz ellenőrző összegeket, a má-sodikhoz sorszámot célszerű a blokkban elhelyezni még a kódolás előtt.

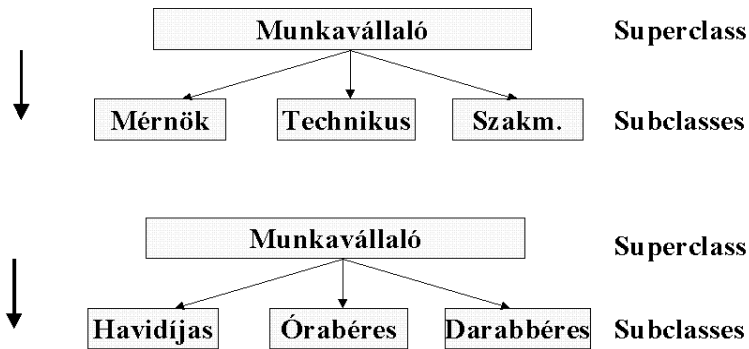
A *digitális kézjegy* arra szolgál, hogy segítségével a címzett megbizonyosodhasson egy üzenet feladójáról, és bizonyíthassa, hogy az illetőtől kapott ilyen üzenetet. Olyasmire van szükség tehát, mint az aláírás, ami könnyen azonosítható, de nehezen hamisítható.

A cél elérhető úgy is, hogy a kényes kommunikációt egy hitelesítő központ közbeiktatá-sával végezzük (mintha tanú előtt beszélénk). Ez a *nem valódi digitális kézjegy*.

Elegánsabb megoldás a *valódi digitális kézjegy*. Ehhez egy nyilvános kulcsú kódolási rendszert lehet felhasználni, mégpedig olyant, mely „megfordítható”, azaz  $E(D(x)) = x$  (az általunk említett módszerek ilyenek). Használjuk a kódolási módszert fordítva, azaz úgy, hogy a titkos fejtő kulccsal rejtünk (és nyilván a nyílt rejtő kulccsal fejtünk). Ekkor a cím-zett, ha a nyílt kulccsal megfejthető üzenetet kap, biztos lehet abban, hogy azt csak a titkos kulcsot ismerő feladó küldhette. Ugyanakkor - mivel a címzett nem ismeri a titkos kulcsot - ha rendelkezik az üzenetnek a nyílt kulccsal megfejthető kódolt változatával, nyilvánvaló, hogy azt ő nem készíthette, vagyis az üzenet a feladótól származik.

### A HOZZÁFÉRÉS-VÉDELEM

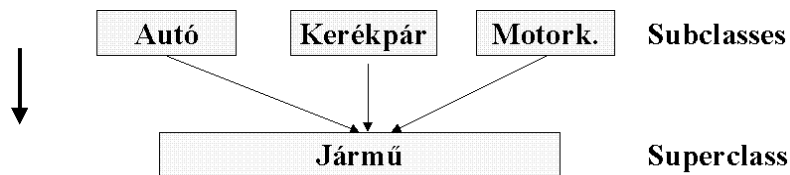
Nem elegendő kiszűrni a jogosulatlan behatolókat, a rendszernek azt is számon kell tartania, hogy a jogos felhasználók hatásköre mire terjed ki. A felhasználó által működtetett *ügynök folyamatok* hatáskörét a *hozzáférési mátrix* szabja meg. Ennek elemeit akár ügynökökhöz, akár adatokhoz kötöten tárolhatjuk.



2. ábra Példák specializációra

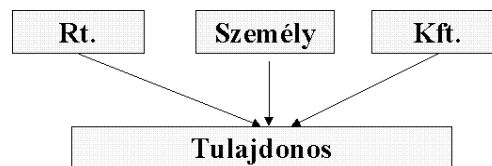
**Generalizáció**

A generalizáció az előző gondolatmenetnek mintegy a fordítottja. Itt a subclass-okból indulunk ki, és megkeressük több subclass elemeinek közös tulajdonságait (3. ábra).



3. ábra Példa generalizációra

Egy különleges subclass az úgynevezett *kategória*, melynek több superclass-a van. A 4. ábrán látható példában a tulajdonos (pl. egy gépkocsi tulajdonosa) egyaránt lehet jogi és természetes személy, a jogi személy lehet Rt. vagy Kft. Mindegyik esetben mást és mást kell a superclass számára nyilvántartani, ezért itt az öröklés szelektív lehet.



4. ábra Példa kategóriára

**Nested Relational Model**

A korábbiakban írtunk a normalizálás jelentőségéről. Minden előnye mellett azonban, a normalizálás természetesen korlátokat is jelent. A Nested Relational Model alkalmazói lazítani kívántak a korlátokon.

E modell annyiban különbözik a szokásostól, hogy nem követeli meg az 1NF normalizálást, az irodalom ezért gyakran N1NF modellként is említi. Olyan esetekben használható jól, ha az objektum hierarchikus struktúrájú [EN94].

Oszám	Onév	Ovez	Dolgozó			Ocím
			Dnév	Gyerekek		
				Gynév	Gykor	

5. ábra Példa nested relational model-re

Az 5. ábra egy olyan adatbázis sémáját mutatja, melyben egy osztály adatai (osztály azonosító száma, osztály neve, osztályvezető, az osztály címe) mellett a dolgozók neve, gyermekeik neve és életkora kerül tárolásra.

Már az Oracle 8 is SQL szinten kezeli a beágyazott táblát.

## Objektum-orientált adatbázis-kezelés

Az objektum-orientált programozási elvek (ezek ismeretét feltételezzük) sikere nyomán felvetődött az ötlet, hogy az adatbázis-kezelést is célszerű lenne hasonló alapokra helyezni. Az adatbázis-kezelés sajátosságai miatt azonban néhány speciális jellegzetességgel számolnunk kell. Így:

- az objektumosztályok állandóak,
- az objektumosztályok osztottak,
- minden objektumnak külön azonosítója van (ez nem azonos a kulccsal, ami különbözhet relációnként),
- megengedettek az összetett objektumok (pl. egy objektumon belüli több reláció, stb.).

Az alábbiakban bemutatjuk, hogy az objektum-orientált programozásnál ismert fogalmak miképpen valósulnak meg

### Zártság

Az értékeket úgynevezett *belső változók* (*instance variable*-ok) tartalmazzák. Ez hasonlít az attribútumhoz, de kívülről rendszerint nem látható, csak az előre definiált *operációk* férhetnek hozzá (metódus).

Egy operációnak két része van:

- a név és a paraméterek (signature, vagy interface),
- az implementáció (method, vagy body).

### Öröklődés és többrétűség

Mind az öröklődés, mind a többrétűség fogalma a szokásosnak megfelelő. Érdekes az operátorok polimorfizmusa, vagyis az a tulajdonság, hogy egy operátor névhez többféle implementáció tartozhat az objektum típusától függően (más néven operator overloading).

Mivel az adatbázis elvileg is a legkülönbébb kapcsolatokat tartalmazhatja, ezek megvalósítása az objektum-orientált elvek (elsősorban a zártság) betartása mellett sok problémát vet fel. Megoldást jelenthet kapcsolatot kereső metódusok alkalmazása, vagy az, ha minden objektum tartalmazza a hozzá kapcsolt objektumok azonosítóját.

## Deduktív adatbázisok

Ezek az adatbázisok a logikai programozás eszközeivel dolgoznak. Így gyakran deklaratív nyelveket (pl. PROLOG) használnak. Ezen a nyelven írják le a *tényeket* és a *szabályokat*. Deduktív adatbázisokat leggyakrabban a *szakértői rendszerek* (expert systems), vagy *tudásbázis kezelő rendszerek* (knowledge base systems) készítésekor használnak.

### Szakértői rendszerek

Az informatika egyik – látszólag témánktól távol eső – területe az úgynevezett *mesterséges intelligencia* (MI, angolul Artificial Intelligence: AI). [Ri91] szerint „Az MI olyan problémák számítógépes megoldásával foglalkozik, melyek megoldásában jelenleg az emberek a jobbak.” [Sá95] felsorolja, hogy mik az emberi problémamegoldás legfőbb jellemzői:

1. hatékony probléma megoldási képesség alternatív lehetőségekkel rendelkező problémák esetén is,
2. kommunikációs képesség,
3. bizonytalan szituációk kezelése,
4. kivételek kezelésének képessége,
5. tanulási képesség.

Az 50-es években általános volt az a meggyőződés, hogy a fentieknek megfelelő rendszer a gondolkodás logikájának algoritmikus utánzásával lehetséges, s erre próbáltak általános megoldást találni. Amikor ennek kivihetlensége bebizonyosodott, a kutatók szűkebb területeken próbálkoztak a megfelelő keresési stratégiák, következtetési módszerek megalkotásával. A 70-es évek végén vált általánossá az a felismerés, hogy „a problémamegoldás képessége nem az alkalmazott formalizmustól és a következtetési módszerektől függ, hanem attól, hogy mennyi, és milyen magasan kvalifikált ismeretanyag áll rendelkezésre az adott tárgyterület vonatkozásában” ([Sá95]). Itt találkozunk e tudományág az adatbázis-kezeléssel, hisz az ismeretanyagot tároló *tudásbázisok* voltaképpen speciális adatbázisok, melyek struktúrája sokszor deduktív alapú.

A szakértői rendszerek felépítése ill. elkészítésének alapvető sémája a következő.

- A felhasználó egy *felhasználói felülettel* (user interface) áll kapcsolatban, mely program a felhasználó és a rendszer közötti dialógust vezérli.
- A felhasználói felület a *következtető rendszerhez* (inference engine) kapcsolódik. Ez a rendszer „lelke”, mely a tudásbázis és esetleg egyéb adatbázisok felhasználásával a válaszokat kidolgozza. Az *okoskodás*, következtetés (reasoning) lehet *adat-* vagy *célvezérelt*.

A tudásbázis szabályok és állítások gyűjteménye, melyet a *tudásmérnök* (knowledge engineer) tölt fel egy *területi szakértő* (domain expert) tudásának számítógépben tárolható alakra „fordításával”. (A szakértői rendszerek – ez idő szerint legalábbis - csak egy-egy elég szűk tudományterületet ölelnek fel.) Fontos, hogy az eredeti és a számítógépes tudás közötti *fordítási szakadék* a lehető legkisebb legyen.

Számos, szakértői rendszerek készítésére alkalmas keretrendszer áll a felhasználók rendelkezésére, melyben a felhasználói felület, a következtető rendszer és a tudásbázis struktúrája adott. Egyes rendszereknél azonban ezek a komponensek is többé-kevésbé változtathatók. A fejlesztés, változtatás a *rendszermérnök* (system engineer) feladata.

A szakértői rendszereknek nem csak biztos állításokat kell kezelniök, mind a rendszerbeli tudás, mind a felhasználó válaszhai tartalmazhatnak bizonytalanságot. Ezt a -100 és 100 közé eső *bizonyossági tényezővel* (CF = certainty factor) írhatjuk le.

Az egyszerű állításokon és összefüggéseken kívül e rendszerek speciális adatelemeket, összetett és hierarchiákban megjelenő *vázakat* (frame) és aktív úgynevezett *démonokat* is tartalmazhatnak.

Alapvető igény a szakértői rendszerekkel szemben, hogy következtetéseiket megmagyarázzák, s ma már az is, hogy képesek legyenek tanulni.

## Adatbányászat

Az adatbázisok alkalmazásának egyik legújabb formája, az úgynevezett *adatbányászat* (data mining). Lényege, hogy a hagyományos felhasználással összehasonlítva, nem pontosan foglalmazzuk meg kérdéseinket, hanem kicsit az adatbázisból szeretnénk megtudni azt is, hogy mit kérdezzünk.

Egy egyszerű példa: míg a „hagyományos” felhasználás kapcsán az adatbázisnak olyan kérdéseket tehetünk fel, mint „a 10 és 20 év közötti korú tanulók hány százaléka használ mobil telefont”, adatbányászás közben efféle is kérdezhetünk: „melyik korosztályra, milyen foglalkozásúakra jellemző a mobil telefon használata”, vagy: „milyen ruházati cikkek jellemzők azokra, akik mobil telefont használnak”, stb.

Az adatbányászat segítségével tehát egy vállalat, vagy intézmény rendelkezésére álló adattömegéből feltárhatók a rejtett összefüggések, ezáltal a meglévő, de rejtett tudás felhasználhatóvá válik. Az ilyen összefüggések ismerete elősegíti például az ügyfelek viselkedésének megértését és előrejelzését, az ügyfelek csoportosítását, vagy akár a műszaki terület adattömegeiben megbúvó minták felderítését. Olyan meglepő eredményekhez is juthatunk így, mint a szakirodalom által (pl. [GU01]) sokszor említett információ: akik eldobható pelenkát vásárolnak, rendszerint sört is vesznek. Így az áruházakban e kettő közé lehet tenni azokat a cikkeket, melyeket a célzott körrel még meg szeretnének vásároltatni.

Egy üzleti információs rendszerben például azt várhatjuk az adatbányászat alkalmazásától, hogy:

1. megelőzi az ügyfelek lemorzsolódását,
2. felderíti az ügyfelek közötti kapcsolatokat,
3. segíti új ügyfelek megszerzését,
4. kideríti a csalásokat,
5. nagy pontossággal jellemzi a ügyfeleket.

Az adatbányászatot a korszerű adatbázis-kezelő rendszerek többsége valamilyen formában támogatja. Kész eszközöket nyújt adatbányászatra pl. az ORACLE (Data Mining Suite, régi nevén Darwin), az IBM (Intelligent Miner) és a Microsoft is (SQL Server Analytic Services). Az egyes eszközök különféle modell típusokat támogatnak, a leggyakrabban a döntési fák, a neurális hálózatok és a clustering használatos. Minderről pl. [FP96]-ban olvashatunk bővebben.

## Az információk integrálása

Napjaink adatbázis-kezelésében egyre nagyobb szerepet kap az információk integrálása. A felhasználó ugyanis nem elégszik meg egy „saját” adatbázis használatával, hanem szélesebb körben, globálisan szeretne információhoz jutni. Ez nem azt jelenti, hogy minden felhasználó teljes elérési jogot kap mindegyik adatbázis használatához (ekkor ugyanis egy, nagyméretű, osztott adatbázis létrehozása lenne a megoldás), hanem, hogy – miközben a lekérdezés módja olyan, mintha csak a „saját” adatbázisával dolgozna – a többi, egyébként független adatbázis megfelelő adatait is eléri.

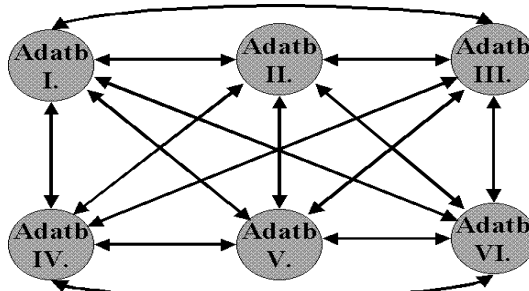
Például tételezzük fel, hogy több könyvkiadó adatbázisában is vannak adatok a kiadott könyvekről, s ezeket valamennyi kiadó felhasználói elérhetik (ugyanakkor az adatbázis más adatait nem feltétlenül).

E feladat megoldása azonban számos problémát vet fel.

- Az – egyébként azonos tartalmú – adatok a különböző adatbázisokban esetleg más formában vannak tárolva (hossz, számbábrázolás, stb.).
- Az adatok kódolása is különbözhet (pl. ha szabványos adatokat, színeket egy-egy kóddal ábrázolnak).
- Nem csak ugyanannak a fogalomnak lehet két adatbázisban más-más neve, de az is előfordulhat, hogy ugyanazon az elnevezésen mást értenek (például „szótár” az egyik kiadónál csak a kétnyelvű szótárakat jelenti, a másiknál az értelmező szótárakat is).
- Egy-egy egyedítípushoz más-más tulajdonság készlet tartozhat (az egyik kiadó nyilvántartja a kiadott példányszámot, de a könyv ajánlott árát nem, a másik fordítva).

## Adatbázisok összekapcsolása

Az egyik lehetséges megoldás az adatbázisok összekapcsolása úgynevezett *adatbázis szövetségbe* (federated databases) úgy, hogy az összekapcsolt adatbázisok bármelyikének felhasználója információhoz juthasson a többi adatbázisból is. Az adatbázisok már említett különbözőségeinek feloldására, bármely két összekapcsolt adatbázis között úgynevezett „fordító” komponenst kell alkalmazni. Ez  $n$  darab összekapcsolt adatbázis esetén  $n(n-1)$ , azaz nagyságrendjében  $n^2$  fordító komponenst jelent (6. ábra). A témát bővebben tárgyalja a [GU01] könyv, s itt találhatunk irodalom felsorolást is.

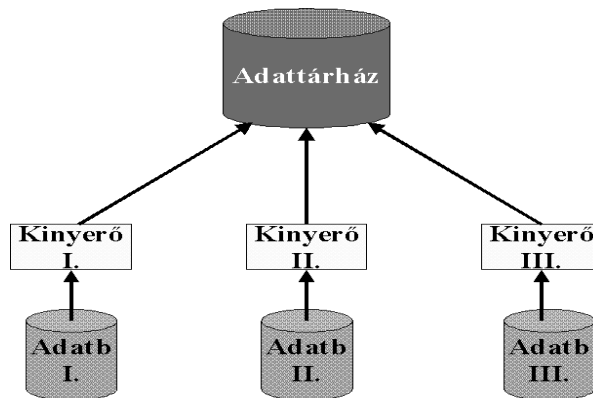


6. ábra Hat összekapcsolt adatbázis 30 fordító komponenssel.

## Adattárház

Ugyanerre a problémára megoldás az úgynevezett *adattárház* létrehozása. Az adatbázisok érintett adatait egy *adatkinyerő* továbbítja az adattárház felé, mely voltaképpen egy hagyományos módon használható adatbázis (7. ábra) két speciális tulajdonsággal:

- adatait különböző adatbázisokból meríti,
- az adatok csak meghatározott időszakonként (pl. éjszakánként) kerülnek frissítésre.

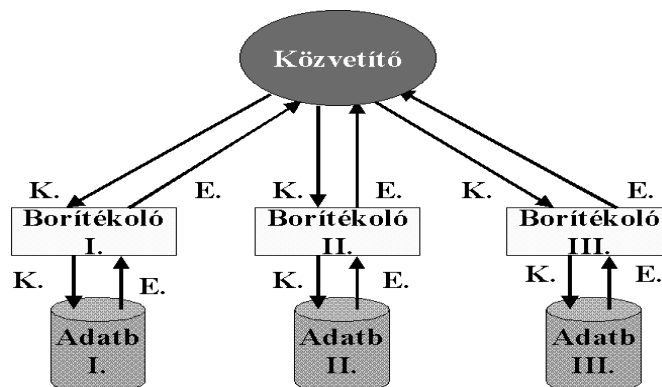


7. ábra Adattárház felépítése

Adattárház tervezéséhez és elkészítéséhez a korszerű adatbázis-kezelő rendszerek komoly támogatást nyújtanak, ilyen eszköz például az Oracle Warehouse Builder.

## Közvetítés

Az adattárház elgondolás továbbvitele lehet, ha az adattárházat fizikailag nem hozzuk létre, hanem egy virtuális adatbázison keresztül tesszük az adatokat elérhetővé. A felhasználó szemszögéből nézve a különbség nem nagy (elsősorban az, hogy itt mindig az aktuális adatokkal dolgozhat), a megoldás menete alapvetően más. A felhasználó által feltett kérdést a közvetítő virtuális adatbázis – mivel adatot ténylegesen nem tárol – továbbadja az úgynevezett *borítékolónak*, s ez szerzi be, majd küldi vissza a *közvetítőnek* az eredményt. A borítékoló végzi el a szükséges fordítási feladatokat is (8. ábra).



8. ábra A közvetítő működése

## II. RÉSZ

# A RELÁCIÓS ADATMODELL

A könyv fő célkitűzése az adatbázis-kezelés gyakorlatának bemutatása, ennek megfelelően e rész tárgyalásában a gyakorlati alkalmazás szempontjából fontos, azt megalapozó relációalgebrára helyezzük a hangsúlyt, és az adatbázis-tervezéshez szükséges függőségek vizsgálatával, a normalizálásokkal csak olyan mélységben foglalkozunk, amely az elméleti áttekinthetéshez feltétlenül szükséges.

A II. rész első fejezete a relációalgebrával foglalkozik, mégpedig a gyakorlattal összhangban a relációkat és a relációműveleteket multihalmazos tárgyalás keretében mutatja be. A lekérdezések, kiválasztások algebrai tervezése két nagy előnnyel jár a szokásos nyelvi módszerekkel szemben. Egyrészt nyilvánvalóan mindenféle implementációs sajátosságtól mentes, másrészt a matematikai objektumokhoz való közvetlen kötődése révén rendkívül tömör, így jól áttekinthető. Végül e fejezet célja az SQL adatbázis-kezelő nyelv algebrai megalapozása is.

A II. rész második és harmadik fejezete foglalkozik a függőségekkel és normalizálásokkal, természetesen azokra az – első fejezetben bevezetett – algebrai fogalmakra építve, amelyek az egzakt tervezés alapját adják. Nem tárgya könyvünknek az adatbázis-tervezés teljes eszköztárának bemutatása, az irodalomban bőségesen utalunk megfelelő – most már magyar nyelven is elérhető – forrásokra az érdeklődők számára.

E rész megértéséhez az Olvasótól csupán alapvető halmazelméleti és algebrai ismereteket tételezünk fel.

## 5. FEJEZET

# Relációalgebra

## Algebrai alapfogalmak

### Jelölések

A halmazokat nagybetűkkel ( $A, B, C, \dots$ ), ezek elemeit általában kisbetűkkel ( $a, b, c, \dots$ ), az üres halmazt a  $\emptyset$  szimbólummal, a természetes számok halmazát  $\mathbf{N}$ -el, egy  $A$  halmaz számosságát pedig  $\mu(A)$  módon jelöljük. Általában az egymást követő egész számok, vagy az ezekkel sorszámozható objektumok felsorolásához a „...” jelölést használjuk. Az  $I_n = \{1, 2, \dots, n\}$  halmazt indexelésre használjuk. Adott  $n \in \mathbf{N}$  esetén  $n$  faktoriális  $n! = 1 * 2 * \dots * n$ .

A halmazokat természetesen rendezetlennek tekintjük, még ha elemeiket az indexük szerint rendezve soroljuk is fel. Egy  $A$  halmazt elemeinek  $\{a_1, a_2, a_3, \dots\}$  felsorolásával, vagy valamely  $T(x)$  tulajdonságával adunk meg, ahol  $T(x)$  valamely ismert halmazon értelmezett logikai függvény, és  $\{x \mid T(x)\}$  jelenti mindazon  $x$ -ek halmazát, melyekre  $T(x)$  teljesül.

Használni fogjuk a logikai algebra „gyorsírás” jeleit. Tehát a szokásos módon „ $\neg$ ” a tagadás, a „ $\forall$ ” a „minden”, a „ $\exists$ ” a „van olyan”, a „ $\nexists$ ” a „nincs olyan”, a „ $\models$ ” a „ból/-ből következik” jele, és a „ $\Leftrightarrow$ ” jel az „akkor és csak akkor” kifejezést szimbolizálja. Például a  $\forall a \in A : T_1 \models T_2$  jelsorozat „kiolvasása”: „minden  $a$  eleme  $A$  esetén, a  $T_1$  állításból következik a  $T_2$  állítás”, továbbá a  $\forall a \in A : \neg(a \in B)$  kifejezés ekvivalens a  $\forall a \in A : a \notin B$  kifejezéssel. (A „:” jel tehát „ $\forall$ ” jelnél az „esetén”, „ $\exists$ ” jelnél a „melyre” szót jelöli.) A logikai ÉS művelet szokásos „ $\wedge$ ” jele helyett gyakran csak vesszőt („,”) írunk, míg a logikai VAGY művelet nevét kiírjuk („vagy”), és a logikai értékeket „IGAZ” és „HAMIS” módon jelöljük. Az univerzális kvantort (a „ $\forall$ ” jelet) a kifejezésekből kiemeljük, tehát  $\forall a \in A : (\forall b \in B : T_{a,b})$  helyett csak  $\forall a \in A, b \in B : T_{a,b}$  kifejezést írunk. Azokat az objektumokat, amelyeket egy függvény-nyel leképezünk (a függvény változóit) a függvény argumentumainak, azokat az objektumokat pedig, amelyeken valamilyen műveletet végzünk a művelet operandusainak fogjuk nevezni. Egy utasításnak a szintaktikailag önálló részeit az utasítás paramétereinek nevezzük.

Az algebrai műveletekhez a hagyományos szimbólumokat használjuk, bár a műveleteket esetenként átértelmezzük. Esetenként előfordulhat, hogy ugyanazt a jelölést több dologra is használjuk, ám a szövegkörnyezetből mindig könnyen eldönthető lesz, hogy éppen melyik értelmezés a helyes.

Végül bevezetjük a „ $\triangleq$ ” szimbólumot a „definíció szerint” kifejezésre (azaz e jel baloldalát a jobboldalán leírtak értelmezik), valamint a bizonyítások végének jelzésére a „ $\blacksquare$ ” jelet.

## A halmazalgebra alapfogalmai

Az alábbiakban áttekintjük a hagyományos halmazalgebra műveleteit. Ez jó lehetőséget ad arra, hogy gyakoroljuk a formális algebra tömör leírást biztosító jelöléseit. Előbb azonban tekintsünk át néhány példát egyszerű halmazok megadására.

### Példa

*Az origó középpontú egységsugarú kör pontjainak halmaza a síkban:*

$$K = \{ \langle x, y \rangle \mid x, y \in \mathbf{R}, x^2 + y^2 = 1 \}, \text{ ahol } \mathbf{R} \text{ a valós számok halmaza, és nyilván } \left\langle \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right\rangle \in K.$$

*Egy egyelemű halmaz:*

$$A = \{ a \}, \text{ és nyilván } a \in A.$$

*Az üres halmaz egy lehetséges definíciója:*

$$\emptyset = \{ x \mid x \in \mathbf{R}, x < 3, x > 5 \}.$$

### Részhalmaz, halmaz bővítése, halmazegyenlőség, hatványhalmaz

Egy  $B$  halmaz *részhalmaza* valamely  $A$  halmaznak, és ezt  $B \subseteq A$  módon jelöljük, ha minden  $b \in B$  esetén  $b \in A$  is teljesül. Ekkor azt is mondjuk, hogy az  $A$  halmaz a  $B$  halmaznak egy *bővítése*. Ezt formálisan az alábbi módon adhatjuk meg:

$$B \subseteq A \Leftrightarrow (\forall x \in B : x \in A).$$

Nyilván az üres halmaz minden halmaz részhalmaza. Valamely  $A, B$  és  $C$  halmazok esetén, ha  $C \subseteq B \subseteq A$ , akkor az  $A$  halmaz a  $C$  halmaznak *tágabb bővítése*, mint a  $B$  halmaz. Tehát egy halmaz legtágabb részhalmaza önmaga. Bevezetjük a *valódi részhalmaz* fogalmát az alábbi módon:

$$B \subset A \Leftrightarrow (B \subseteq A, \exists x \in A : x \notin B),$$

és ha  $B$  valódi részhalmaza az  $A$ -nak, akkor nyilván az  $A$  *valódi bővítése*  $B$ -nek. A *halmazegyenlőség* fogalmát az alábbi módon definiálhatjuk:

$$B = A \Leftrightarrow (B \subseteq A, A \subseteq B).$$

Megjegyezzük, hogy a halmazegyenlőség fenti definíciójának gondolatát sok matematikai bizonyítás alkalmazza. Végül egy  $A$  halmaz *összes részalmazainak halmazát* az  $A$  halmaz *hatványhalmazának* nevezzük és  $2^A$  módon jelöljük. Tehát

$$2^A \triangleq \{ B \mid B \subseteq A \}.$$

Nyilván  $\mu(2^A) = 2^{\mu(A)}$ , és a fentiek alapján  $\emptyset \in 2^A$ , ahol  $\emptyset$  az üres halmaz.

### Példa

Legyen  $A = \{ a, b, c, d \}$ ,  $B = \{ a, b, c \}$  és  $C = \{ c, b, d, a \}$ . Ekkor nyilván  $B \subseteq A$ , sőt  $B \subset A$  is teljesül. Nyilván  $A = C$  és például  $\emptyset \subseteq B$ , továbbá  $2^B = \{ \emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\} \}$ , és  $2^\emptyset = \{ \emptyset \}$ , ahol természetesen  $\{ \emptyset \} \neq \emptyset$ .

## Halmazok metszete

Valamely  $A$  és  $B$  halmazok *metszete*

$$A \cap B \triangleq \{ x \mid x \in B, x \in A \}.$$

Nyilván  $A \cap B \subseteq A$ , és  $A \cap B = B \cap A$ .

### Példa

Legyen  $A = \{ a, b, c, d \}$ ,  $B = \{ c, d, e, f \}$  és  $C = \{ a, b, c \}$ . Ekkor nyilván  $A \cap B = \{ c, d \}$  és  $A \cap C = C$ , továbbá  $A \cap \emptyset = \emptyset$ .

## Halmazvetítés (relatív részhalmaz)

Valamely  $A$  halmaznak azt a részhalmazát, mely egy  $B$  halmazzal való metszete során keletkezik, az  $A$  halmaz *B-vetületének*, vagy másképpen a *B-relatív részhalmazának* nevezzük, és  $A|_B$  módon jelöljük. Tehát

$$A|_B \triangleq A \cap B,$$

ahol  $A$ -t a halmazvetítés *vetített halmazának*,  $B$ -t pedig a *vetítőhalmazának* is nevezzük.

Nyilván  $A|_B = B|_A$ ,  $A|_A = A$ , és természetesen  $B \subseteq A$  esetén  $A|_B = B$ . Egyébként pedig minden vonatkozásban a halmazmetszet műveletére elmondottak teljesülnek itt is.

### Példa

Legyen  $A = \{ a, b, c, d \}$ ,  $B = \{ c, d, e, f \}$  és  $C = \{ a, b, c \}$ . Ekkor nyilván  $A|_B = B|_A = \{ c, d \}$  és  $A|_C = C|_A = C$ , továbbá  $A|_{\emptyset} = \emptyset$ .

## Halmazok különbsége

Valamely  $A$  és  $B$  halmazok *különbsége*

$$A \setminus B \triangleq \{ x \mid x \in A, x \notin B \}.$$

Nyilván  $A \setminus B \subseteq A$ , és könnyen belátható, hogy  $A \cap (B \setminus C) = (A \cap B) \setminus (A \cap C)$ . Ez utóbbi természetesen  $A|_{(B \setminus C)} = A|_B \setminus A|_C$  alakban is felírható.

### Példa

Legyen  $A = \{ a, b, c, d \}$ ,  $B = \{ c, d, e, f \}$  és  $C = \{ a, b, c \}$ . Ekkor nyilván  $A \setminus B = \{ a, b \}$ ,  $B \setminus A = \{ e, f \}$ ,  $A \setminus C = \{ d \}$ ,  $C \setminus A = \emptyset$ , és  $C \setminus \emptyset = C$ .

## Halmazok egyesítése, uniója

Valamely  $A$  és  $B$  halmazok *egyesítése*, vagy *uniója*

$$A \cup B \triangleq \{ x \mid x \in A \text{ vagy } x \in B \}.$$

Nyilván  $A, B \subseteq A \cup B$ , és  $A \cup B = B \cup A$ .

### Példa

Legyen  $A = \{ a, b, c, d \}$ ,  $B = \{ c, d, e, f \}$  és  $C = \{ a, b, c \}$ . Ekkor nyilván  $A \cup B = \{ a, b, c, d, e, f \}$ ,  $A \cup C = A$ ,  $A \cup \emptyset = A$ .

## Halmazok pontozott egyesítése, minősített név

Esetenként szükségünk van arra, hogy a halmazegyesítés után minden elemről el tudjuk dönteni, hogy melyik halmazból származik. Ezt úgy érjük el, hogy minden elem nevét kiegészítjük annak a halmaznak a nevével, ahonnan vettük. Ezt nevezzük az elem *minősített*, vagy *pontozott nevének*. Például egy  $a \in A$  elemnek  $A.a$  a minősített neve. A halmazok pontozott egyesítésében az elemek minősített neveikkel szerepelnek.

Ezekután tehát valamely  $A$  és  $B$  halmazok *pontozott egyesítése*

$$A \dot{\cup} B \triangleq \{ X.x \mid (x \in A \text{ vagy } x \in B), (x \in A \models X = A), (x \in B \models X = B) \}.$$

A megváltozott elemnevek miatt nyilván  $A, B \not\subseteq A \dot{\cup} B$ , de ezúttal is fennáll  $A \dot{\cup} B = B \dot{\cup} A$ .

### Példa

Legyen  $A = \{ a, b \}$ ,  $B = \{ b, c \}$ . Ekkor  $A \dot{\cup} B = \{ A.a, A.b, B.b, B.c \}$ .

## Halmazok szimmetrikus differenciája

Valamely  $A$  és  $B$  halmazok *szimmetrikus differenciája*

$$A \Delta B \triangleq (A \setminus B) \cup (B \setminus A),$$

vagy másként felírva

$$A \Delta B \triangleq \{ x \mid (x \in A, x \notin B), \text{ vagy } (x \notin A, x \in B) \}.$$

Nyilván  $A \Delta B = (A \cup B) \setminus (B \cap A)$ ,  $A \Delta B \subseteq A$ ,  $A \Delta B \subseteq B$ , és  $A \Delta B = B \Delta A$ .

### Példa

Legyen  $A = \{ a, b, c, d \}$ ,  $B = \{ c, d, e, f \}$  és  $C = \{ a, b, c \}$ . Ekkor nyilván  $A \Delta B = \{ a, b, e, f \}$ ,  $A \Delta C = \{ d \}$ ,  $A \Delta \emptyset = A$ .

## Multihalmaz-műveletek

A hagyományos halmazalgebrának azt a kiterjesztését nevezzük *multihalmaz-algebrának*, amelyben a halmazok elem-többszöröződéseket tartalmazhatnak, azaz multihalmazok, a műveletek pedig úgynevezett *multihalmaz-műveletek* (ezeket röviden *multiműveleteknek* is nevezzük). A multiműveletek jellegzetessége, hogy multihalmazokon vannak értelmezve, és az eredményük is multihalmaz.

A multihalmaz-algebra természetesen tartalmazza a hagyományos halmazalgebrát (melyet megkülönböztetésképpen *monohalmaz-algebrának* nevezünk, műveleteit pedig *monoműveleteknek*, amelyek tehát csak monohalmazokon vannak értelmezve). A monohalmazon teljesülő állítások a műveletek egyszerű átírása után azonban nem feltétlenül teljesülnek multihalmazokon.

## Multihalmaz, monohalmaz, redukálás

Az olyan halmazjelölésű objektumot, melyben egy elem többször is előfordulhat, *multihalmaznak* nevezzük. Ha hangsúlyozni akarjuk, hogy minden elem csak egyszer fordulhat elő (azaz hagyományos halmazról van szó), akkor *monohalmazról* beszélünk.

Egy  $B$  monohalmazt egy  $A$  multihalmaz redukáltjának nevezünk és  $B = \text{Reduc}(A)$  módon jelölünk, ha  $B$  az  $A$  összes különböző elemét tartalmazza. A  $\text{Reduc}$  függvényt redukáló függvénynek nevezzük. Ha  $A = \text{Reduc}(A)$ , akkor  $A$  nyilván egy monohalmaz.

#### Megjegyzés

A multihalmaz algebrailag egy olyan indexelt-halmaz (pontosabban indexelő-függvény), melynél az alkalmazásokban nincs jelentősége a sorrendnek. A továbbiakban a multihalmazt a hagyományos halmaz (monohalmaz) általánosításának tekintjük, így elnevezésben általában csak halmazról beszélünk, de általánosítani fogjuk a vonatkozó halmazműveleteket is.

Megjegyezzük, hogy az adatbázis-kezelés gyakorlatában a multihalmazok használata a műveletek gyorsabb végrehajtását eredményezi, mivel nincs szükség az összes elem átvizsgálására az azonosak törlése érdekében.

#### Példa

Legyen  $A = \{ a, a, b, c \}$ , és  $B = \{ a, b, c \}$ . Ekkor nyilván  $B = \text{Reduc}(A)$ .

### Halmaz eleme

Valamely  $a$  objektum eleme egy  $A$  halmaznak, ha  $a \in \text{Reduc}(A)$ . Erre az általánosított elem-tartalmazásra a hagyományos jelölést használjuk, azaz ha  $a \in \text{Reduc}(A)$ , akkor ezt  $a \in A$  módon jelöljük. Ha az  $A$  egy monohalmaz, akkor nyilván az általánosított elem-tartalmazás a hagyományos elem-tartalmazás fogalmába megy át. Ha egy  $b$  objektum nem eleme az  $A$  halmaznak, akkor értelemszerűen a  $b \notin A$  jelölést használjuk.

#### Megjegyzés

A hagyományos algebrai tárgyalásban (ahol tehát a multihalmaz egy indexelt-halmaz), a multihalmaz eleme egy páros, melynek egyik komponense az az objektum, melyet a fentiekben tartalmazott elemnek nevezünk, a másik komponense pedig egy index.

### Multiplikátor, multiplicitás

Egy  $A$  halmaz multiplikátora az  $M_A : \text{Reduc}(A) \rightarrow \mathbf{N}$  függvény, ahol  $\mathbf{N}$  a természetes számok halmaza, és minden  $a \in A$  esetén  $M_A(a)$  megadja, hogy az  $a$  elem hányszor fordul elő az  $A$  halmazban, vagyis az  $a$  elem multiplicitását az  $A$  halmazban.

A továbbiakban, ha valamely  $b$  objektumra  $b \notin A$ , akkor az  $M_A(b) = 0$  jelölést is fogjuk használni. Ha  $A$  egy monohalmaz, akkor nyilván minden  $a \in A$  esetén  $M_A(a) = 1$ .

### Valódi multihalmaz

Ha egy  $A$  halmaznak van legalább egy olyan  $a \in A$  eleme, melyre  $M_A(a) > 1$ , akkor  $A$ -t valódi multihalmaznak nevezzük.

Ez egyben arra is utal, hogy a multihalmazok közé a monohalmazokat is beleértjük, és csak a valódi multihalmazok esetén biztos, hogy nem monohalmazokról beszélünk. Ha  $A \neq \text{Reduc}(A)$ , akkor  $A$  nyilván egy valódi multihalmaz.

## Részhalmaz, halmazegyenlőség

Egy  $B$  halmaz *részhalmaza* valamely  $A$  halmaznak, és ezt  $B \subseteq A$  módon jelöljük, ha minden  $b \in B$  esetén  $M_B(b) \leq M_A(b)$ .

Ha az  $A$  halmaz monohalmaz, akkor nyilván az  $A$  minden részhalmaza is monohalmaz, és természetesen a fenti definíció a hagyományos részhalmaz-értelmezéssel ekvivalens.

Egy monohalmaz bővítése viszont már lehet multihalmaz. Ennek alapján beszélhetünk *monobővítésről* (ez azonos a hagyományos bővítéssel), és *multibővítésről*. A hagyományos „valódi részhalmaz” és „halmazegyenlőség” fogalmat nyilván egyszerűen kiterjeszthetjük multihalmazokra.

### Példa

Legyen  $A = \{a, a, b, c\}$ , és  $B = \{a, b, c\}$ . Ekkor nyilván  $B \subseteq A$ , sőt  $B \subset A$  is teljesül, továbbá  $A \subseteq A$ , és persze  $\emptyset \subseteq A$ .

## Halmazok metszete

Valamely  $A$  és  $B$  halmazok *metszete*  $C = A \cap B$ , ahol egy  $x$  objektum eleme a  $C$  halmaznak, ha  $M_A(x) > 0$  és  $M_B(x) > 0$ . Ekkor minden  $x \in C$  esetén  $M_C(x) = \min(M_A(x), M_B(x))$ . Nyilván egy monohalmaz és egy multihalmaz metszete monohalmaz, és természetesen  $A \cap A = A$ .

Monohalmazok esetén a metszet-fogalom a hagyományos metszet-fogalomba megy át, vagyis ekkor  $A \cap B$  az  $A$  halmaz és a  $B$  halmaz közös elemeit tartalmazza.

A halmazok metszete alapján értelmezzük a *halmazvetítést* (a relatív részhalmazt) multihalmazok esetén is.

## Halmazok különbsége

Valamely  $A$  és  $B$  halmazok *különbsége*  $C = A \setminus B$ , ahol egy  $x$  objektum eleme a  $C$  halmaznak, ha  $M_A(x) - M_B(x) > 0$ . Ekkor minden  $x \in C$  esetén  $M_C(x) = M_A(x) - M_B(x)$ .

Monohalmazok esetén a különbség-fogalom is nyilván a hagyományos halmazkülönbség-fogalomba megy át, vagyis ekkor  $A \setminus B$  azokat az elemeit tartalmazza az  $A$  halmaznak, melyeket a  $B$  halmaz nem tartalmaz.

Mivel  $A \setminus B \subseteq A$ , így a korábban mondottak értelmében, ha az  $A$  halmaz monohalmaz, akkor az  $A \setminus B$  különbség-halmaz is az.

A halmazok különbsége alapján értelmezzük a szimmetrikus differenciát multihalmazok esetén is.

## Halmazok multiuniója

Legyen  $A$  és  $B$  két halmaz, melyeknek lehet közös elemük (azaz  $A \cap B \neq \emptyset$  megengedett).

Az  $A$  és a  $B$  halmazok *multiuniója*  $C = A \uplus B$ , ahol valamely  $x$  objektum eleme a  $C$  halmaznak (azaz  $x \in C$ ), ha  $M_A(x) > 0$  vagy  $M_B(x) > 0$ . Ekkor minden  $x \in C$  esetén  $M_C(x) = M_A(x) + M_B(x)$ . Nyilván ekkor a  $C$  multibővítése mind az  $A$ , mind a  $B$  halmaznak.

A továbbiakban használni fogjuk a *multiunió*, illetve a *monounió módon való bővítés* kifejezéseit, melyek értelemszerűen egy halmaz olyan bővítését jelenti, melynek során a bővítő elem többszörözheti, illetve nem többszörözheti a bővítendő halmaz vele azonos elemét.

*Megjegyzés*

A multiunió már nem egy hagyományos halmazalgebrai művelet (az egyesítés, azaz monounió) általánosítása, hanem egy új művelet. Ha például valamely  $A$  és  $B$  halmazok monohalmazok, és van közös elemük, akkor multiuniójuk már valódi multihalmaz (például  $a \in A \cap B$ ,  $C = A \uplus B$ ,  $D = A \cup B$  esetén  $M_C(a) = 2$ ,  $M_D(a) = 1$ ). Az eddigi műveletek közül ez az első, amely monohalmazokból multihalmazt is előállíthat.

Az előbbiekből következik, hogy ha két monohalmazt egyesíteni akarunk, akkor meg kell fontoljuk, hogy monouniót, vagy multiuniót használjunk. Nyilvánvaló ugyanis, hogy a monohalmazokon, mint speciális multihalmazokon szintén értelmezhető a multiunió. Ügyeljünk arra, hogy valódi multihalmazok között semmiképpen nem végezhető el monounió, még akkor sem, ha nem akarjuk az elemtöbbszöröződések megjeleníteni az eredményhalmazban. Erre a következőkben bevezetésre kerülő, úgynevezett redukált egyesítés szolgál.

## Halmazalgebrai azonosságok

Tekintsük az alábbi, a halmazalgebrából jól ismert két azonosságot annak érdekében, hogy rámutathassunk a multihalmazok és multiműveletek megfontolt kezelésének jelentőségére:

1. Tetszőleges  $A$  és  $B$  halmaz esetén  $A \cap B = A \setminus (A \setminus B)$ .
2. Tetszőleges  $A$ ,  $B$  és  $C$  monohalmazok esetén  $(A \cup B) \setminus C = (A \setminus C) \cup (B \setminus C)$ .

*Megjegyzés*

Könnyen belátható, hogy míg az 1. azonosság a hagyományos halmazalgebrában és a multihalmaz-algebrában egyaránt fennáll, a 2. azonosság már csak a hagyományos halmazalgebrában teljesül általánosan (azaz multiuniót alkalmazva a monounió helyett már nem mindig). Tegyük fel ugyanis, hogy valamely  $x \in A \cap B \cap C$  elemre  $M_A(x) = M_B(x) = M_C(x) = 1$ , és legyen  $D = (A \uplus B) \setminus C$ , valamint  $E = (A \setminus C) \uplus (B \setminus C)$ . Ekkor  $M_D(x) = 1$ , azonban  $M_E(x) = 0$ . Az eltérés oka az, hogy míg  $\{x\} \cup \{x\} = \{x\}$ , addig  $\{x\} \uplus \{x\} \neq \{x\}$ .

Tehát a hagyományos halmazalgebrai azonosságokat ellenőrizni kell a multihalmazok körében, mielőtt ott alkalmaznánk.

## Multiplikált halmaz

Valamely  $A$  halmaz *multiplikáltja* egy  $B$  halmaznak, ha  $\text{Reduc}(A) = \text{Reduc}(B)$  és  $B \subseteq A$ .

Ha egy  $A$  halmaz multiplikáltja egy  $B$  halmaznak, akkor ezt  $A \in \text{Multi}(B)$  módon jelöljük, ahol  $\text{Multi}(B)$  a  $B$  halmaz multiplikáltjainak halmaza. Nyilván tetszőleges  $B$  halmaz esetén (tehát monohalmaz esetén is)  $B \in \text{Multi}(B)$ .

## Redukáló műveletek

A multihalmazzal kapcsolatos fogalmak bevezetésének oka egyszerűen az, hogy a tárgyalásunk alapját képező adattáblák (melyeket a később bemutatásra kerülő relációkkal modellezzünk) valójában multihalmazok, ugyanis egyes műveletek eredményeként lehetnek azonos soraik. Az Oracle-rendszer is több multiműveletet vezet be, az eddig tárgyaltak közül például a *multiuniót* (UNION ALL néven).

Sok multiműveletet azonban a gyakorlati adatbáziskezelő-rendszerek monoműveletszerűen valósítanak meg. Ez alatt azt kell érteni, hogy az egyes műveletek eredményeként már vagy eleve nem jönnek létre az elemtöbbszörözések, vagy az eredményből a rendszer törli azokat. Ennek ellenére e műveleteket is multiműveleteknek kell tekintenünk, hiszen ténylegesen kezelik a multihalmazokat, még ha a műveletek eredményét redukálják is. Az ilyen típusú műveleteket fogjuk *redukáló műveleteknek* nevezni.

A redukáló műveleteket tehát multihalmazokon (azaz mono- és valódi multihalmazokon) értelmezzük, de az eredményük monohalmaz. A megkülönböztetés érdekében azokat a multiműveleteket, melyek nem redukáló típusúak, *valódi multiműveleteknek* nevezzük.

Például az Oracle-rendszerben megvalósított halmazműveletek (a fent említett UNION ALL műveleten kívül) mind redukáló típusúak; az INTERSECT, vagyis a metszet, a MINUS, vagyis a halmazkivonás és a UNION, vagyis az egyesítés.

## Redukált részhalmaz, redukált halmazegyenlőség

Egy  $B$  monohalmaz *redukált részhalmaza* valamely  $A$  multihalmaznak, és ezt  $B \subseteq^R A$  módon jelöljük, ha  $\text{Reduc}(B) \subseteq \text{Reduc}(A)$ .

E fogalom alapján értelmezzük a *redukált valódi részhalmaz*, valamint a *redukált halmazegyenlőség* fogalmát is. Nyilván tetszőleges  $A$  halmaz esetén  $A \subseteq^R A$ , és  $A \stackrel{R}{=} A$ , továbbá minden  $B \in \text{Multi}(A)$  esetén  $B \stackrel{R}{=} A$ .

Ha az  $A$  monohalmaz, akkor a fenti definíciók természetesen a hagyományos részhalmaz-, illetve halmazegyenlőség-értelmezéssel ekvivalensek.

### Példa

Legyen  $A = \{a, a, b, c\}$ ,  $B = \{a, b, c\}$ ,  $C = \{a, b\}$ , és  $D = \{a, b, b, c\}$ . Ekkor nyilván  $B \stackrel{R}{=} A$ ,  $D \stackrel{R}{=} A$ ,  $C \subseteq^R A$ , sőt  $C \subset^R A$  is teljesül, és természetesen  $\emptyset \subseteq^R A$ , de nem igaz, hogy  $B \subset^R A$ , bár  $B \subset A$ .

## Halmazok redukált metszete

Valamely  $A$  és  $B$  halmazok *redukált metszete*

$$A \mathbin{\mathbb{R}} B \triangleq \text{Reduc}(A \cap B).$$

E fogalom alapján értelmezzük a *redukált halmazvetítés* (*redukált relatív részhalmaz*) fogalmát is. Nyilván tetszőleges  $A$  halmaz esetén  $A \mathbin{\mathbb{R}} A \stackrel{R}{=} A$ .

Monohalmazok esetén a redukált metszet fogalma természetesen a hagyományos metszet-fogalomba megy át.

### Példa

A fenti példa halmazai esetén nyilván  $A \mathbin{\mathbb{R}} D = B$ .

## Halmazok redukált különbsége

Valamely  $A$  és  $B$  halmazok *redukált különbsége*

$$A \mathbin{\mathbb{R}} B \triangleq \text{Reduc}(A \setminus B).$$

E fogalom alapján értelmezzük a *redukált szimmetrikus differencia* fogalmát is. Nyilván tetszőleges  $A$  és  $B$  halmazok esetén  $A \setminus^R A = \emptyset$ ,  $A \setminus^R B \subseteq A$ , és  $A \setminus^R \emptyset \subseteq A$ .

Monohalmazok esetén a redukált különbség fogalma is természetesen a hagyományos különbség-fogalomba megy át.

#### Példa

Legyen  $A = \{a, a, b, c, c\}$ ,  $B = \{b, c\}$ , és  $C = \{a, b, b, c\}$ . Ekkor nyilván  $A \setminus^R C = \emptyset$ , és  $A \setminus^R B = \{a\}$ .

### Halmazok redukált egyesítése (uniója)

Valamely  $A$  és  $B$  halmazok *redukált egyesítése*

$$A \uplus B \triangleq \text{Reduc}(A \uplus B).$$

Nyilván tetszőleges  $A$  és  $B$  halmazok esetén  $A \uplus A \subseteq A$ ,  $A \uplus \emptyset \subseteq A$ , és  $A, B \subseteq A \uplus B$ . Láthatóan a redukált egyesítés művelete fogalmilag a hagyományos egyesítéshez, a monounióhoz hasonlít leginkább (csak az nem értelmezett multihalmazok esetén).

#### Példa

Legyen  $A = \{a, a, b, c, c\}$ ,  $B = \{a, b, c\}$ , és  $C = \{a, b, b\}$ . Ekkor nyilván  $A \uplus C = B$ .

## Vektoralgebrai alapfogalmak

### Vektor, komponens, index, monovektor, multivektor

Egy  $a$  halmaz elemeinek valamilyen kötött sorrendű elrendezését az  $a$  halmaz *vektorának* nevezzük, és  $\underline{a}$  módon jelöljük. Használjuk még a *vektor alaphalmaz* fogalmat is, mely egy  $a$  halmaz  $\underline{a}$  vektora esetén maga az  $a$  halmaz.

Egy  $\underline{a}$  vektor hosszát  $\mu(\underline{a})$  jelöli, és a vektor származtatásából következően  $\mu(\underline{a}) = \mu(a)$ , ahol  $\mu(a)$  az  $a$  halmaz elemeinek száma. Nyilván egy  $n$ -elemű halmaznak legfeljebb  $n!$  számú különböző vektora van (ennyiféleképpen lehet ugyanis  $n$  számú elemet sorbarendezni, ha mind különböző).

Annak érdekében, hogy az  $a$  halmaz elemeinek az  $\underline{a}$  vektorbeli helye egyértelműen ki legyen jelölve, az  $a$  halmaz minden eleméhez különböző, pozitív egész számot rendelünk, melyeket *indexnek* nevezünk. Multihalmazban az egyes elemek a multiplicitásuknak megfelelő számú különböző indexet kapnak, monohalmazban tehát minden elem csak egyet. Egy  $\underline{a}$  vektor *indexhalmazát*  $\text{Ind}(\underline{a})$  módon jelöljük. Nyilván  $\text{Ind}(\underline{a}) \subset \mathbb{N}$  és  $\mu(\text{Ind}(\underline{a})) = \mu(\underline{a})$ .

Az  $a$  halmaz elemeit az  $\underline{a}$  vektor *komponenseinek* is nevezzük, tehát egy vektor elemei indexezett komponensek. Az  $\underline{a}$  vektor  $i$  indexű komponensét  $\underline{a}(i)$  jelöli, ahol  $\underline{a}(i) \in a$ , és  $i \in \text{Ind}(\underline{a})$ . A „komponens” szó helyett esetenként használjuk az „elem” szót is, és egy  $\underline{a}$  vektor valamely  $\underline{a}(i)$  komponensére  $a_i \in a$ , illetve  $a_i \in \underline{a}$  módon is fogunk hivatkozni.

Gyakran van szükségünk arra, hogy egy vektor komponenseire – az indexhalmaz nélkül – közvetlenül a vektorban való előfordulásuk sorszáma szerint hivatkozzunk. Ezt az úgynevezett sorszámindex segítségével tehetjük meg. Egy  $\underline{a}$  vektor  $k$ -ik komponensét  $\underline{a}[k]$  jelöli, ahol természetesen  $\underline{a}[k] \in a$ , és  $k \in I_{\mu(\underline{a})}$ , melyet az  $\underline{a}[k]$  komponens sorszámindexének nevezünk. Egy  $\underline{a}$  vektort sorszámindexezett komponenseinek segítségével

$$\underline{a} = \langle \underline{a}[1], \dots, \underline{a}[\mu(\underline{a})] \rangle$$

módon is megadhatunk.

Bevezetjük még az *üres vektor* fogalmát a  $\underline{\emptyset} = \langle \rangle$  módon. Nyilván  $\mu(\underline{\emptyset}) = 0$ , és az üres vektor alaphalmaza az üres halmaz ( $\emptyset$ ).

Esetenként előfordul, hogy egy halmazművelet eredményének (mely persze maga is halmaz) valamely vektorára szeretnénk hivatkozni. Ezt a teljes halmazkifejezés aláhúzásával érjük el. Például valamely  $a$  és  $b$  halmazok metszetének egy vektorát  $\underline{a \cap b}$  módon jelöljük.

Ha egy vektor minden komponense különböző, akkor *monovektornak* nevezzük, ha lehetnek azonos komponensei is, akkor *multivektornak*, és ha valóban vannak is azonos komponensei, akkor *valódi multivektornak* nevezzük. Nyilván a monovektor egy speciális multivektor, és persze a monovektor alaphalmaza monohalmaz, egy multivektoré pedig multihalmaz. A továbbiakban általában multivektorokat tételezünk fel, amikor vektorokat említünk. Ha valahol csak monovektort engedünk meg, akkor azt külön kikötjük.

### Megjegyzés

A vektor definíciója szerint a vektorhoz tehát tartozik egy *indexelő-függvény*, mely elvégzi a kölcsönösen egyértelmű megfeleltetést a vektor alaphalmazának elemei és a vektor komponensei között, vagyis „kiosztja” a komponensek indexeit.

Ezen indexkiosztás (vagyis az indexelés) során az  $a$  halmaz minden eleméhez különböző index kerül, tehát az  $a$  halmaz esetlegesen azonos elemeihez is (vagyis abban az esetben is, ha az  $a$  halmaz egy valódi multihalmaz).

A vektorok jelentőségét az adja, hogy a későbbiekben fontos szerepet játszó Descartes-szorzatnak az elemei vektorok (valójában a Descartes-szorzás adja a vektorok algebrailag pontos definícióját), a relációs adatbázis-kezelés gyakorlatában pedig az adatbázis valamely objektumának egyes tulajdonságait a tulajdonságok rendezett leírásából (a rekordból) éppen a sorrend alapján tudjuk azonosítani.

## Komponenshalmaz-függvény

A vektor definíciója szerint kölcsönösen egyértelmű megfeleltetés áll fenn egy halmaz elemei és a halmaz valamely vektorának komponensei között. Ennek megfelelően, ha egy halmaz valamely vektorához rendelünk egy halmazt oly módon, hogy a vektor egyes elemeit megfosztjuk az indexüktől, akkor visszakapjuk az eredeti halmazt. Az alábbi függvény ezt a hozzárendelést végzi el.

Egy  $\underline{a} = \langle a_1, \dots, a_n \rangle$  vektor komponenshalmaz-függvénye a fenti jelölésekkel:

$$Comp(\underline{a}) \triangleq \bigsqcup_{i \in Ind(\underline{a})} \{ \underline{a}(i) \}. \quad (*)$$

E függvény segítségével egy  $\underline{a}$  vektor komponenseinek *generáló-halmaza*, vagy egyszerűen csak az  $\underline{a}$  vektor alaphalmaza:  $a = Comp(\underline{a})$ .

### Példa

Tekintsük az  $\underline{a} = \langle 3, 2, 2, 5, 1 \rangle$  valódi multivektort. Ennek alaphalmaza a  $Comp(\underline{a}) = \{ 3, 2, 2, 5, 1 \}$  valódi multihalmaz.

## A vektor fogalmának értelmezése

A halmaz és a vektor közötti különbség lényege az, hogy míg a halmazban az elemek felírásának sorrendje tetszőleges (a halmaz nem „tartalmaz” információt elemeinek sorrendjéről, teljesen mindegy ezért, hogy milyen sorrendben írjuk azokat fel), addig a vektort a komponensei és azok sorrendje együttesen határozza meg. Tehát például  $a = \{X, Y, Z\}$ ,  $b = \{Z, X, Y\}$ ,  $\underline{a} = \langle X, Y, Z \rangle$ ,  $\underline{b} = \langle Z, X, Y \rangle$  esetén  $a = b$ , és  $\underline{a} \neq \underline{b}$ .

Ne tévesszen meg bennünket, hogy az  $a = \{a_1, \dots, a_n\}$  halmaz és az  $\underline{a} = \langle a_1, \dots, a_n \rangle$  vektor jelölésében az indexelés sorrendje azonos! Bár az  $\underline{a}$  vektorbeli komponensek sorrendjét valóban az indexelés definiálja, az  $a$  halmaz azért még rendezetlen, és itt az indexelés csak az egyes elemek megkülönböztetését szolgáló jelölés.

A komponens-indexek azonban a vektorok elméleti tárgyalásában is csupán jelzik a komponensek rendezettségét, ugyanis egy  $a = \{a_1, \dots, a_n\}$  halmaz legfeljebb  $n!$  számú különböző vektorából az  $\langle a_1, \dots, a_n \rangle$  csak egyet reprezentál, vagyis nincs jelentősége annak, hogy az elemek indexei ugyanolyan sorrendben vannak a halmazmegadásban, mint a vektormegadásban. Egyébként a vektorok indexhalmaza gyakran  $I_n$ , de nem mindig. Ennek megfelelően nyilván  $\underline{b} = \langle b_2, b_3, b_6, b_8 \rangle$  is egy vektor.

A definíció fontos része a „halmaz elemeinek valamilyen kötött sorrendű elrendezését” kitétel, mely arra utal, hogy egyrészt egy vektor halmazból való előállításakor a halmaz minden elemét felhasználjuk, másrészt pedig semmilyen más elemet nem használunk fel (beleértve ebbe azt, hogy még ugyanazt az elemet sem használhatjuk fel többször). Eszerint tehát egy vektornak csak akkor lehetnek azonos komponensei, ha az őt generáló halmaz egy valódi multihalmaz. Ezt fejezi ki a komponenshalmaz-függvényt definiáló (\*) összefüggés is, mely szerint egy halmaz elemei és valamely hozzárendelt vektor komponensei között kölcsönösen egyértelmű leképezés létezik. A (\*) összefüggés lényege tehát, hogy minden vektorhoz egyértelműen tartozik egy (multi)halmaz, és minden halmazhoz hozzárendelhető (a komponens-sorrendtől eltekintve egyértelműen) egy vektor.

Végül figyeljünk fel arra, hogy – hasonlóan a multihalmazhoz – egy vektor is egy indexelt halmaz, ám a vektornál – ellentétben a multihalmazzal – már nagyon is fontos a sorrend. A multihalmazok esetén a „Halmaz eleme” fogalom kifejtésénél már jeleztük, hogy egy halmaz elemeinek indexelésével ezen elemeket tulajdonképpen olyan összetett elemekkel – párosokkal – helyettesítjük, melyekben az egyik összetevő maga az eredeti elem (vektorok esetén ezt komponensnek neveztük), a másik pedig az index. Az „eleme” fogalom, illetve az „ $\in$ ” szimbólum használatának vektorokra való kiterjesztését majd a Descartes-szorzás fogja algebraiailag pontosan értelmezni.

## Normál vektor, vektor normál alakja

Ha egy  $n$  hosszúságú  $\underline{a}$  vektor indexhalmaza  $Ind(\underline{a}) = I_n$ , akkor az  $\underline{a}$  vektort *normál vektornak* nevezzük. Általában bennünket nem is érdekel egy vektornak az indexhalmaza, csak komponenseinek sorrendje. Ekkor a vektort komponenseinek indexelése nélkül, azok rendezett sorozataként adjuk meg (például  $\underline{a} = \langle x, y, z \rangle$  alakban). Ilyenkor jellemzően a vektort normál vektornak tekintjük. Természetesen minden vektorhoz egyértelműen hozzárendelhető az a normálvektor, melyben a komponensek rendezett sorozata azonos. Egy  $\underline{a}$  vektorhoz ilyenmódon hozzárendelhető normál vektort az  $\underline{a}$  vektor *normál alakjának*, vagy röviden *normáltjának* nevezzük és  $\underline{a}^{\text{norm}}$  módon jelöljük.

Tehát tetszőleges  $n$ -hosszúságú  $\underline{a}$  vektor és  $k \in I_n$  sorszámindex esetén

$$\underline{a}^{\text{norm}}(k) \triangleq \underline{a}[k], \text{ vagyis}$$

$$\text{Ind}(\underline{a}^{\text{norm}}) = I_n, \text{ és}$$

$$\underline{a} = \langle \underline{a}^{\text{norm}}(1), \dots, \underline{a}^{\text{norm}}(n) \rangle.$$

## Ekvivalens vektorok

Tetszőleges  $\underline{a}$  és  $\underline{b}$  vektorok *ekvivalensek* és ezt

$$\underline{a} \equiv \underline{b}$$

módon jelöljük, ha normál alakjuk (vagyis komponens-sorrendjük) azonos, azaz

$$\underline{a}^{\text{norm}} = \underline{b}^{\text{norm}}.$$

Az ekvivalens vektorokra nyilvánvalóan fennállnak az alábbi állítások:

### Állítás

Ekvivalens vektorok alaphalmazai azonosak, azaz tetszőleges  $\underline{a}$  és  $\underline{b}$  vektorok esetén

$$\text{ha } \underline{a} \equiv \underline{b}, \text{ akkor } \text{Comp}(\underline{a}) = \text{Comp}(\underline{b}), \text{ azaz } \underline{a} = \underline{b}.$$

### Állítás

A vektorok ekvivalenciája

1. *reflexív*,  
azaz tetszőleges  $\underline{a}$  vektor esetén  
$$\underline{a} \equiv \underline{a},$$
2. *szimmetrikus*,  
azaz tetszőleges  $\underline{a}$  és  $\underline{b}$  vektorok esetén  
ha  $\underline{a} \equiv \underline{b}$ , akkor  $\underline{b} \equiv \underline{a}$  is teljesül, és
3. *transzítív*,  
azaz tetszőleges  $\underline{a}$ ,  $\underline{b}$  és  $\underline{c}$  vektorok esetén  
ha  $\underline{a} \equiv \underline{b}$ , és  $\underline{b} \equiv \underline{c}$ , akkor  $\underline{a} \equiv \underline{c}$  is teljesül.

## Alvektor

Legyen  $\underline{a}$  és  $\underline{b}$  két vektor, valamint  $a$  és  $b$  ezek alaphalmazai. Ekkor a  $\underline{b}$  *alvektora* az  $\underline{a}$  vektornak, azaz

$$\underline{b} \subseteq \underline{a},$$

ha van olyan  $\underline{c}$  vektor, melyre

1.  $\underline{c} \subseteq \underline{a}$ , ahol  $\underline{c}$  a  $\underline{a}$  vektor alaphalmaza,
2.  $\text{Ind}(\underline{c}) \subseteq \text{Ind}(\underline{a})$ , és
3. minden  $i \in \text{Ind}(\underline{c})$  esetén  $\underline{c}(i) = \underline{a}(i)$ , és
4.  $\underline{c} \equiv \underline{b}$ .

Ha  $\underline{b} \subset \underline{a}$ , akkor a  $\underline{b} \subsetneq \underline{a}$  jelölést is használjuk. Nyilván tetszőleges  $\underline{a}$  vektor esetén  $\emptyset \subseteq \underline{a}$  (vagyis az üres vektor minden vektor alvektora), továbbá  $\underline{a} \subseteq \underline{a}$ .

**Megjegyzés**

A definíció szerint tehát egy vektorból úgy származtathatunk legegyszerűbben alvektorokat, hogy egyes komponenseit elhagyjuk. További alvektorokat alkotnak az ezekkel ekvivalens vektorok. Az alvektor legfontosabb tulajdonsága tehát az, hogy a tartalmazott komponensekre vonatkozóan megőrzi az eredeti vektor *rendezettségét*.

Az alvektor fogalmának bevezetését láthatóan az tette lehetővé, hogy a vektor komponenseinek egy rendezett halmazzal (a természetes számok  $\mathbb{N}$  halmazának egy részhalmazával) való indexelése által rendeztük magukat a komponenseket is.

A fenti definícióban az  $\underline{a}$  és  $\underline{b}$  vektorok természetesen lehetnek valódi multivektorok is. Az  $\underline{a} = \langle 1, 2, 1, 2 \rangle$  és  $\underline{b} = \langle 1, 2 \rangle$  vektorok esetén csak  $\underline{b} \subseteq \underline{a}$  teljesül, az  $\underline{a} \subseteq \underline{b}$  nem.

**Példa**

Tekintsük az  $\underline{a} = \langle 'A', 'L', 'M', 'A' \rangle$ ,  $\underline{b} = \langle 'A', 'L' \rangle$ ,  $\underline{c} = \langle 'L', 'A' \rangle$ , és  $\underline{d} = \langle 'M', 'L' \rangle$  vektorokat. Nyilvánvalóan  $\underline{a} \subseteq \underline{a}$ ,  $\underline{b} \subseteq \underline{a}$ , és  $\underline{c} \subseteq \underline{a}$  egyaránt teljesül, míg  $\underline{d} \subseteq \underline{a}$  nem teljesül.

**Közös gyök**

1. Két vektor közös gyökének nevezünk egy nem-üres vektort, ha az mindkettőnek alvektora. Tehát valamely  $\underline{a}$  és  $\underline{b}$  vektorok *közös gyökeinek halmaza*

$$q(\underline{a} \mid \underline{b}) \triangleq \{ \underline{c} \mid \underline{c} \subseteq \underline{a}, \underline{c} \subseteq \underline{b}, \underline{c} \neq \emptyset \},$$

és értelmezzük a *legnagyobb közös gyökeinek halmazát* is az alábbi módon

$$Q(\underline{a} \mid \underline{b}) \triangleq \{ \underline{c} \mid \underline{c} \in q(\underline{a} \mid \underline{b}), \forall \underline{d} \in q(\underline{a} \mid \underline{b}) : \mu(\underline{d}) \leq \mu(\underline{c}) \}.$$

Nyilván fennáll a  $q(\underline{a} \mid \underline{b}) = q(\underline{b} \mid \underline{a})$  és  $Q(\underline{a} \mid \underline{b}) = Q(\underline{b} \mid \underline{a})$  szimmetria tulajdonság, azaz e halmazok függetlenek a paraméterek sorrendjétől.

2. A közös gyök fogalmát kiterjesztjük azokra az esetekre is, ahol a két objektum valamilye, vagy akár mindegyike halmaz. Tekintsük tehát az  $\underline{a}$  és  $\underline{b}$  vektorokat, valamint  $a$  és  $b$  halmazokat. Ekkor a *közös gyökök halmaza*

$$q(\underline{a} \mid b) \triangleq \{ \underline{c} \mid \underline{c} \subseteq \underline{a}, c \subseteq b, \underline{c} \neq \emptyset \},$$

$$q(a \mid \underline{b}) \triangleq \{ \underline{c} \mid c \subseteq a, \underline{c} \subseteq \underline{b}, \underline{c} \neq \emptyset \},$$

melyekben  $c$  a  $\underline{c}$  vektor alaphalmaza, és

$$q(a \mid b) \triangleq \{ c \mid c \subseteq a, c \subseteq b, c \neq \emptyset \},$$

továbbá természetesen a fentihez hasonló módon értelmezzük a *legnagyobb közös gyökök halmazát* is. Nyilván a fenti szimmetria tulajdonság ekkor is fennáll.

Könnyen belátható, hogy  $q(\underline{a} \mid \underline{b}) \subseteq q(\underline{a} \mid b)$ , és természetesen  $q(\underline{a} \mid \underline{b}) \subseteq q(a \mid \underline{b})$ .

**Példa**

Tekintsük az  $\underline{a} = \langle 'F', 'E', 'H', 'É', 'R' \rangle$  és  $\underline{b} = \langle 'É', 'B', 'E', 'R' \rangle$  vektorokat. Ekkor  $q(\underline{a} \mid \underline{b}) = \{ \langle 'E' \rangle, \langle 'É' \rangle, \langle 'R' \rangle, \langle 'E', 'R' \rangle, \langle 'É', 'R' \rangle \} = q(\underline{b} \mid \underline{a})$ ,  $q(\underline{a} \mid b) = \{ \langle 'E' \rangle, \langle 'É' \rangle, \langle 'R' \rangle, \langle 'E', 'É' \rangle, \langle 'E', 'R' \rangle, \langle 'É', 'R' \rangle, \langle 'E', 'É', 'R' \rangle \} = q(b \mid \underline{a})$ ,  $q(a \mid \underline{b}) = \{ \langle 'É' \rangle, \langle 'E' \rangle, \langle 'R' \rangle, \langle 'É', 'E' \rangle, \langle 'É', 'R' \rangle, \langle 'E', 'R' \rangle, \langle 'É', 'E', 'R' \rangle \} = q(\underline{b} \mid a)$ ,  $q(a \mid b) = \{ \{ 'E' \}, \{ 'É' \}, \{ 'R' \}, \{ 'E', 'É' \}, \{ 'E', 'R' \}, \{ 'É', 'R' \}, \{ 'E', 'É', 'R' \} \} = q(b \mid a)$ .

## Hasonló vektorok

Két vektor hasonlóságát felépíthetjük a közös gyök fogalmára. A legegyszerűbb megközelítés szerint valamely  $\underline{a}$  és  $\underline{b}$  vektorok *hasonlóak*, ha van közös gyökük, vagyis

$$\underline{a} \sim \underline{b},$$

ha van olyan  $\underline{c} \neq \underline{0}$  vektor, melyre  $\underline{c} \subseteq \underline{a}$ , és  $\underline{c} \subseteq \underline{b}$ , azaz

$$\underline{a} \sim \underline{b} \Leftrightarrow q(\underline{a} | \underline{b}) \neq \emptyset.$$

Ha  $\underline{b} \subseteq \underline{a}$ , akkor nyilván  $\underline{b} \sim \underline{a}$  is teljesül, és az ekvivalens vektorok hasonlóak is.

### Állítás

A vektorok hasonlósága

1. *reflexív*,  
azaz tetszőleges  $\underline{a}$  vektor esetén  
 $\underline{a} \sim \underline{a}$ , és
2. *szimmetrikus*,  
azaz tetszőleges  $\underline{a}$  és  $\underline{b}$  vektorok esetén  
ha  $\underline{a} \sim \underline{b}$ , akkor  $\underline{b} \sim \underline{a}$  is teljesül.

### Megjegyzés

A fenti két állítás a definíciók közvetlen következménye. Tulajdonképpen e két állításnál érdekesebb egy harmadik (negatív) állítás, amely szerint a vektorok hasonlósága nem tranzitív, azaz tetszőleges  $\underline{a}$ ,  $\underline{b}$  és  $\underline{c}$  vektorok esetén, abból hogy  $\underline{a} \sim \underline{b}$ , továbbá  $\underline{b} \sim \underline{c}$  teljesül, még egyáltalán nem biztos, hogy  $\underline{a} \sim \underline{c}$  is teljesül. (Algebrailag ez a legfontosabb különbség a vektorok ekvivalenciája és hasonlósága között.)

Megjegyezzük, hogy éppen a hasonlóság fenti tulajdonságai miatt *hasonlósági relációnak* nevezzük mindazon relációkat, melyek reflexívek és szimmetrikusak. Speciális esetként persze az ekvivalencia is hasonlósági reláció, melyet azonban (mivel rá további tulajdonságként a tranzitivitás is teljesül) külön néven *ekvivalencia relációnak* nevezünk.

A tranzitivitás tehát általában nem teljesül hasonlóság esetén. Tekintsük ugyanis az  $\underline{a} = \langle 'A', 'L', 'O', 'M' \rangle$ , a  $\underline{b} = \langle 'L', 'O', 'M', 'B' \rangle$ , valamint a  $\underline{c} = \langle 'B', 'Á', 'B' \rangle$  vektorokat. Nyilván  $\underline{a} \sim \underline{b}$ , és  $\underline{b} \sim \underline{c}$  teljesül, míg viszont  $\underline{a} \sim \underline{c}$  nem teljesül.

### Példa

Tekintsük az  $\underline{a} = \langle 'A', 'L', 'M', 'A' \rangle$ , és a  $\underline{b} = \langle 'F', 'A', 'L', 'A', 'P' \rangle$  vektorokat. Mivel  $q(\underline{a} | \underline{b}) = \{ \langle 'A', 'L' \rangle, \langle 'A', 'A' \rangle, \langle 'A', 'L' \rangle, \langle 'A', 'A' \rangle, \langle 'L', 'A' \rangle, \langle 'A', 'L', 'A' \rangle \} \neq \emptyset$ , így az  $\underline{a}$  és  $\underline{b}$  vektorok hasonlóak, azaz  $\underline{a} \sim \underline{b}$ , továbbá legnagyobb közös gyökük az  $\langle 'A', 'L', 'A' \rangle$  vektor, melyre természetesen  $\langle 'A', 'L', 'A' \rangle \sim \underline{a}$ ,  $\langle 'A', 'L', 'A' \rangle \sim \underline{b}$ .

## Vektorok hasonlósági függvénye

A fenti hasonlóság definícióban két vektor már akkor is hasonlóknak minősül, ha csak egyetlen közös komponensük van. Ezért gyakran érdekes az olyan megközelítés, mely a hasonlóság fogalmához értéket is rendel. Például mondhatjuk, hogy két vektor hasonlóság értéke

legyen a legnagyobb közös gyökük méretének és a két vektor közül a hosszabb méretének a hányadosa. Ekkor valamely  $\underline{a}$  és  $\underline{b}$  vektorok *hasonlósági függvénye*

$$H(\underline{a}, \underline{b}) = \frac{\mu(\underline{c})}{\mu(\underline{d})},$$

ahol  $\underline{c} \in Q(\underline{a} \mid \underline{b})$ , és

$$\underline{d} = \text{Max}(\underline{a}, \underline{b}),$$

melyben

$$\text{Max}(\underline{a}, \underline{b}) = \underline{a}, \text{ ha } \mu(\underline{b}) \leq \mu(\underline{a}), \text{ és}$$

$$\text{Max}(\underline{a}, \underline{b}) = \underline{b}, \text{ egyébként.}$$

Könnyen belátható, hogy e hasonlósági függvény szimmetrikus (azaz  $H(\underline{a}, \underline{b}) = H(\underline{b}, \underline{a})$ ) és normalizált (azaz  $H(\underline{a}, \underline{b}) \in [0,1]$ ).

#### Megjegyzés

A hasonlóság fogalmak általános tulajdonsága a szimmetria, de a hasonlósági függvény-nyel szemben sokszor elvárunk egyéb mértékszerű tulajdonságokat is. Egy  $M$  halmaz, és tetszőleges  $x, y, z \in M$  esetén az  $M$  felett értelmezett  $\rho$  valós függvény metrikát alkot, ha *a.)*  $\rho(x, y) \geq 0$ , *b.)*  $\rho(x, y) = \rho(y, x)$ , *c.)*  $(\rho(x, y) = 0) \Leftrightarrow (x = y)$ , *d.)*  $\rho(x, y) + \rho(y, z) \geq \rho(x, z)$ . Ez a metrika azonban láthatóan távolság fogalom. Ahhoz, hogy ebből hasonlóságszerű függvény legyen, valamilyen módon komplementst kell képezni belőle. Legegyszerűbb a reciprokot képezni, persze ekkor az azonosságot a „végtelen” hasonlósági „érték” képviseli. Ha – mint a fenti esetben is – a hasonlósági függvény normalizált, akkor a maximális értékre vett additív komplementstre – jelen esetben az  $(1 - H(\underline{a}, \underline{b}))$  értékre – előírhatjuk, hogy az legyen metrika. Általában azonban nincs szükségünk ilyen szigorú feltételre. A reflexivitásnak megfelelő *c.)* tulajdonság (mely tehát az objektumok önmagukkal való maximális hasonlóságát fejezi ki), valamint a *b.)* szimmetria tulajdonság előírása legtöbbször elegendő.

Természetesen a fentitől eltérő sok más módszer is alkalmazható a vektorhasonlóság kifejezésére. A legegyszerűbb, ha az összehasonlítandó vektorok közös komponenseinek mennyiségéből indulunk ki, ám ez nem tükrözi a vektorok szerkezeti sajátosságait. Jobb megoldáshoz jutunk, ha a két vektor közös gyökeket tartalmazó halmaz számosságát tekintjük a hasonlóság értékének, ám ez nehezen normalizálható (a  $[0,1]$  zárt tartományra). Még jobb, ha az összehasonlítandó vektorok komponenseinek egymáshoz viszonyított rendezetlenségét vesszük alapul, mely rendezetlenség mérésére a permutációk elméletéből ismert inverzió fogalma lehet alkalmas. Sokszor célszerű összetett mérték fogalmat értelmezni, és esetenként a kiszámítás hatékonysága érdekében feladjuk még a szimmetria követelményt is.

Mivel az adatbázisok témakörében számunkra a legegyszerűbb hasonlóság értelmezés is megfelel addig, amíg adatbázisok (adattáblák) szerkezetét hasonlítjuk össze, így a továbbiakban csak az  $\underline{a} \sim \underline{b}$  típusú hasonlóság fogalmat, vagyis a hasonlóságot, mint hasonlósági relációt használjuk.

### Vektorvetítés (halmaz-relatív és vektor-relatív alvektor)

A vektorvetítéssel kiterjesztjük a halmazvetítés (a relatív részhalmaz) fogalmát a vektorokra. Ez tárgyalásunk egyik legfontosabb segédfogalma, nagyon sok további fogalmat fogunk erre

építeni. Általános értelmezése azonban elég bonyolult (ekkor meghatározása a legnehezebb algoritmuselméleti problémák közé tartozik). Mivel a nehézségeket alapvetően az okozza, hogy a vetített vektor lehet multivektor is, ezért a fogalmat először monovektorokra vezetjük be. (Szemléletessége miatt ezt nevezhetjük a vektorvetítés intuitív definíciójának is.)

A monovektor-vetítés fogalmának bevezetése azért is igen fontos, mivel a vektorvetítés származtatott fogalmaiban – az adatbázis-kezelés témakörén belül maradván – a vetített vektor egy tulajdonságvektor lesz (ezt majd attribútumvektornak fogjuk nevezni) mely a természetéből következően mindig monovektor.

### MONOVEKTOR-VETÍTÉS

Legyen  $\underline{a}$  egy vektor,  $a$  ennek alaphalmaza, és  $x$  egy másik halmaz. Ekkor az  $\underline{a}$  vektor  $x$ -vetületeként (az  $\underline{a}$  vektornak az  $x$  halmazra vonatkozó alvektoraként, vagyis az  $x$ -relatív alvektoraként) tekintjük az  $\underline{a}$  vektornak a leghosszabb olyan alvektorát, mely csupa  $x$  halmazbeli elemet tartalmaz, vagyis a legnagyobb közös gyök egy elemét, azaz legyen

$$\underline{a}|_x \in Q(\underline{a} | x),$$

ahol az  $\underline{a}$  vektort a vetítés *vetített vektorának* nevezzük.

Ezt a vektorvetítést *monovektor-vetítésnek*, nevezzük, mivel (mint az alábbi példákban látni fogjuk) csak monovektorok esetén ad egyértelmű eredményt. (Megjegyezzük, hogy ha  $x$  egy multihalmaz, akkor a  $\subseteq$  műveletet természetesen multiműveletként kell értelmezni.)

#### Példa

Legyen  $\underline{a} = \langle 'F', 'E', 'H', 'É', 'R' \rangle$  és  $\underline{x} = \langle 'É', 'B', 'E', 'R' \rangle$ . Ekkor

$$\underline{a}|_x \in \{ \langle 'E', 'É', 'R' \rangle \}, \text{ és}$$

$$\underline{x}|_a \in \{ \langle 'É', 'E', 'R' \rangle \}.$$

E példa jól szemlélteti az alábbi állításokat.

#### Állítás

Ha  $\underline{a}$  monovektor, akkor tetszőleges  $x$  halmaz esetén

1.  $\underline{a}|_x$  egyértelműen meghatározott,
2.  $\underline{a}|_x$  is monovektor, és
3.  $\text{Comp}(\underline{a}|_x) = a|_x$ .

#### Példa

Figyeljük meg a definíció eredményét multivektor esetén. Legyen  $\underline{a} = \langle 'A', 'L', 'M', 'A' \rangle$  és  $\underline{x} = \langle 'F', 'A', 'K', 'A', 'L', 'A', 'P' \rangle$ . Ekkor

$$\underline{a}|_x \in \{ \langle 'A', 'L', 'A' \rangle \}, \text{ és}$$

$$\underline{x}|_a \in \{ \langle 'A', 'A', 'L' \rangle, \langle 'A', 'L', 'A' \rangle, \langle 'A', 'L', 'A' \rangle \}.$$

#### Megjegyzés

Láthatóan valódi multivektorok esetén a vektorvetület részben ekvivalens, részben nem ekvivalens vektorokat eredményez.

Az ekvivalens eredmények kezelése nem okoz problémát, hiszen a gyakorlatban általában úgyis csak a normál alakok érdekelnek bennünket. A nem ekvivalens eredményvektorok

azonban jelzik, hogy e definíciót csak akkor célszerű használni, ha a vetített vektor mono-vektor. (Ezt erősíti az is, hogy intuíciónk szerint az  $\underline{x}|_a$  vetítés megfelelő eredménye az  $\langle 'A', 'A', 'L' \rangle$  vektor.

Hasonló a helyzet  $\underline{a} = \langle 'A', 'B', 'A' \rangle$  és  $\underline{x} = \langle 'B', 'A' \rangle$  vektorok esetén is, ahol bár az  $\underline{a}$  vektor  $x$ -vetületeinek halmaza  $\{\langle 'A', 'B' \rangle, \langle 'B', 'A' \rangle\}$ , azonban csak az  $\underline{a}|_x = \langle 'A', 'B' \rangle$  eredményt érezzük helyesnek. (Megjegyezzük, hogy ebben az esetben  $\underline{a}|_x = \langle 'A', 'B', 'A' \rangle$  nem jöhet szóba, mivel a  $\subseteq$  műveletet multiműveletként való értelmezése ezt kizárja.)

Ezen észrevételek alapján ugyan javíthatnánk a fenti definíciót annak érdekében, hogy valódi multivektorra is megfelelő legyen, ám így épp a definíció szemléletességét veszíténénk el. Ehelyett az alábbiakban megmutatjuk a vektorvetítés egy konstruktív (tehát a fogalmat egy algoritmus vázlatán keresztül bevezető) definícióját, mely már valódi multivektorok esetén is jól alkalmazható.

### MULTIVEKTOR-VETÍTÉS

Legyenek  $\underline{a}$  és  $\underline{x}$  tetszőleges vektorok, valamint  $a$  és  $x$  ezek alaphalmazai (tehát  $\underline{a}$  és  $\underline{x}$  lehetnek akár valódi multivektorok is).

1. Az  $\underline{a}$  vektor  $x$ -vetülete (az  $\underline{a}$  vektornak az  $x$  halmazra vonatkozó alvektora, vagy másként az  $x$ -relatív alvektora) az

$$\underline{a}|_x$$

módon jelölt vektor, mely az alábbi módon állítható elő:

- 1.1. Határozzuk meg a  $c = a|_x$  halmazt.
- 1.2. Tekintsük a  $c$  halmaz elemeinek valamely elrendezésével létrejött  $\underline{c}$  vektort, és annak  $Ind(\underline{c})$  indexhalmazát.
- 1.3. Hozzunk létre egy  $i : Ind(\underline{c}) \rightarrow Ind(\underline{a})$  indextranszformációt (függvényt), ahol
  - 1.3.1. minden  $k, l \in Ind(\underline{c})$  esetén,  
ha  $k \neq l$ , akkor  $i(k) \neq i(l)$ , továbbá
  - 1.3.2. minden  $k \in Ind(\underline{c})$  esetén,  
 $\underline{c}(k) = \underline{a}(i(k))$ .
- 1.4. Ekkor minden  $k \in Ind(\underline{c})$  esetén,  
 $\underline{a}|_x(i(k)) = \underline{c}(k)$ , és  
 $Ind(\underline{a}|_x) = \{i(k) \mid k \in Ind(\underline{c})\}$ .
2. Az  $\underline{a}$  vektor  $\underline{x}$ -vetülete (az  $\underline{a}$  vektornak az  $\underline{x}$  vektorra vonatkozó alvektora, vagyis az  $\underline{x}$ -relatív alvektora)  
 $\underline{a}|_{\underline{x}} \triangleq \underline{x}|_a$ .
3. Végül a fentiek alapján egy halmaznak is értelmezzük valamely vektorra vonatkozó rendezett részhalmazát, vagyis alvektorát mégpedig az alábbi módon:  
Az  $a$  halmaz  $\underline{x}$ -vetülete (az  $a$  halmaznak az  $\underline{x}$ -relatív alvektora)  
 $a|_{\underline{x}} \triangleq \underline{x}|_a$ .

A fenti  $\underline{a}|_x$ ,  $\underline{a}|_{\underline{x}}$  és  $a|_{\underline{x}}$  esetekben az  $\underline{a}$ -t a vektorvetítés *vetített vektorának*,  $a$ -t a *vetített halmazának*,  $\underline{x}$ -t a *vetítővektorának*, és  $x$ -et a *vetítőhalmazának* is nevezzük. E vektorvetítéseket esetenként *egyszerű vektorvetítéseknek* is fogjuk nevezni (különösen az összetett vektorokra

való alkalmazás esetén). A bevezetett fogalmak nyilvánvaló következményeit fogalmazzák meg az alábbi állítások.

### Állítás

Tetszőleges  $\underline{a}$  vektor esetén

1.  $\underline{a} \upharpoonright_a = \underline{a}$ , és
2.  $\underline{a} \upharpoonright_\emptyset = \underline{a} \upharpoonright_\emptyset = \underline{a} \upharpoonright_\emptyset = \emptyset$ .

### Állítás

Tetszőleges  $\underline{a}$  és  $\underline{x}$  vektorok esetén

1.  $\underline{a} \upharpoonright_x \subseteq \underline{a}$ ,  $\underline{a} \upharpoonright_x \subseteq \underline{x}$ , és  $\underline{a} \upharpoonright_x \subseteq \underline{x}$ , továbbá
2.  $\text{Comp}(\underline{a} \upharpoonright_x) = \text{Comp}(\underline{a} \upharpoonright_x) = \text{Comp}(\underline{a} \upharpoonright_x) = a|_x$ .

### Megjegyzés

A vektorvetítés fenti esetei különböző feladatokban hasznosak. A halmazzal történő vektorvetítés révén lehet egy halmaz elemeivel kijelölni a megtartandó komponenseket. Vektorral történő vektorvetítés esetén a megtartandó komponensek sorrendjét a komponens-kijelölő vektor (a vetítővektor) határozza meg.

Megjegyezzük, hogy a vektorvetítés elvégzése általános esetben nagyon számításgényes feladat. A mintaillesztési feladatok családjába tartozik, és az elvégzéséhez szükséges elemi lépések száma bizonyíthatóan exponenciálisan nő a feldolgozott vektor méretével. A mintaillesztéssel kapcsolatos algoritmuselméleti problémákat azáltal kerültük el, hogy magát a mintaillesztést jelentő indexhalmaz-hozzárendelést és indexleképezést (az indextranszformációt) csak a tulajdonságain keresztül határoztuk meg.

Az alábbi példában lépésről-lépésre követjük a fenti definíció konstrukciós lépéseit. Látni fogjuk, hogy heurisztikus lesz az indexkiosztás és az indexleképezés. Fontos azonban, hogy az esetlegesség csupán látszólagos, ugyanis a lehetséges változatok bármelyikének választása ugyanahhoz az eredményhez vezet.

### Példa

Tekintsük az előbbi példát, tehát legyen  $\underline{a} = \langle 'A', 'L', 'M', 'A' \rangle$  és  $\underline{x} = \langle 'F', 'A', 'K', 'A', 'L', 'A', 'P' \rangle$ . Hozzuk létre a  $\underline{b} = \underline{a} \upharpoonright_x$  vetületvektort a fenti definíció 1. pontjában leírt konstrukció szerint (ahol  $x$  az  $\underline{x}$  vektor alaphalmaza).

1. Határozzuk meg a  $c = a|_x$  halmazt:  

$$c = a|_x = \{ 'A', 'A', 'L' \}.$$
2. Ennek alapján hozzunk létre egy  $\underline{c}$  vektort, és rendeljünk hozzá egy indexhalmazt:  

$$\underline{c} = \langle 'A', 'A', 'L' \rangle, \text{ és legyen ekkor } \text{Ind}(\underline{c}) = I_3.$$
3. Hozzuk létre a definíció-szerinti  $i : \text{Ind}(\underline{c}) \rightarrow \text{Ind}(\underline{a})$  indextranszformációt:  
 Legyen  $\text{Ind}(\underline{a}) = I_4$ , és  $i = \{1 \mapsto 1, 2 \mapsto 4, 3 \mapsto 2\}$ , melyben  $k \mapsto l$  az  $i$  indextranszformáció (függvény) egy elemi függvényleképezése, ahol  $k \in \text{Ind}(\underline{c})$ ,  $l \in \text{Ind}(\underline{a})$ .  
 (Másképpen is definiálhattuk volna az  $i$ -t, például  $i = \{1 \mapsto 4, 2 \mapsto 1, 3 \mapsto 2\}$  módon.)

4. Állítsuk elő ezek után a  $\underline{b} = \underline{a} \upharpoonright_x$  vetületvektort:

Ekkor minden  $k \in \text{Ind}(\underline{c})$  indexhez elő kell állítanunk a  $\underline{b}$  vektor egyes komponenseit a  $\underline{b}(i(k)) = \underline{c}(k)$  kapcsolat alapján, azaz

$$k = 1 \text{ esetén } \underline{b}(i(k)) = \underline{b}(i(1)) = \underline{b}(1) = \underline{c}(k) = \underline{c}(1) = 'A',$$

$$k = 2 \text{ esetén } \underline{b}(i(k)) = \underline{b}(i(2)) = \underline{b}(4) = \underline{c}(k) = \underline{c}(2) = 'A',$$

$$k = 3 \text{ esetén } \underline{b}(i(k)) = \underline{b}(i(3)) = \underline{b}(2) = \underline{c}(k) = \underline{c}(3) = 'L'.$$

Tehát  $\underline{b} = \langle \underline{b}(1), \underline{b}(2), \underline{b}(4) \rangle = \langle 'A', 'L', 'A' \rangle$ . (Ugyanerre az eredményre jutottunk volna, ha az  $i = \{1 \mapsto 4, 2 \mapsto 1, 3 \mapsto 2\}$  indextranszformációt választottuk volna, vagy, ha a  $c = a|_x$  halmazhoz más elrendezésű  $\underline{c}$  vektort választottunk volna.)

#### Állítás

Ha  $\underline{a}$  és  $\underline{x}$  monovektorok, akkor a vektorvetület (a definícióból következően) egyszerűbben is meghatározható az alábbi módon:

Minden  $i \in \text{Ind}(\underline{a})$  esetén, ha  $\underline{a}(i) \in x$ , akkor

$$i \in \text{Ind}(\underline{a} \upharpoonright_x), \text{ és}$$

$$\underline{a} \upharpoonright_x(i) = \underline{a}(i),$$

ahol  $x$  az  $\underline{x}$  vektor alaphalmaza.

#### Példa

Tekintsünk ezúttal monovektorokat. Legyen  $\underline{a} = \langle 3, 5, 4, 7, 1 \rangle$ ,  $b = \{3, 5, 4, 7, 1\}$ , és  $\underline{x} = \langle 7, 2, 4, 5, 3 \rangle$ . Ekkor nyilván  $\underline{a} \upharpoonright_x = \langle 3, 5, 4, 7 \rangle$ , és  $\underline{a} \upharpoonright_{\underline{x}} = b \upharpoonright_{\underline{x}} = \langle 7, 4, 5, 3 \rangle$ .

### Vektorláncolás

Valamely  $\underline{a}$  és  $\underline{b}$  vektorok *láncolata* egy  $\underline{c}$  vektor, jelölésben

$$\underline{c} = \underline{a} + \underline{b},$$

ha

$$1. \mu(\underline{c}) = \mu(\underline{a}) + \mu(\underline{b}), \text{ és}$$

$$2. \text{ minden } i \in I_{\mu(\underline{c})} \text{ sorszámindex esetén}$$

$$\underline{c}[i] = \underline{a}[i], \text{ ha } i \leq \mu(\underline{a}), \text{ és}$$

$$\underline{c}[i] = \underline{b}[i - \mu(\underline{a})] \text{ egyébként.}$$

#### Megjegyzés

A definíció nem írja elő, hogy valamely  $\underline{a}$  és  $\underline{b}$  vektorok esetén mi legyen az  $\underline{a} + \underline{b}$  vektor indexhalmaza, de a gyakorlatban általában az  $I_{\mu(\underline{a}) + \mu(\underline{b})}$  halmazt szoktuk választani.

#### Állítás

A láncolás definíciójából nyilvánvalóan következik

$$1. \underline{a} + \underline{\emptyset} = \underline{\emptyset} + \underline{a} = \underline{a},$$

$$2. \underline{a} \subseteq \underline{a} + \underline{b}, \text{ és } \underline{b} \subseteq \underline{a} + \underline{b}, \text{ továbbá}$$

$$3. (\underline{a} + \underline{b}) \upharpoonright_{\underline{a}} = \underline{a}, \text{ és } (\underline{a} + \underline{b}) \upharpoonright_{\underline{b}} = \underline{b}.$$

**Példa**

$$\langle 'F', 'A' \rangle + \langle 'K', 'A', 'L', 'A', 'P' \rangle = \langle 'F', 'A', 'K', 'A', 'L', 'A', 'P' \rangle.$$

**Vektorok pontozott láncolása**

Esetenként szükségünk lesz arra, hogy vektorok láncolatában meg tudjuk különböztetni az egyes komponenseket aszerint, hogy melyik láncból származnak. Ekkor használjuk a vektorok pontozott láncolását, melynek során minden komponenst a halmazok pontozott egyesítésénél bemutatott módon megjelölünk azzal a vektorral, amelyből származik.

Tehát valamely  $\underline{a}$  és  $\underline{b}$  vektorok *pontozott láncolata* egy  $\underline{c}$  vektor, jelölésben

$$\underline{c} = \underline{a} \dot{+} \underline{b},$$

ha

1.  $\mu(\underline{c}) = \mu(\underline{a}) + \mu(\underline{b})$ , és
2. minden  $i \in I_{\mu(\underline{c})}$  sorszámindex esetén

$$\underline{c}[i] = \underline{a}[\underline{a}[i]], \text{ ha } i \leq \mu(\underline{a}), \text{ és}$$

$$\underline{c}[i] = \underline{b}[\underline{b}[i - \mu(\underline{a})]] \text{ egyébként.}$$

**Példa**

Legyen  $\underline{a} = \langle 1, 2, 3 \rangle$ ,  $\underline{b} = \langle 3, 4 \rangle$ . Ekkor  $\underline{a} \dot{+} \underline{b} = \langle \underline{a}.1, \underline{a}.2, \underline{a}.3, \underline{b}.3, \underline{b}.4 \rangle$ .

**Vektorkivonás**

Valamely  $\underline{a}$  és  $\underline{b}$  vektorok *különbsége*

$$\underline{a} - \underline{b} \triangleq (\underline{a} \setminus \underline{b}) \upharpoonright_{\underline{a}},$$

ahol  $\underline{a}$ , illetve  $\underline{b}$  az  $\underline{a}$ , illetve  $\underline{b}$  vektorok alaphalmazai. Nyilván  $\mu(\underline{a} - \underline{b}) = \mu(\underline{a} \setminus \underline{b})$ .

Megjegyezzük, hogy a vektorvetítés definíciója szerint a vektorkivonás fogalmának fenti bevezetése egyenértékű módon átírható az

$$\underline{a} - \underline{b} \triangleq \underline{a} \upharpoonright_{(\underline{a} \setminus \underline{b})},$$

alakba is.

**Példa**

1.  $\langle 'E', 'G', 'E', 'R' \rangle - \langle 'R', 'E', 'G', 'E' \rangle = \underline{\emptyset}$ .
2.  $\langle 'E', 'G', 'E', 'R' \rangle - \langle 'E', 'G', 'E', 'R' \rangle = \underline{\emptyset}$ .
3.  $\langle 'K', 'E', 'R', 'É', 'K' \rangle - \langle 'P', 'É', 'K' \rangle = \langle 'K', 'E', 'R' \rangle$ .

**Állítás**

Valamely  $\underline{a}$ ,  $\underline{b}$  és  $\underline{c}$  vektorok esetén

1.  $\underline{a} \upharpoonright_{(\underline{b} \setminus \underline{c})} = \underline{a} \upharpoonright_{\underline{b}} - \underline{a} \upharpoonright_{\underline{c}}$ .
2.  $\underline{a} \upharpoonright_{(\underline{a} \setminus \underline{b})} = \underline{a} - \underline{a} \upharpoonright_{\underline{b}}$ .
3.  $\underline{a} - \underline{b} = \underline{a} - \underline{a} \upharpoonright_{\underline{b}}$ .

**Bizonyítás**

1. A bizonyítás alapgondolata az, hogy szétválasztjuk a vektorokra vonatkozó azonosságokban az alaphalmazokra, valamint a bennük szereplő elemek rendezettségére vonatkozó állításokat.

Mivel a vektorvetítésre korábban bemutatott állítás szerint egy vektor vektorvetületének alaphalmaza a vektor alaphalmazának halmazvetülete, így

$$\text{Comp}(\underline{a}|_{(b \setminus c)}) = \underline{a}|_{(b \setminus c)}. \quad (1)$$

A halmazkivonásnál kimondott állítás szerint

$$\underline{a}|_{(b \setminus c)} = \underline{a}|_b \setminus \underline{a}|_c. \quad (2)$$

Mivel az 1. állítás mindkét oldalán szereplő vektorok rendezettségét az  $\underline{a}$  vektor rendezettsége határozza meg, így az (1) és (2) állításokból már következik az 1. állítás. ■

2. Ez az állítás az előzőből, valamint az

$$\underline{a}|_a = \underline{a} \quad (3)$$

állításból következik. ■

3. Ez az állítás az 1. és 2. állításokból, valamint a vektorkivonás definíciójából közvetlenül következik. ■

**Egyesítő láncolás**

Valamely  $\underline{a}$  és  $\underline{b}$  vektorok *egyesítő láncolata*

$$\underline{a} \cup \underline{b} \triangleq \underline{a} + (\underline{b} - \underline{a}).$$

**Példa**

1.  $\langle 'K', 'E', 'R', 'É', 'K' \rangle \cup \langle 'P', 'É', 'K' \rangle =$   
 $= \langle 'K', 'E', 'R', 'É', 'K' \rangle + (\langle 'P', 'É', 'K' \rangle - \langle 'K', 'E', 'R', 'É', 'K' \rangle) =$   
 $= \langle 'K', 'E', 'R', 'É', 'K' \rangle + \langle 'P' \rangle = \langle 'K', 'E', 'R', 'É', 'K', 'P' \rangle.$
2.  $\langle 'P', 'É', 'K' \rangle \cup \langle 'K', 'E', 'R', 'É', 'K' \rangle =$   
 $= \langle 'P', 'É', 'K' \rangle + (\langle 'K', 'E', 'R', 'É', 'K' \rangle - \langle 'P', 'É', 'K' \rangle) =$   
 $= \langle 'P', 'É', 'K' \rangle + \langle 'K', 'E', 'R' \rangle = \langle 'P', 'É', 'K', 'K', 'E', 'R' \rangle.$
3.  $\langle 'E', 'G', 'É', 'R' \rangle \cup \langle 'É', 'G', 'E', 'R' \rangle =$   
 $= \langle 'E', 'G', 'É', 'R' \rangle + (\langle 'É', 'G', 'E', 'R' \rangle - \langle 'E', 'G', 'É', 'R' \rangle) =$   
 $= \langle 'E', 'G', 'É', 'R' \rangle + \emptyset = \langle 'E', 'G', 'É', 'R' \rangle.$

**Állítás**

Valamely  $\underline{a}$  és  $\underline{b}$  vektorok, valamint  $\underline{c} = \underline{a} \cup \underline{b}$  esetén nyilván teljesülnek az alábbiak:

1.  $\underline{c} = \underline{a} \cup (\underline{b} \setminus \underline{a}),$
2.  $\underline{c}|_a = \underline{a},$
3.  $\underline{c}|_b = \underline{b}.$

4. ha  $\underline{a}$  és  $\underline{b}$  monovektorok, akkor  $\underline{c}$  is monovektor, és  $c = a \cup b$ .

Megjegyezzük, hogy a  $\cup$  műveletet a 4. állításbeli tulajdonsága miatt nevezzük egyesítő láncolásnak. (A fentiekben  $a$ ,  $b$ , illetve  $c$  nyilván az  $\underline{a}$ ,  $\underline{b}$ , illetve  $\underline{c}$  vektorok alaphalmazai.)

## Halmazrendszerek

### Egyszerű és összetett elemek

Az eddigiekben akár halmazról, akár vektorról volt szó, az elemeivel nem törődtünk, azokat valami oszthatatlan dolognak képzeltük, valami olyannak, mint a régi görögök az atomot.

A továbbiakban látni fogjuk, hogy lesznek olyan objektumaink (halmazok, vektorok), melyeknek az elemei halmazok, vektorok, relációk, vagy függvények, stb. Ezeket az objektumokat sokszor ezután is csak a hagyományos algebrai eszközökkel vizsgáljuk, és ilyenkor természetesen nem lesz érdekes, hogy elemeiknek van-e valamilyen belső szerkezete, vagy nincs. A tárgyalásunk fő vonalába tartozó relációk, függőségek esetén azonban általában nagyon is érdekel bennünket az összetett objektumok viselkedése, egymással való kapcsolata. Ennek érdekében bevezetjük az egyszerű és az összetett elemek fogalmát.

Ezekután tehát *egyszerű elemeknek* nevezzük az olyan elemeit valamely halmaznak, vektornak, amelyek az adott tárgyalásban semmilyen részlettel, résztulajdonsággal nem rendelkeznek, és *összetett elemeknek* azokat, melyeknek felépítéséről, résztulajdonságairól valamilyen leírást adunk, ez utóbbiakra vonatkozóan esetleg valamilyen kikötést, megszorítást teszünk. Hangsúlyozzuk, hogy egy elem ilyen módon való megítélése (tehát, hogy egyszerűnek, vagy összetettnek tekintjük) erősen függ attól a környezettől, melyben tárgyaljuk. Végül, ha egy halmaz, illetve vektor csak egyszerű elemeket tartalmaz, akkor *egyszerű halmaznak*, illetve *egyszerű vektornak*, egyébként pedig *összetett halmaznak*, illetve *összetett vektornak* nevezzük. Az összetett halmazokat és vektorokat röviden csak *halmazrendszereknek* nevezzük. Megjegyezzük, hogy a szakirodalomban gyakran *hiper* előtaggal jelzik az olyan halmazokat, algebrai objektumokat, melyeknek elemei összetett szerkezettel rendelkeznek. Így beszélnek hiperhalmazokról, hiperterekről, stb.

### Halmazrendszerek típusai

A halmazrendszereket – a gyakorlati igényekkel összhangban – négy csoportba soroljuk:

- *Hiperhalmazoknak* nevezzük az olyan halmazokat, melyeknek minden eleme halmaz. A továbbiakban csak az úgynevezett *mono-hiperhalmazokkal* fogunk foglalkozni, vagyis az olyanokkal, melyeknek minden elemi halmaza monohalmaz.
- *Halmazvektoroknak* nevezzük az olyan vektorokat, melyeknek minden komponense halmaz. Ezek közül is csak az úgynevezett *mono-halmazvektorokkal* foglalkozunk, tehát azokkal, melyeknek minden halmaza monohalmaz.
- *Vektorhalmazoknak* az olyan halmazokat nevezzük, melyeknek minden eleme vektor. Ezek közül csak az úgynevezett *rektanguláris vektorhalmazokkal* foglalkozunk, vagyis csak azokkal, melyeknek minden elemi vektora azonos hosszúságú.
- *Hipervektoroknak* az olyan vektorokat nevezzük, melyeknek minden eleme vektor. Vizsgálataink szempontjából ezek közül csak az úgynevezett *rektanguláris multi-*

*hipervektorok* érdekesek, vagyis csak azok, melyeknek minden elemi vektora azonos hosszúságú, de azonos elemeket (azonos vektorokat) már tartalmazhatnak, és maguk a vektorok is tartalmazhatnak azonos komponenseket. A relációs adatbázisok alapvető objektumait alkotó, úgynevezett adattáblát a hipervektor modellelzi a leghitelesebben, ám tárgyalásunk korlátozott jellege miatt ezekkel nem foglalkozunk.

A halmazrendszerek közül tehát az első kettő tulajdonképpen speciális halmaz, a második kettő pedig speciális vektor (bár algebrailag már maguk a vektorok is speciális halmazoknak tekinthetők). A továbbiakban, amikor ezekkel az összetett objektumokkal foglalkozunk mindig fennáll annak a lehetősége, hogy eltekintsünk összetett jellegüktől, és egyszerű halmaznak, vagy vektornak tekintve őket, az azokon bevezetett műveleteket használjuk. Természetesen az igazán érdekes műveletek, transzformációk éppen azok lesznek, melyeknél az összetett jellegüket kihasználjuk.

## Hiperhalmaz

Az olyan halmazt, melynek minden eleme halmaz, *hiperhalmaznak* nevezzük. Valamely  $A = \{A_1, \dots, A_n\}$  hiperhalmazt *mono-hiperhalmaznak* nevezünk, ha minden  $A_i \in A$  egy monohalmaz. Tehát egy hiperhalmazban az egyes elemi halmazok lehetnek azonosak, csak azoknak már nem lehetnek (egy halmazon belül) azonos elemeik. A továbbiakban hiperhalmazon mindig mono-hiperhalmazt értünk. Egy

$A = \{A_1, \dots, A_n\}$  hiperhalmazt esetenként

$A = \{A_i \mid i \in I_n\}$  módon, vagy még általánosabban

$A = \{A_i \mid i \in N\}$  módon is fogunk jelölni, ahol  $N \subset \mathbf{N}$  egy véges halmaz.

## Hiperhalmaz metszete és egyesítése

Ez természetes általánosítása két (mono)halmaz metszetének, illetve egyesítésének, azaz valamely  $\{A_i \mid i \in N\}$  hiperhalmaz metszete, illetve egyesítése (monouniója):

$$\bigcap_{i \in N} A_i \triangleq \{x \mid \forall i \in N : x \in A_i\}, \text{ illetve } \bigcup_{i \in N} A_i \triangleq \{x \mid \exists i \in N : x \in A_i\}.$$

### Példa

Legyen  $A = \{A_1, A_5, A_6, A_7\}$  egy hiperhalmaz, ahol  $A_1 = \{1, 2, 3, 4\}$ ,  $A_5 = \{2, 3\}$ ,  $A_6 = \{3, 6, 9\}$ ,  $A_7 = \{2, 3\}$  és  $N = \{1, 5, 6, 7\}$ . Ekkor

$$\bigcap_{i \in N} A_i = \{3\}, \text{ és } \bigcup_{i \in N} A_i = \{1, 2, 3, 4, 6, 9\}.$$

## Halmazvektor

Egy  $A = \{A_1, \dots, A_n\}$  hiperhalmaz elemeinek valamilyen kötött sorrendű elrendezését az  $A$  *hiperhalmaz egy halmazvektorának* nevezzük és  $\underline{A}$  vektorként jelöljük. A továbbiakban fogunk egy  $\underline{A}$  *halmazvektor alaphalmazáról*, vagy *hiperhalmazáról* is beszélni, amely alatt pedig egy olyan  $A$  hiperhalmazt értünk melynek elemi halmazai az  $\underline{A}$  halmazvektor komponenseit alkotó halmazok.

Az  $\underline{A}$  halmazvektor  $i$ -edik komponensét ( $i \in I_n$ ) alkotó halmazt  $\underline{A}(i)$  módon is fogjuk jelölni. Ekkor tehát az  $\underline{A}$  halmazvektort felírhatjuk

$$\underline{A} = \langle \underline{A}(1), \dots, \underline{A}(n) \rangle$$

alakban is, ahol minden  $i \in I_n$  esetén  $\underline{A}(i) \in A$ , de használni fogjuk az

$$A_i \triangleq \underline{A}(i)$$

összerendelés alapján az

$$\underline{A} = \langle A_1, \dots, A_n \rangle \text{ jelölést is.}$$

A gyakorlatban ez utóbbi jelölést használjuk inkább, elsősorban az egyszerűsége miatt, azonban ezúttal is vegyük figyelembe, hogy az indexek az elemek megkülönböztetésére és a rendezés jelzésére szolgálnak csupán.

Egy  $\underline{A} = \langle A_1, \dots, A_n \rangle$  halmazvektort *mono-halmazvektornak* nevezzük, ha minden  $A_i \in A$  egy monohalmaz. Tehát egy mono-halmazvektorban az egyes komponensek (elemi halmazok) lehetnek azonosak, csak a komponenseknek (mint halmazoknak) nem lehetnek azonos elemeik. A továbbiakban halmazvektoron mindig mono-halmazvektort értünk.

## Halmazvektor egyszerű vektorvetítése, alvektora

Az egyszerű vektorokra bevezetett vektorvetítést akkor alkalmazzuk halmazvektorra, ha nem kívánjuk figyelembe venni a komponensek összetett jellegét. Természetesen ekkor használhatjuk a bevezetett halmaz-relatív, vagy vektor-relatív vektorvetítés bármelyikét, és az alvektor fogalom is megegyezik az egyszerű vektoroknál bevezetettel.

### Példa

Legyenek  $\underline{A}$  és  $\underline{B}$  halmazvektorok, ahol  $\underline{A} = \langle A_1, A_2, A_3 \rangle$ ,  $\underline{B} = \langle A_4, A_3, A_2 \rangle$ ,  $\underline{C} = \langle A_2, A_3 \rangle$ , és  $\underline{D} = \langle A_3, A_2 \rangle$ . Ekkor nyilván  $\underline{A} \upharpoonright_B = \underline{C}$  és  $\underline{C} \subseteq \underline{A}$ , valamint  $\underline{A} \upharpoonright_B = \underline{D}$  és  $\underline{D} \subseteq \underline{B}$ , továbbá természetesen  $\underline{C}, \underline{D} \subseteq A$ , ahol  $\underline{C}, \underline{D}$  és  $A$  a  $\underline{C}, \underline{D}$  és  $\underline{A}$  halmazvektorok alaphalmazai, azaz  $\underline{C} = \underline{D} = \{A_2, A_3\}$  és  $A = \{A_1, A_2, A_3\}$ .

## Halmazvektor-vetítés

Tekintsünk valamely  $\underline{A}$  és  $\underline{B}$  halmazvektorokat, melyekre  $\mu(\underline{B}) = \mu(\underline{A})$ . Ekkor az  $\underline{A}$  halmazvektor  $\underline{B}$ -halmazvektor-vetülete egy

$$\underline{C} = \underline{A} \upharpoonright_B$$

halmazvektor, ahol

1.  $\mu(\underline{C}) = \mu(\underline{A})$ , és
2. minden  $i \in \text{Ind}(\underline{A})$  és  $C_i = \underline{C}(i)$  esetén  $C_i = A_i|_{B_i}$ , ahol  $A_i = \underline{A}(i)$  és  $B_i = \underline{B}(i)$ .

Ekkor  $\underline{A}$ -t e vetítés *vetítettvektorának*, a  $\underline{B}$ -t pedig e vetítés *vetítővektorának* is nevezzük.

### Megjegyzés

Figyeljünk fel arra a lényeges különbségre, amely a halmazvektor-vetítés és a vektorvetítés között van. A vektorvetítés esetén nem törődünk azzal, hogy milyen elemekből áll a vektor (pontosabban az elemeket egyszerűnek tételezzük fel), amely mögött egy hagyományos halmazművelet áll (például a  $\underline{C} = \underline{A} \upharpoonright_B$  komponensei az  $A|_B$ , vagy más jelöléssel az  $A \cap B$  elemei). Ezzel szemben a halmazvektor-vetítés esetén e hagyományos halmazművelet

a vetített és a vetítő halmazvektorok komponenseit alkotó halmazok között áll fenn. Tehát míg a vektorvetítés esetén kapott vektor alvektora a vetített vektornak, és ennek megfelelően általában rövidebb, addig a halmazvektor-vetítés eredményeként kapott (halmaz)vektor hossza megegyezik a vetített (halmaz)vektor hosszával, a halmazvektor-vetítés a komponensek számát nem változtatja meg. Azt is mondhatjuk tehát, hogy míg a vektorvetítés nem minden komponenszt „visz át”, de amit átvisz azt teljes egészében (nyilván akkor is, ha a vektor elemei például halmazok), addig a halmazvektor-vetítés minden komponenszt „átvisz”, de az egyes komponenseket (azaz komponenshalmazokat) csak részlegesen.

### Példa

Legyenek  $\underline{A}$  és  $\underline{B}$  halmazvektorok, ahol  $\underline{A} = \langle A_1, A_2, A_3 \rangle$ , melyben  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x\}$ , továbbá  $\underline{B} = \langle B_1, B_2, B_3 \rangle$ , melyben  $B_1 = \{3, 4, 5\}$ ,  $B_2 = \{b\}$ ,  $B_3 = \{x\}$ , valamint  $\underline{C} = \langle B_2, B_1, B_3 \rangle$  és  $\underline{D} = \langle B_1, B_2 \rangle$ . Ekkor nyilván  $\underline{A} \parallel \underline{B} = \langle \{3, 4\}, \{b\}, \{x\} \rangle$ , és  $\underline{A} \parallel \underline{C} = \langle \emptyset, \emptyset, \{x\} \rangle$ . Az  $\underline{A} \parallel \underline{D}$  nem végezhető el, mivel  $\mu(\underline{D}) \neq \mu(\underline{A})$ .

## Halmazvektor résztartománya

Legyen  $\underline{A}$  egy halmazvektor. Egy  $\underline{C}$  halmazvektor *résztartománya* az  $\underline{A}$  halmazvektornak, azaz

$$\underline{C} \subseteq^* \underline{A},$$

ha

1.  $\mu(\underline{C}) = \mu(\underline{A})$ ,
2. minden  $C_i \in \underline{C}$  esetén  $C_i \subseteq A_i$ , ahol  $A_i \in \underline{A}$ .

Nyilván tetszőleges  $\underline{A}$  és  $\underline{B}$  halmazvektorok esetén  $\underline{A} \parallel \underline{B} \subseteq^* \underline{A}$ , és  $\underline{A} \subseteq^* \underline{A}$ , továbbá  $\underline{B} \subseteq \underline{A}$  esetén  $\underline{B} \subseteq^* \underline{A}$  is teljesül.

### Példa

Legyen  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x\}$ ,  $A_4 = \{x\}$ ,  $A_5 = \{2, 3\}$ ,  $A_6 = \{3, 4, 5\}$ ,  $A_7 = \{b\}$ , továbbá  $\underline{A} = \langle A_1, A_2, A_3 \rangle$ ,  $\underline{B} = \langle A_5, A_7, A_4 \rangle$ ,  $\underline{C} = \langle A_1, A_2 \rangle$ ,  $\underline{D} = \langle A_6, A_7, A_4 \rangle$ ,  $\underline{E} = \langle A_5, A_7 \rangle$ , és  $\underline{F} = \langle A_2, A_3, A_1 \rangle$ . Ekkor nyilván  $\underline{C} \subseteq \underline{A}$ ,  $\underline{E} \subseteq \underline{A}$ ,  $\underline{C} \subseteq^* \underline{A}$ ,  $\underline{E} \subseteq^* \underline{A}$ ,  $\underline{B} \subseteq^* \underline{A}$ , és természetesen  $\underline{A} \subseteq \underline{A}$ , és  $\underline{A} \subseteq^* \underline{A}$ . Nem teljesül azonban  $\underline{B} \subseteq \underline{A}$ , mivel  $A_5 \neq A_1$  és  $A_7 \neq A_2$ , nem teljesül  $\underline{D} \subseteq^* \underline{A}$ , mivel  $A_6$  nem részhalmaza  $A_1$ -nek, nem teljesül  $\underline{E} \subseteq \underline{A}$ , mivel túl kevés komponense van, végül pedig  $\underline{F} \subseteq^* \underline{A}$  sem, mivel a komponensek sorrendje nem megfelelő.

## Vektorhalmaz

Egy  $A$  halmazt *vektorhalmaznak* nevezünk, ha minden eleme vektor. Egy  $A = \{\underline{A}_1, \dots, \underline{A}_n\}$  vektorhalmazt *rektangulárisnak* nevezzük, ha minden  $\underline{A}_i, \underline{A}_j \in A$  esetén  $\mu(\underline{A}_i) = \mu(\underline{A}_j)$ . A továbbiakban vektorhalmaz alatt mindig rektanguláris vektorhalmazt értünk.

Egy  $A$  vektorhalmaz *mono-vektorhalmaz*, ha minden minden  $\underline{A}_i, \underline{A}_j \in A$ ,  $i \neq j$  esetén  $\underline{A}_i \neq \underline{A}_j$ , egyébként pedig *multi-vektorhalmaz*. Megjegyezzük, hogy e definíciók megengedik, hogy egy vektorban azonos komponensek lehessenek.

## Komponenshalmaz-függvény kiterjesztése vektorhalmazra

Egy  $A$  vektorhalmaz komponenshalmaz-függvénye

$$\text{Comp}(A) \triangleq \bigsqcup_{A_i \in A} \text{Comp}(A_i),$$

ahol  $A_i$  az  $A$  vektor alaphalmaza. A komponenshalmaz-függvény segítségével egyetlen halmazba egyesíthetjük egy vektorhalmaz vektorainak komponenseit.

## Vektorhalmaz egyszerű halmazvetítése, részhalmaza

Az egyszerű halmazokra bevezetett halmazvetítést akkor alkalmazzuk vektorhalmazokra, ha nem kívánjuk figyelembe venni az elemek összetett jellegét. Ekkor természetesen a részhalmaz fogalom is megegyezik az egyszerű halmazoknál bevezetettel.

### Példa

Legyenek  $A$  és  $B$  vektorhalmazok, ahol  $A = \{\underline{A}_1, \underline{A}_2, \underline{A}_3\}$ ,  $B = \{\underline{A}_2, \underline{A}_3, \underline{A}_4\}$ , és  $C = \{\underline{A}_2, \underline{A}_3\}$ . Ekkor nyilván  $A|_B = C$ , és  $C \subset A$ .

## Vektorhalmaz-vetítés (bevezető gondolatok)

Az eddigi tárgyalás logikájának megfelelően most jött el az ideje annak, hogy definiáljuk a vektorhalmazok olyan vetítését, mely figyelembe veszi a vektorhalmazok elemeinek összetett jellegét, vagyis azt, hogy azok vektorok.

A vektorhalmazok számunkra a legfontosabb összetett objektumot, halmazrendszert képviselik, mivel a tárgyalásunk alapját képező adattáblákat is vektorhalmazként (azaz relációként) fogjuk tekinteni. (A korábban említettek szerint a hipervektorként történő definiálásnak ugyan lényeges előnyei lennének, ám ezzel igen eltérnénk az adatbázisok algebrájának hagyományos tárgyalásától).

A vektorok fogalmának bevezetésénél már említettük, hogy a vektor pontos értelmezését majd a Descartes-szorzás fogja adni. A Descartes-szorzat – melyet az alábbiakban definiálunk – egy különleges rektanguláris vektorhalmaz. Különlegessége elsősorban abban rejlik, hogy az elemeit alkotó vektorok komponenseinek megadja a „származtatását”, vagyis azokat a halmazokat, amelyekből a vektorkomponensek származnak. Ez jelentősen egyszerűsíti egyrészt a vektorok egyes komponenseinek azonosítását, indexelését, másrészt az olyan vektorhalmaz-transzformációk értelmezését, mint például a relációalgebrában szokásosan bevezetett, úgynevezett relációműveletek, és ezek egyikeként éppen a vektorhalmaz-vetítés.

A vektorhalmaz fogalmának bevezetésénél már említettük, hogy a vektorhalmazok közül csak a rektangulárisokkal fogunk foglalkozni. Mivel ezen túlmenően elmondhatjuk, hogy tárgyalásunk szempontjából elegendő az olyan vektorhalmazokkal foglalkoznunk, melyek a Descartes-szorzat fogalmából származtathatók, ezért a vektorhalmaz-vetítés fogalmának bevezetését, most megszakítjuk annak érdekében, hogy bemutathassuk a Descartes-szorzást, és majd ennek segítségével térhessünk vissza a vektorhalmaz-vetítésre.

## Halmazok Descartes-szorzata, rekord

Valamely  $X$  és  $Y$  halmazok *Descartes-szorzata*

$$X \times Y \triangleq \{ \langle x, y \rangle \mid x \in X, y \in Y \}$$

vektorokból álló halmaz. Például, ha  $X = \{ a, b \}$  és  $Y = \{ 1, 2, 3 \}$ , akkor  $X \times Y = \{ \langle a, 1 \rangle, \langle a, 2 \rangle, \langle a, 3 \rangle, \langle b, 1 \rangle, \langle b, 2 \rangle, \langle b, 3 \rangle \}$ .

Definíciószerűen

$$\emptyset \times \emptyset \triangleq \emptyset,$$

és tetszőleges  $X = \{ a_1, \dots, a_n \}$  halmaz esetén

$$X \times \emptyset \triangleq \emptyset \times X \triangleq \{ \langle a_1 \rangle, \dots, \langle a_n \rangle \}.$$

A Descartes-szorzást kiterjesztjük kettőnél több halmazra is. Tekintsük az  $A = \{ A_1, \dots, A_n \}$  hiperhalmazt, és ennek valamilyen kötött sorrendű elrendezéseként az  $\underline{A} = \langle A_1, \dots, A_n \rangle$  halmazvektort. Ekkor az  $\underline{A}$  halmazvektor fölötti *Descartes-szorzat*

$$\times_{i \in \text{Ind}(\underline{A})} A_i = A_1 \times \dots \times A_n, \text{ mivel } \text{Ind}(\underline{A}) = I_n.$$

Szokás az alábbi tömörebb jelölést használni:

$$\times \underline{A} \triangleq \times_{i \in \text{Ind}(\underline{A})} A_i, \text{ és nyilván}$$

$$\times \underline{A} = \{ \langle a_1, \dots, a_n \rangle \mid a_1 \in A_1, \dots, a_n \in A_n, \forall i \in I_n : A_i \in A \},$$

ahol  $A$  az  $\underline{A}$  halmazvektor alaphalmaza.

Egy  $\underline{A}$  halmazvektor fölötti Descartes-szorzatot a továbbiakban  $\mathbf{D}(\underline{A})$  módon jelölünk, és ekkor az  $\underline{A}$  halmazvektort a  $\mathbf{D}(\underline{A})$  Descartes-szorzat *alapvektorának* is nevezzük. Tehát

$$\mathbf{D}(\underline{A}) \triangleq \times \underline{A}.$$

A Descartes-szorzat elemeit *rekordoknak* nevezzük. Valamely  $\underline{A}$  halmazvektor esetén egy  $r \in \mathbf{D}(\underline{A})$  rekord tehát egy olyan vektor, melyre minden  $i \in I_n$  esetén  $r(i) \in A(i)$ . Ennek alapján bevezetjük az

$$r(A_i) \triangleq r(i)$$

jelölést, ahol  $i \in I_n$ . Így a rekordok komponenseit az alapvektor elemi halmazzaival indexezzük, vagyis azokkal a halmazokkal, melyekből e komponensek, mint elemek származnak.

Ha  $\mu(\underline{A}) = n$ , akkor a  $\mathbf{D}(\underline{A})$  elemeit *n-elemű rekordoknak*, vagy *n-eseknek*, ha  $n = 2$ , akkor *párosoknak* is nevezzük.

Nyilván egy  $\mathbf{D}(\underline{A})$  Descartes-szorzat rekordjainak száma

$$\mu(\mathbf{D}(\underline{A})) = \mu(A_1) * \dots * \mu(A_n),$$

ahol minden  $i \in I_n$  esetén  $A_i \in A$ .

A Descartes-szorzat fenti definíciójában az  $\underline{A}$  alapvektor indexhalmaza  $I_n$ . Bár legtöbbször valóban ezt használjuk, azonban előfordul (különösen ha az alapvektor valamilyen származtatás eredménye), hogy az  $\text{Ind}(\underline{A})$  indexhalmaz a természetes számok valamely általánosabb részhalmaza. Így a Descartes-szorzat általános definíciója:

$$\mathbf{D}(\underline{A}) \triangleq \{ \langle a_1, \dots, a_n \rangle \mid \forall k \in I_n, j_k \in \text{Ind}(\underline{A}) : (a_{j_k} \in A_{j_k}, A_{j_k} \in A) \}, \text{ ahol } n = \mu(\underline{A}).$$

A többi bevezetett fogalom definícióját ennek megfelelően kell természetesen átírni.

A továbbiakban – az egyszerűség kedvéért – a Descartes-szorzat rekordjainak komponenseit általában az  $I_n$  segítségével fogjuk indexezni. Ha valahol a későbbiekben mégis az általánosabb indexezésre volna szükség, akkor ott minden további megjegyzés nélkül azt fogjuk használni.

#### Megjegyzés

A továbbiakban egy Descartes-szorzatot kizárólag olyan alapvektoron (halmazvektoron) fogunk használni, mely mono-halmazvektor. (Ez a feltétele ugyanis annak, hogy a Descartes-szorzat monohalmaz legyen, ami pedig a továbbiakban kiemelkedően fontos a számunkra.) A fentiek alapján nyilvánvaló a következő állítás.

#### Állítás

Ha  $\underline{A}$  egy mono-halmazvektor, akkor a  $\mathbf{D}(\underline{A})$  Descartes-szorzat egy rektanguláris mono-vektorhalmaz.

### Az egytényezős Descartes-szorzat

A Descartes-szorzat fogalmának kiterjesztéseként értelmezzük a  $\mathbf{D}(\underline{A}) = \times \underline{A}$  alakot  $\mu(\underline{A}) = 1$  esetben is. (Ez természetesen a Descartes-szorzat definíciójából nem származtatható.) Ekkor a Descartes-szorzat elemei (rekordjai) olyan egyelemű vektorok, melyek az alaphalmaz egy-egy elemét tartalmazzák, azaz tetszőleges  $B$  halmaz és

$$\underline{A} = \langle B \rangle \text{ esetén}$$

$$\mathbf{D}(\underline{A}) \triangleq \emptyset \times B.$$

Ekkor nyilván  $\mu(\mathbf{D}(\underline{A})) = \mu(B)$  és  $\text{Comp}(\mathbf{D}(\underline{A})) = B$ .

Megjegyezzük, hogy az egyelemű rekordokból álló „Descartes-szorzat”-ra azért van szükségünk, mert be fogunk vezetni olyan transzformációkat, melyek eredményeként keletkező rektanguláris vektorhalmazok rekordjaiban csökken a komponensek száma (márpedig a rektanguláris vektorhalmazokat a Descartes-szorzathoz fogjuk algebrailag származtatni – lásd a „Vektorhalmaz (a Descartes-szorzathoz származtatva)” című pontot a későbbiekben).

### Néhány megjegyzés a Descartes-szorzás elvégzéséről

A Descartes-szorzás – a definíciója szerint – egy halmazvektorból egy vektorhalmazt állít elő (az  $\underline{A}$  halmazvektorból a  $\mathbf{D}(\underline{A})$  vektorhalmazt), vagyis egyszerű elemeket tartalmazó halmazokból egy összetett elemeket tartalmazó halmazt hoz létre. E tulajdonsága teszi az algebra legfontosabb transzformációjává, ám épp e tulajdonsága miatt nem tekinthetjük algebrai értelemben műveletnek. Az alábbiakban megmutatjuk, hogy milyen gyakorlati nehézségeket okoz a Descartes-szorzás alkalmazása minden olyan esetben, amikor a műveletszerű használat kényelmes lenne, és azt is, hogy e nehézségeket hogyan tudjuk (némi ügyeskedéssel) megkerülni.

Figyeljünk fel arra, hogy tetszőleges  $A_1, A_2, A_3$  halmazok esetén

$$A_1 \times A_2 \times A_3 \neq (A_1 \times A_2) \times A_3 \neq A_1 \times (A_2 \times A_3),$$

vagyis a Descartes szorzás nem asszociatív.

Mit is jelent az asszociativitás? Tekintsük például a valós számok szorzását. Nyilván  $(2 \cdot 3.5) \cdot \pi = 2 \cdot (3.5 \cdot \pi)$ , amit az egyszerű jelölés érdekében  $2 \cdot 3.5 \cdot \pi$  módon is szoktunk jelölni. Tehát az egyes szorzások elvégzésének sorrendje tetszőleges, vagyis a szorzás asszociatív (átzárójelezhető, csoportosítható). Nem asszociatív viszont a kivonás. Tekintsük például a  $10-5-1$  műveletsorozatot. Ezt csak a szokásos balról-jobbra történő kiértékeléssel végezhethetjük el, azaz  $(10-5)-1$  módon, különben a helyes 4 érték helyett a helytelen 6 értéket kapnánk (természetesen más a helyzet, ha zárójelezéssel már eleve más végrehajtási sorrendet írunk elő.). Nem asszociatív a szorzások közül például a mátrixok szorzása sem, még akkor se, ha a szorzásban csak négyzetes mátrixok vesznek részt.

A fenti nem asszociatív műveleteknek azonban van mégis egy közös tulajdonságuk az asszociatív műveletekkel. Vegyük észre, hogy a fent felsorolt műveleteknél abban ugyan volt különbség, hogy milyen bináris (két operandusú) csoportosításban adnak helyes eredményt, abban viszont egységesek voltak, hogy az esetleges zárójelezéssel módosított, de alapvetően balról-jobbra történő, láncszerű (tehát a bináris részműveletekre lebontott) végrehajtás mindegyiknél helyes eredményt ad. Láncműveleteknek is hívjuk ezért ezeket, hiszen (az esetleges zárójelezésektől eltekintve) a balról-jobbra történő végrehajtás menetében a keletkező eredmény és a sorban (láncban) jobbra következő elem között végezzük el az újabb műveletet. Tulajdonképpen az történik, hogy például egy  $n$ -tényezős álló szorzást visszavezetünk  $(n-1)$  darab bináris szorzásra.

Mi a helyzet ezzel szemben a Descartes-szorzással? Valamely  $A_1 = \{1, 2\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{\alpha, \beta\}$  halmazok esetén a definíció szerinti Descartes-szorzatuk:

$$A_1 \times A_2 \times A_3 = \{\langle 1, a, \alpha \rangle, \langle 1, a, \beta \rangle, \langle 1, b, \alpha \rangle, \langle 1, b, \beta \rangle, \\ \langle 2, a, \alpha \rangle, \langle 2, a, \beta \rangle, \langle 2, b, \alpha \rangle, \langle 2, b, \beta \rangle\}.$$

Ha azonban a Descartes-szorzást láncműveletként végezzük el (balról-jobbra), akkor az alábbi eredményt kapjuk:

$$(A_1 \times A_2) \times A_3 = \{\langle \langle 1, a \rangle, \alpha \rangle, \langle \langle 1, a \rangle, \beta \rangle, \langle \langle 1, b \rangle, \alpha \rangle, \langle \langle 1, b \rangle, \beta \rangle, \\ \langle \langle 2, a \rangle, \alpha \rangle, \langle \langle 2, a \rangle, \beta \rangle, \langle \langle 2, b \rangle, \alpha \rangle, \langle \langle 2, b \rangle, \beta \rangle\},$$

ami láthatóan az előzőtől eltérő, tehát helytelen eredmény. (Ennek belátásához elég arra gondolni, hogy a Descartes-szorzat elemei a rekordok, tehát egy további műveletben azok már nem bonthatók fel.) *A Descartes-szorzás tehát nem asszociatív, és láncszerűen nem elvégezhető művelet.* (Ebből egyébként már következik, hogy nem asszociatív.)

Mondhatná erre valaki: „No és?” Hát ez baj, mégpedig nagy baj, hiszen a gyakorlatban éppen azért végezzük láncszerűen a műveleteket, mert így egyszerű, könnyen elvégezhető és csak kis számú algoritmusra (műveletsorozatra) van szükségünk. (Képzeld el hova vezetne, ha külön kiszámítási módszerre volna szükség a kéttényezős, a háromtényezős, stb. szorzások elvégzésére!) Valahogy tehát mégiscsak láncszerűvé kellene tenni a Descartes-szorzást! Egy kis csalással ezt meg is tudjuk tenni.

A *Descartes-szorzás láncszerű műveletvégzésének a módszere* az, hogy a második szorzástól kezdve, amikor a láncban egyet jobbra lépünk, és egy új műveletet készülünk elvégezni, akkor a baloldali rekordban, mint az addigi műveletvégzés eredményében „felnyitjuk” a jobboldali „)” zárójelet, „beengedjük” az új elemet, majd szépen „visszacukjuk” azt. Hát ez bizony csalás, mert ennek algebrailag semmi értelme, de ilyen a világ, a problémákat valahogy meg kell oldani! Az eredmény végülis helyes, és ez a fő! Azért persze érdemes megvizsgálni, hogy a Descartes-szorzás e furcsa tulajdonságát mi okozza.

Figyeljünk fel arra, hogy a láncműveletek operandusai és az eredményük is ugyanabból a halmazból, a művelet úgynevezett *alaphalmazából* valók. (Éppen e tulajdonságával fogjuk a későbbiekben definiálni a művelet fogalmát.) Ezzel szemben a Descartes-szorzásnak általában már az operandusai is különböző halmazból származnak (lásd a fenti példát).

Tekintsük azonban az egyszerűség kedvéért az  $A \times A$  Descartes-szorzatot (melyben tehát mindkét operandus halmaza azonos). Az  $A \times A$  eredménye – mint tudjuk – olyan  $\langle a, b \rangle$  párosokból álló halmaz, melyben  $a, b \in A$ . Nyilvánvaló tehát, hogy  $\langle a, b \rangle \notin A$ , vagyis a Descartes-szorzás algebrai értelemben *nem művelet*!

A fentieket összefoglalva megállapíthatjuk, hogy egy algebrai művelet a művelet operandusainak rendezett  $n$ -esét (bináris művelet esetén az operandus-párosokat) a művelet alaphalmazának egy elemébe képezi le. Ez a leképezés teszi lehetővé egy összetett műveletsor „láncszerű” elvégzését, hiszen akkor nem más történik, mint balról-jobbra haladva minden bináris műveletet helyettesítünk az eredményével. A Descartes-szorzás esetén azonban ez a leképezés elmarad. Egy  $A \times A$  bináris Descartes-szorzás eredménye maga a rendezett  $\langle a, b \rangle$  páros (ahol  $a, b \in A$ ), mely így semmiképpen nem lehet eleme az operandusok  $A$  halmazának. Mivel a Descartes-szorzásnál a fent említett leképezés (helyettesítés) elmarad, így nyilván nem lehet láncszerűen elvégezni például egy  $A \times A \times A \times A$  Descartes-szorzást.

## Halmazok hatványozása

A Descartes-szorzás alapján egy  $A$  halmaz  $n$ -edik hatványát is értelmezzük, mégpedig a következőképpen:

$$A^n \triangleq A_1 \times \dots \times A_n, \text{ ahol minden } i \in I_n \text{ esetén } A_i = A.$$

Természetesen a halmazok hatványozását is (hasonlóan a Descartes-szorzáshoz) kizárólag monohalmazokra fogjuk használni.

Például az  $n$ -dimenziós Euklideszi tér felírható  $\mathbf{R}^n$  alakban ( $\mathbf{R}$  a valós számok halmaza).

### Megjegyzés

1. A Descartes-szorzás elvégzésére tett fenti megjegyzés szerint  $A^{n+1} \neq A^n \times A$ , de az ott leírt módon azért nyilván el tudjuk végezni láncszerűen a halmazhatványozást.
2. Egy  $A$  halmaz  $n$ -edik hatványát tekinthetjük úgy, mint egy olyan  $A$  hiperhalmaz fölötti Descartes-szorzatot, melyben minden  $A_i \in A$  esetén  $A_i = A$ .

## Rekordvetítés (halmaz-relatív és vektor-relatív alrekord)

Legyen  $\underline{A}$  egy halmazvektor,  $\mathbf{D}(\underline{A})$  az ezen értelmezett Descartes-szorzat, valamint  $\underline{r} \in \mathbf{D}(\underline{A})$ . Legyen továbbá  $B$  egy hiperhalmaz, és  $\underline{B}$  ennek egy vektora.

1. Az  $\underline{r}$  rekord  $B$ -vetülete (az  $\underline{r}$  rekordnak a  $B$  hiperhalmazra vonatkozó alrekordja, vagy másként az  $\underline{r}$  rekord  $B$ -alrekordja) egy  $\underline{s}$  vektor, azaz

$$\underline{s} = \underline{r} \upharpoonright_B,$$

ha

- 1.1.  $\underline{s} \in \mathbf{D}(\underline{A} \upharpoonright_B)$ , és

- 1.2. minden  $i \in \text{Ind}(\underline{A} \upharpoonright_B)$  és  $A_i = \underline{A} \upharpoonright_B(i)$  esetén  $\underline{s}(A_i) = \underline{r}(A_i)$ .

2. Az  $\underline{r}$  rekord  $\underline{B}$ -vetülete (az  $\underline{r}$  rekordnak a  $\underline{B}$  halmazvektorra vonatkozó alrekordja, vagy másként az  $\underline{r}$  rekord  $\underline{B}$ -alrekordja) egy  $\underline{s}$  vektor, azaz

$$\underline{s} = \underline{r} \upharpoonright_{\underline{B}},$$

ha

$$2.1. \quad \underline{s} \in \mathbf{D}(\underline{A} \upharpoonright_{\underline{B}}), \text{ és}$$

$$2.2. \quad \text{minden } i \in \text{Ind}(\underline{A} \upharpoonright_{\underline{B}}) \text{ és } A_i = \underline{A} \upharpoonright_{\underline{B}}(i) \text{ esetén } \underline{s}(A_i) = \underline{r}(A_i).$$

A fenti  $\underline{r} \upharpoonright_{\underline{B}}$  és  $\underline{r} \upharpoonright_{\underline{B}}$  esetekben az  $\underline{r}$ -t a rekordvetítés *vetített rekordjának*,  $\underline{B}$ -t a *vetítő-hiperhalmazának*, és  $\underline{B}$ -t a *vetítő-halmazvektorának* is nevezzük, és nyilván

$$\underline{s} = \text{Comp}(\underline{r} \upharpoonright_{\underline{B}}) = \text{Comp}(\underline{r} \upharpoonright_{\underline{B}}) = \{ \underline{r}(A_i) \mid A_i \in \underline{A} \upharpoonright_{\underline{B}} \},$$

ahol  $\underline{A}$  egy hiperhalmaz, mégpedig az  $\underline{A}$  halmazvektor alaphalmaza.

### Példa

Legyen  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x\}$ ,  $A_4 = \{x, y\}$ , továbbá  $\underline{A} = \langle A_1, A_2, A_3 \rangle$ ,  $\underline{B} = \langle A_2, A_1, A_4 \rangle$ , végül  $\underline{r} \in \mathbf{D}(\underline{A})$  és  $\underline{r} = \langle 3, b, x \rangle$ . Határozzuk meg az  $\underline{s} = \underline{r} \upharpoonright_{\underline{B}}$  és a  $\underline{t} = \underline{r} \upharpoonright_{\underline{B}}$  vektorokat.

### Megoldás

1. lépés (A halmazvektorok egyszerű vetületeinek meghatározása)

$$\underline{A} \upharpoonright_{\underline{B}} = \langle A_1, A_2 \rangle, \text{ és}$$

$$\underline{A} \upharpoonright_{\underline{B}} = \langle A_2, A_1 \rangle.$$

2. lépés (A rekordvetületek meghatározása)

$$\underline{s} = \underline{r} \upharpoonright_{\underline{B}} \text{ esetén } \underline{s} \in \mathbf{D}(\underline{A} \upharpoonright_{\underline{B}}), \text{ azaz}$$

$$\underline{s}(A_1) = \underline{r}(A_1) = \langle 3, b, x \rangle(A_1) = 3, \text{ és}$$

$$\underline{s}(A_2) = \underline{r}(A_2) = \langle 3, b, x \rangle(A_2) = b, \text{ tehát}$$

$$\underline{s} = \langle 3, b \rangle.$$

$$\underline{t} = \underline{r} \upharpoonright_{\underline{B}} \text{ esetén } \underline{t} \in \mathbf{D}(\underline{A} \upharpoonright_{\underline{B}}), \text{ azaz}$$

$$\underline{t}(A_1) = \underline{r}(A_1) = \langle 3, b, x \rangle(A_1) = 3, \text{ és}$$

$$\underline{t}(A_2) = \underline{r}(A_2) = \langle 3, b, x \rangle(A_2) = b, \text{ tehát}$$

$$\underline{t} = \langle b, 3 \rangle.$$

### Megjegyzés

Mivel a rekordok vektorok, így az *alrekord* fogalma az alvektor fogalmával. Ebből következően egy  $\underline{r}$  rekordnak tetszőleges  $\underline{B}$  hiperhalmazra vonatkozó  $\underline{r} \upharpoonright_{\underline{B}}$  vetülete nyilván alvektora (alrekordja) az  $\underline{r}$  rekordnak, azaz  $\underline{r} \upharpoonright_{\underline{B}} \subseteq \underline{r}$ , de általában nem teljesül  $\underline{r} \upharpoonright_{\underline{B}} \subseteq \underline{r}$ .

## Hasonló rekordok (relatív azonos rekordok)

### 1. Definíció

Legyen  $\underline{A}$  egy halmazvektor,  $\mathbf{D}(\underline{A})$  az ezen értelmezett Descartes-szorzat,  $\underline{r}, \underline{s} \in \mathbf{D}(\underline{A})$ , továbbá  $\underline{B}$  egy hiperhalmaz. Ekkor az  $\underline{r}$  és az  $\underline{s}$  a  $\underline{B}$  hiperhalmazra vonatkozóan *relatív azonos rekordok*, vagy röviden *B-hasonló rekordok*, ha a  $\underline{B}$ -vetületeik azonosak, azaz

$$\underline{r} \sim_{\underline{B}} \underline{s}, \text{ ha } \underline{r} \upharpoonright_{\underline{B}} = \underline{s} \upharpoonright_{\underline{B}}.$$

A rekordok relatív azonosságának definiálásában megkerülhetjük a rekord vetületének (relatív alrekordjának) fogalmát az alábbi módon:

### 2. Definíció

Legyen  $\underline{A}$  egy halmazvektor,  $\mathbf{D}(\underline{A})$  az ezen értelmezett Descartes-szorzat,  $\underline{r}, \underline{s} \in \mathbf{D}(\underline{A})$ , továbbá  $A$  az  $\underline{A}$  halmazvektor hiperhalmaza és  $B$  egy másik hiperhalmaz. Ekkor az  $\underline{r}$  és az  $\underline{s}$  a  $B$  hiperhalmazra vonatkozóan *relatív azonos rekordok*, vagy röviden *B-hasonló rekordok*, ha az  $\underline{r}$  és az  $\underline{s}$  vektorok komponensei *páronként egyenlők*, azaz

$$\underline{r} \stackrel{B}{\sim} \underline{s}, \text{ ha minden } A_i \in A|_B \text{ esetén } \underline{r}(A_i) = \underline{s}(A_i).$$

### 3. Definíció

Legyen  $\underline{A}$  egy halmazvektor,  $\mathbf{D}(\underline{A})$  az ezen értelmezett Descartes-szorzat,  $\underline{r}, \underline{s} \in \mathbf{D}(\underline{A})$ , továbbá  $\underline{B}$  egy másik halmazvektor. Ekkor az  $\underline{r}$  és az  $\underline{s}$  a  $\underline{B}$  halmazvektorra vonatkozóan *relatív azonos rekordok*, vagy röviden  *$\underline{B}$ -hasonló rekordok*, ha a  $\underline{B}$ -vetületeik azonosak, azaz

$$\underline{r} \stackrel{\underline{B}}{\sim} \underline{s}, \text{ ha } \underline{r} \upharpoonright_{\underline{B}} = \underline{s} \upharpoonright_{\underline{B}}.$$

### Megjegyzés

A rekordok hasonlóságának első két definíciója ekvivalens, hiszen a 2. definíció az 1. definícióban szereplő rekord-vetület fogalmának kifejtésén alapul. A 2. definíció tehát egyszerűbb, ezért általában majd erre hivatkozunk, az 1. viszont tömörebb, szemléletesebb.

A 3. definícióban a  $B$  hiperhalmaz helyett szereplő  $\underline{B}$  halmazvektor láthatóan szintén nem változtatja meg a rekordhasonlóság fogalmát, mindössze a rekordvetületek komponenseinek sorrendje, ebből következően pedig a feltételként szereplő vetület-egyenlőségben a komponens-egyenlőség kiértékelési sorrendje módosul.

A rekordok hiperhalmazra (halmazvektorra) vonatkozó hasonlósága igen fontos fogalom, mivel *rekordtulajdonságként fejezi ki a rekordok részleges egyezőségét*. Segítségével származtatjuk majd a relációsémák osztályozását, és a funkcionális függőséget is.

Végül figyeljünk fel a vektorhasonlóság és a rekordhasonlóság fogalmának kapcsolatára. Könnyen belátható, hogy  $\underline{r} \stackrel{B}{\sim} \underline{s}$ , illetve  $\underline{r} \stackrel{\underline{B}}{\sim} \underline{s}$  esetén  $\underline{r} \sim \underline{s}$  is fennáll, vagyis a rekordhasonlóságból következik a vektorhasonlóság.

## Vektorhalmaz (a Descartes-szorzatból származtatva)

Részben a vektor fogalmának bevezetésénél, részben a vektorhalmaz-vetítés bevezető gondolatainál már említettük, hogy a vektor pontos fogalmát a Descartes-szorzaton keresztül tudjuk értelmezni. Ezt a fentiekben megtettük. Ebből azonban következik, hogy a vektorokból álló halmaznál is lehetséges a Descartes-szorzattól kiindulni. A módszer lényege az, hogy egy vektorhalmazt valamely  $A$  halmazvektor fölött értelmezett  $\mathbf{D}(\underline{A})$  Descartes-szorzat részhalmazaként tekintünk, illetve amikor a multi-vektorhalmazokat is meg akarjuk engedni, akkor redukált részhalmazaként tekintünk.

Fontosnak tartjuk megjegyezni, hogy a vektorhalmaz Descartes-szorzat nélkül is értelmezhető, egyszerűen vektorok halmazaként. Sok esetben ez a célszerű. Amikor az elemi vektorok között nincs algebrai kapcsolat (például nem lehet beszélni a komponensek azonos halmazból való származtatásáról), vagy nemrektanguláris vektorhalmazokat vizsgálunk, akkor a Descartes-szorzatra való hivatkozás nehézkes, esetleg lehetetlen.

Annak oka, hogy a továbbiakban a vektorhalmazoknak a fent jelzett (Descartes-szorzat részhalmazaként való) értelmezését használjuk, éppen az, hogy az általunk tárgyalt vektorhalmazok egyrészt rektangulárisak, másrészt az elemi rekordjai között nagyon is erős kap-

csolat áll fenn (éppen az egyes rekordok azonos pozíción elhelyezkedő komponenseinek azonos halmazból való származtatása).

Az alábbi definícióban már erre az értelmezésre láthatunk példát. E speciális származtatás jelzésére a továbbiakban a vektorhalmazokat (a Descartes-szorzat jelöléséhez hasonlóan) egyrészt vastag álló betűvel jelezzük, másrészt jelölésükben hivatkozunk arra a halmazvektorra, amelyből származtattuk. Például egy  $\underline{A}$  halmazvektoron értelmezett vektorhalmazt  $\mathbf{R}(\underline{A})$  módon jelölünk (ahol tehát  $\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$ ).

A Descartes-szorzat definíciójánál már említettük, hogy a továbbiakban e fogalmat kizárólag monovektoron fogjuk használni, annak érdekében, hogy a Descartes-szorzat monohalmaz legyen (ami a későbbiekben lesz fontos a számunkra). Ha  $\underline{A}$  monovektor, akkor tehát  $\mathbf{D}(\underline{A})$  egy monohalmaz, és így nyilván  $\mathbf{R}(\underline{A})$  is monohalmaz, ha  $\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$ . Az alábbi definíció azonban akkor is értelmezhető, ha a vektorhalmaz, amire a vetítést értelmezzük nem monohalmaz (hanem valódi multihalmaz), így az  $\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$  feltétel helyett az ennél gyengébb  $\mathbf{R}(\underline{A}) \subseteq^R \mathbf{D}(\underline{A})$  feltételt szabjuk csak meg. (A későbbiekben egy  $\mathbf{D}(\underline{A})$  Descartes-szorzat redukált részhalmazát relációnak fogjuk nevezni, és  $R(\underline{A})$  módon jelöljük.)

### Vektorhalmaz-vetítés (halmaz-relatív és vektor-relatív vetítés)

Legyen  $\underline{A}$  egy mono-halmazvektor, és  $\mathbf{R}(\underline{A})$  egy ezen értelmezett vektorhalmaz (melyre tehát  $\mathbf{R}(\underline{A}) \subseteq^R \mathbf{D}(\underline{A})$  teljesül, ahol  $\mathbf{D}(\underline{A})$  az  $\underline{A}$ -n értelmezett Descartes-szorzat). Legyen továbbá  $B$  egy hiperhalmaz,  $\underline{B}$  pedig ennek egy vektora.

1. Az  $\mathbf{R}(\underline{A})$  vektorhalmaznak a  $B$  halmaz-relatív vektorhalmaz-vetülete, vagy  $B$ -vetülete:

$$\mathbf{R}(\underline{A})\|_B \triangleq \{r \upharpoonright_B \mid r \in \mathbf{R}(\underline{A})\}.$$

2. Az  $\mathbf{R}(\underline{A})$  vektorhalmaznak a  $\underline{B}$  vektor-relatív vektorhalmaz-vetülete, vagy  $\underline{B}$ -vetülete:

$$\mathbf{R}(\underline{A})\|_{\underline{B}} \triangleq \{r \upharpoonright_{\underline{B}} \mid r \in \mathbf{R}(\underline{A})\}.$$

Vegyük észre, hogy az  $\mathbf{R}(\underline{A})\|_B$ , és az  $\mathbf{R}(\underline{A})\|_{\underline{B}}$  vektorhalmazok akkor is lehetnek valódi multihalmazok, ha  $\mathbf{R}(\underline{A})$  monohalmaz. Ennek oka, hogy vektorhalmaz-vetítés során egyes rekordokból olyan komponensek is kiválhatnak, amelyek azokat egyedivé tették  $\mathbf{R}(\underline{A})$ -ban.

#### Állítás

1. Tetszőleges  $\underline{A}$  mono-halmazvektor,  $\mathbf{R}(\underline{A})$  vektorhalmaz és  $B$  hiperhalmaz esetén

- 1.1.  $\mathbf{R}(\underline{A})\|_B$ , és  $\mathbf{R}(\underline{A})\|_{\underline{B}}$  szintén vektorhalmazok, és

- 1.2.  $\mu(\mathbf{R}(\underline{A})\|_B) = \mu(\mathbf{R}(\underline{A})\|_{\underline{B}}) = \mu(\mathbf{R}(\underline{A}))$ .

2. Tetszőleges  $\underline{A}$  mono-halmazvektor és  $\mathbf{R}(\underline{A})$  vektorhalmaz esetén

$$\mathbf{R}(\underline{A})\|_A = \mathbf{R}(\underline{A})\|_{\underline{A}} = \mathbf{R}(\underline{A}),$$

ahol  $A$  az  $\underline{A}$  halmazvektor alaphalmaza.

3. Tetszőleges  $\underline{A}$  és  $\underline{B}$  halmazvektorok, valamint  $\mathbf{R}(\underline{A})$  vektorhalmaz esetén,

$$\mathbf{R}(\underline{A})\|_{\underline{B}} \subseteq^R \mathbf{D}(\underline{A} \upharpoonright_{\underline{B}}).$$

#### Példa

Legyen  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x, y\}$ ,  $A_4 = \{a, x\}$ , továbbá  $\underline{A} = \langle A_1, A_2, A_3 \rangle$ ,  $\underline{B} = \langle A_2, A_1, A_4 \rangle$  két halmazvektor, végül  $\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$  egy vektorhalmaz, ahol  $\mathbf{R}(\underline{A}) = \{\langle 3, a, x \rangle,$

$\langle 3, a, y \rangle, \langle 4, b, x \rangle\}$ . Ekkor nyilván  $\mathbf{R}\langle \underline{A} \rangle \parallel_B = \{\langle 3, a \rangle, \langle 3, a \rangle, \langle 4, b \rangle\}$ , és  $\mathbf{R}\langle \underline{A} \rangle \parallel_B = \{\langle a, 3 \rangle, \langle a, 3 \rangle, \langle b, 4 \rangle\}$  (tehát valódi multihalmazokat kaptunk!), továbbá  $\mathbf{R}\langle \underline{A} \rangle \parallel_A = \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{A}} = \mathbf{R}\langle \underline{A} \rangle$ .

## Alvektorhalmaz

Legyen  $\underline{A}$  egy mono-halmazvektor, és  $\mathbf{R}\langle \underline{A} \rangle$  egy ezen értelmezett vektorhalmaz. Ekkor egy  $\mathbf{S}$  vektorhalmaz *alvektorhalmaza* az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaznak, azaz

$$\mathbf{S} \subseteq \mathbf{R}\langle \underline{A} \rangle,$$

ha

1.  $\mu(\mathbf{S}) = \mu(\mathbf{R}\langle \underline{A} \rangle)$ ,
2. létezik az  $\mathbf{R}\langle \underline{A} \rangle$  és az  $\mathbf{S}$  elemi vektorainak egy bijektív (kölsönösen egyértelmű össze-  
rendelést biztosító)  $i : I_{\mu(\mathbf{S})} \rightarrow I_{\mu(\mathbf{R}\langle \underline{A} \rangle)}$  indextranszformációja (vagyis minden  $r_k \in \mathbf{R}\langle \underline{A} \rangle$  elemi  
vektorhoz kölsönösen egyértelműen hozzárendelhető egy  $s_{i(k)} \in \mathbf{S}$  vektor), melyre min-  
den  $r_k \in \mathbf{R}\langle \underline{A} \rangle$  esetén

$$s_{i(k)} \subseteq r_k.$$

### Megjegyzés

Összetett halmazok esetén figyelmen kívül hagyhatjuk, hogy az elemek összetett objektu-  
mok, így egy vektorhalmaznak értelmezhetjük a részhalmazát hagyományos halmazalgebrai  
módon. Ekkor a részhalmaz az eredeti vektorhalmaz egyes elemi vektorait fogja tartalmazni  
(lásd a „Vektorhalmaz egyszerű halmazvetítése, részhalmaza” pontot). Az alvektorhalmaz  
ezzel szemben minden elemi vektort tartalmaz, de nem teljes egészében, pontosabban az  
alvektorhalmaz elemi vektorai az eredeti vektorhalmaz elemi vektorainak alvektorai.

### Példa

Legyen  $\underline{A} = \langle A_1, A_2, A_3 \rangle$  egy halmazvektor, melyben  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x, y\}$ ,  
továbbá  $\mathbf{R}\langle \underline{A} \rangle \subseteq \mathbf{D}\langle \underline{A} \rangle$  egy vektorhalmaz, ahol  $\mathbf{R}\langle \underline{A} \rangle = \{\langle 3, a, x \rangle, \langle 3, a, y \rangle, \langle 4, b, x \rangle\}$ . Ekkor  
nyilván  $\{\langle 3, a \rangle, \langle 3, a \rangle, \langle 4, b \rangle\} \subseteq \mathbf{R}\langle \underline{A} \rangle$ .

### Állítás

A definícióból egyszerűen következnek az alábbiak

1.  $\mathbf{R}\langle \underline{A} \rangle \subseteq \mathbf{R}\langle \underline{A} \rangle$ .
2. Tetszőleges  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaz és  $B$  hiperhalmaz esetén, ha  $\mathbf{R}\langle \underline{A} \rangle \parallel_B \neq \emptyset$ , akkor

$$\mathbf{R}\langle \underline{A} \rangle \parallel_B \subseteq \mathbf{R}\langle \underline{A} \rangle.$$

## Hasonlósági vetítés

Legyen  $\underline{A}$  és  $\underline{B}$  két mono-halmazvektor, melyekre  $\underline{A} \upharpoonright_B \neq \emptyset$ , továbbá  $\mathbf{R}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle$  ezeken  
értelmezett két vektorhalmaz (azaz  $\mathbf{R}\langle \underline{A} \rangle \subseteq \mathbf{D}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle \subseteq \mathbf{D}\langle \underline{B} \rangle$ ).

Ekkor az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaznak az  $\mathbf{S}\langle \underline{B} \rangle$  vektorhalmazra vonatkozó *hasonlósági vetülete*,  
vagy röviden az  $\mathbf{S}\langle \underline{B} \rangle$ -*hasonló vetülete* az alábbi vektorhalmaz:

$$\mathbf{R}\langle \underline{A} \rangle \parallel_{\mathbf{S}\langle \underline{B} \rangle} \triangleq \{ r \mid r \in \mathbf{R}\langle \underline{A} \rangle, \exists s \in \mathbf{S}\langle \underline{B} \rangle : s \stackrel{\underline{A} \upharpoonright_B}{\sim} r \}.$$

**Állítás**

A definícióból egyszerűen következnek az alábbiak

$$1. \mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{R}\langle \underline{A} \rangle} = \mathbf{R}\langle \underline{A} \rangle.$$

2. Ha  $\underline{A} \upharpoonright_{\underline{B}} \neq \emptyset$  és  $\mathbf{Q} = \mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}}$ , akkor

$$\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{Q}} = \mathbf{R}\langle \underline{A} \rangle,$$

és a vektorhalmaz-vetítésnél kimondott  $\mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}} \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{B} \rangle$  állítás miatt  $\mathbf{Q} = \mathbf{Q}\langle \underline{A} \upharpoonright_{\underline{B}} \rangle$ .

**Példa**

Tekintsük az  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x, y\}$  és  $A_4 = \{u, v\}$  halmazokat, továbbá az  $\underline{A} = \langle A_1, A_2, A_3 \rangle$  és  $\underline{B} = \langle A_2, A_1, A_4 \rangle$  halmazvektorokat, végül pedig az  $\mathbf{R}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle$  vektorhalmazokat, ahol  $\mathbf{R}\langle \underline{A} \rangle = \{\langle 3, a, x \rangle, \langle 3, a, y \rangle, \langle 4, b, x \rangle\}$  és  $\mathbf{S}\langle \underline{B} \rangle = \{\langle a, 3, u \rangle, \langle b, 3, v \rangle\}$ . Ekkor nyilván  $\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{S}\langle \underline{B} \rangle} = \{\langle 3, a, x \rangle, \langle 3, a, y \rangle\}$ , és természetesen  $\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{R}\langle \underline{A} \rangle} = \mathbf{R}\langle \underline{A} \rangle$ .

Legyen  $\mathbf{Q} = \mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}}$ . Ekkor  $\mathbf{Q} = \{\langle a, 3 \rangle, \langle a, 3 \rangle, \langle b, 4 \rangle\}$  (láthatóan valóban  $\mathbf{Q} \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{A} \upharpoonright_{\underline{B}} \rangle$ ), és  $\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{Q}} = \mathbf{R}\langle \underline{A} \rangle$ .

**Hasonló vektorhalmazok**

Tekintsük az  $\underline{A}$ ,  $\underline{B}$  és  $\underline{C}$  mono-halmazvektorokat, valamint az  $\mathbf{R}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle$  vektorhalmazokat (ahol  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{B} \rangle$ ).

Ekkor az  $\mathbf{R}\langle \underline{A} \rangle$  és az  $\mathbf{S}\langle \underline{B} \rangle$  vektorhalmazok hasonlóak a  $\underline{C}$  halmazvektorra vonatkozóan, vagy röviden  $\underline{C}$ -hasonlóak, azaz

$$\mathbf{R}\langle \underline{A} \rangle \stackrel{\underline{C}}{\approx} \mathbf{S}\langle \underline{B} \rangle,$$

ha

1.  $\underline{A} \upharpoonright_{\underline{B}} \neq \emptyset$ ,
2.  $(\underline{A} \upharpoonright_{\underline{B}}) \upharpoonright_{\underline{C}} \neq \emptyset$ , és
3. minden  $\underline{r} \in \mathbf{R}\langle \underline{A} \rangle$  és  $\underline{s} \in \mathbf{S}\langle \underline{B} \rangle$  esetén  $\underline{r} \stackrel{\underline{C}}{\approx} \underline{s}$ .

**Állítás**

A fenti jelöléseket alkalmazva a definícióból következően fennállnak az alábbi állítások:

1.  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\underline{C}}{\approx} \mathbf{R}\langle \underline{A} \rangle$ .
2.  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\underline{C}}{\approx} \mathbf{S}\langle \underline{B} \rangle \models \mathbf{S}\langle \underline{B} \rangle \stackrel{\underline{C}}{\approx} \mathbf{R}\langle \underline{A} \rangle$ .
3.  $\mathbf{S}\langle \underline{B} \rangle \stackrel{\underline{C}}{\approx} \mathbf{R}\langle \underline{A} \rangle \models \mathbf{S}\langle \underline{B} \rangle \stackrel{\underline{C}}{\approx} \mathbf{R}\langle \underline{A} \rangle$ .

**Példa**

Tekintsük az  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x, y\}$ ,  $A_4 = \{u, v, w\}$  és  $A_5 = \{y, 2\}$  halmazokat, továbbá az  $\underline{A} = \langle A_1, A_2, A_3 \rangle$ ,  $\underline{B} = \langle A_2, A_1, A_4 \rangle$  és  $\underline{C} = \langle A_1, A_5, A_2 \rangle$  halmazvektorokat, végül pedig az  $\underline{A}$  és  $\underline{B}$  fölötti  $\mathbf{R}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle$  vektorhalmazokat, ahol  $\mathbf{R}\langle \underline{A} \rangle = \{\langle 3, a, x \rangle, \langle 3, a, y \rangle\}$  és  $\mathbf{S}\langle \underline{B} \rangle = \{\langle a, 3, u \rangle, \langle a, 3, v \rangle, \langle a, 3, w \rangle\}$ . Ekkor láthatóan  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\underline{C}}{\approx} \mathbf{S}\langle \underline{B} \rangle$ , hiszen bármelyik rekordját tekintjük is az  $\mathbf{R}\langle \underline{A} \rangle$  vagy az  $\mathbf{S}\langle \underline{B} \rangle$  vektorhalmazoknak, azok az  $A_1$  és  $A_2$  attribútumon mind megegyeznek, az  $A_5$  attribútumon pedig egyiknek sincs értéke.

### Transzvertált vetítés

Legyen  $\underline{A}$  egy mono-halmazvektor,  $\mathbf{R}\langle \underline{A} \rangle$  egy ezen értelmezett vektorhalmaz, és  $A$  az  $\underline{A}$  alaphalmaza.

1. Az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaznak egy  $B$  hiperhalmazra vonatkozó transzvertált vetülete, vagy röviden a  $B$ -transzvertáltja egy  $\underline{S}$  halmazvektor, azaz

$$\underline{S} = \mathbf{R}\langle \underline{A} \rangle \#_B,$$

ha

$$1.1. \mu(\underline{S}) = \mu(A|_B), \text{ és}$$

$$1.2. \text{ minden } S_i \in \underline{S} \text{ esetén}$$

$$S_i = \bigcup_{\ell \in \mathbf{R}\langle \underline{A} \rangle} \{\ell(A_i)\}, \text{ ahol } A_i = \underline{A} \upharpoonright_B(i),$$

tehát az  $\underline{S}$  halmazvektor sorrendezése az  $\underline{A}$  halmazvektor szerint történik.

2. Az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaznak egy  $\underline{B}$  halmazvektorra vonatkozó transzvertált vetülete, vagy röviden a  $\underline{B}$ -transzvertáltja egy  $\underline{S}$  halmazvektor, azaz

$$\underline{S} = \mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}},$$

ha

$$2.1. \mu(\underline{S}) = \mu(A|_{\underline{B}}), \text{ és}$$

$$2.2. \text{ minden } S_i \in \underline{S} \text{ esetén}$$

$$S_i = \bigcup_{\ell \in \mathbf{R}\langle \underline{A} \rangle} \{\ell(A_i)\}, \text{ ahol } A_i = \underline{A} \upharpoonright_{\underline{B}}(i),$$

tehát az  $\underline{S}$  halmazvektor sorrendezése a  $\underline{B}$  halmazvektor szerint történik.

#### Példa

Legyen  $\underline{A} = \langle A_1, A_2, A_3 \rangle$  és  $\underline{B} = \langle A_3, A_1, A_4 \rangle$  két halmazvektor, ahol  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x, y\}$ ,  $A_4 = \{u, v\}$ , és  $\mathbf{R}\langle \underline{A} \rangle$  egy vektorhalmaz az  $\underline{A}$  fölött, ahol  $\mathbf{R}\langle \underline{A} \rangle = \{\langle 3, a, x \rangle, \langle 3, a, y \rangle, \langle 4, b, x \rangle\}$ . Ekkor nyilván  $\mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}} = \langle \{x, y\}, \{3, 4\} \rangle$ .

### Transzvertálás

Legyen  $\underline{A}$  egy mono-halmazvektor, és  $\mathbf{R}\langle \underline{A} \rangle$  egy ezen értelmezett vektorhalmaz. Ekkor az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaz transzvertáltja

$$\mathbf{R}^\# \langle \underline{A} \rangle \hat{=} \mathbf{R}\langle \underline{A} \rangle \#_A,$$

ahol  $A$  az  $\underline{A}$  halmazvektor alaphalmaza. Természetesen  $\mathbf{R}^\# \langle \underline{A} \rangle$  egy halmazvektor. Az alábbi állítások a definícióból közvetlenül következnek.

#### Állítás

1. Tetszőleges  $\underline{A}$  mono-halmazvektor esetén

$$\mathbf{D}^\# \langle \underline{A} \rangle = \underline{A}.$$

2. Tetszőleges  $\underline{A}$  mono-halmazvektor és ezen értelmezett  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaz esetén

$$2.1. \mathbf{R}^\# \langle \underline{A} \rangle \subseteq^* \underline{A}, \text{ és}$$

$$2.2. \mu(\mathbf{R}^\# \langle \underline{A} \rangle) = \mu(\underline{A}).$$

3. Tetszőleges  $\underline{A}$  és  $\underline{B}$  mono-halmazvektorok esetén

$$\mathbf{R}(\underline{A}) \#_B \subseteq^* \mathbf{R}^{\#}(\underline{A}),$$

ahol  $B$  a  $\underline{B}$  halmazvektor alaphalmaza.

#### Példa

Tekintsük az előző példát. Ekkor nyilván  $\mathbf{R}^{\#}(\underline{A}) = \langle \{3, 4\}, \{a, b\}, \{x, y\} \rangle$ , melyre láthatóan teljesül, hogy  $\mathbf{R}^{\#}(\underline{A}) \subseteq^* \underline{A}$ .

### Megjegyzések a transzvertáláshoz és a transzvertált vetítéshez

Nyilván csak rektanguláris vektorhalmaznak lehet értelmezni a transzvertált vetületét és a transzvertáltját (a Descartes-szorzat bármely részhalmaza ilyen vektorhalmaz!).

Egy  $\mathbf{R}(\underline{A})$  vektorhalmaz transzvertálásával tulajdonképpen előállítjuk az  $\mathbf{R}(\underline{A})$  rekordjaiban szereplő egyes komponensek értékkészleteit, vagyis azokat a halmazokat, melyekből az  $\mathbf{R}(\underline{A})$  rekordjainak egyes komponensei az értékeiket veszik. Az  $\mathbf{R}^{\#}(\underline{A})$  transzvertált egyes komponenseit alkotó halmazokat tehát az adott komponenshez tartozó *értékkészletnek*, magát az  $\mathbf{R}^{\#}(\underline{A})$  transzvertáltat pedig az  $\mathbf{R}(\underline{A})$  *értéktartományának* is tekinthetjük. Ez a szemlélet mutatja a transzvertálás igazi jelentőségét.

Amikor azonban vektorhalmazból transzvertálással halmazvektort képezünk, akkor egy nagyon fontos információt elveszítünk, mégpedig a halmazvektor halmazaiiban szereplő elemek közötti kapcsolat információját. A fenti példa esetén az  $\mathbf{R}^{\#}(\underline{A})$  transzvertált egyes komponenseit alkotó halmazokat tekintve a  $\langle 4, a, x \rangle$  akár rekordja is lehetne az  $\mathbf{R}(\underline{A})$  vektorhalmaznak (bár tudjuk nem rekordja). Ebből pedig az következik, hogy egy vektorhalmaz transzvertált halmazvektorából már nem tudjuk visszaállítani az eredeti vektorhalmazt.

A fenti két bekezdésben mondottak szemléltetéseként tekintsük az  $\underline{A} = \langle X, Y, Z \rangle$  halmazvektort, mint egy háromdimenziós metrikus teret (ahol tehát az  $X$ , az  $Y$  és a  $Z$  valós számokat tartalmazó halmazok a „koordinátatengelyek”), és alkossanak itt gömböt egy  $\mathbf{R}(\underline{A})$  vektorhalmaz vektorai, mint az  $\underline{A}$  tér pontjai (helyvektorai). Ekkor az  $\mathbf{R}^{\#}(\underline{A})$  halmazvektor egyes komponensei az  $\mathbf{R}(\underline{A})$  gömb egyes tengelyekre eső vetületei, vagyis az  $\mathbf{R}(\underline{A})$  vektorhalmaz értéktartománya az  $X$ , az  $Y$  és a  $Z$  halmazban. Magát az  $\mathbf{R}^{\#}(\underline{A})$  halmazvektort tehát úgy is tekinthetjük, mint az  $\underline{A}$  tér egy kocka alakú alterét, ahol e kocka lapjai az  $\mathbf{R}(\underline{A})$  gömb érintősíkjain fekszenek.

### Vektorhalmaz parciális és teljes keretei

Legyen  $\underline{A}$  egy mono-halmazvektor, és  $\mathbf{R}(\underline{A})$  egy ezen értelmezett vektorhalmaz.

1. Az  $\mathbf{R}(\underline{A})$  vektorhalmaznak egy  $\underline{B}$  halmazvektorra vonatkozó vektor-értéktartománya, vagy röviden a  $\underline{B}$ -parciális vektorkerete

$$\text{vFrame}(\mathbf{R}(\underline{A}), \underline{B}) \triangleq \mathbf{R}(\underline{A}) \#_{\underline{B}}.$$

2. Az  $\mathbf{R}(\underline{A})$  vektorhalmaznak az  $\underline{A}$  halmazvektorra vonatkozó vektor-értéktartománya, vagy röviden a teljes vektorkerete

$$\text{vFrame}(\mathbf{R}(\underline{A}), \underline{A}).$$

3. Az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaznak egy  $\underline{B}$  halmazvektorra vonatkozó halmaz-értéktartománya, vagy röviden a  $\underline{B}$ -parciális halmazkerete

$$sFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{B}) \triangleq \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}}.$$

4. Az  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaznak az  $\underline{A}$  halmazvektorra vonatkozó halmaz-értéktartománya, vagy röviden a teljes halmazkerete

$$sFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{A}).$$

#### Megjegyzés

A parciális keretek segítségével fogjuk a relációk értéktartományát, valamint a függvények értelmezési tartományát (magját) és értékkészletét (képét) értelmezni.

Figyeljünk fel arra, hogy a halmazkeret definíciójában szereplő vektorhalmaz-vetítés (az összes vetületképzés közül egyedüli módon) akkor is eredményezhet valódi multihalmazt, ha az argumentumában szereplő vektorhalmaz ( $\mathbf{R}\langle \underline{A} \rangle$ ) monohalmaz.

#### Állítás

Tetszőleges  $\underline{A}$  mono-halmazvektor és egy ezen értelmezett  $\mathbf{R}\langle \underline{A} \rangle$  vektorhalmaz esetén a definícióból következően fennállnak az alábbi állítások:

1.  $vFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{A}) = \mathbf{R}^\# \langle \underline{A} \rangle$ .
2.  $sFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{A}) = \mathbf{R}\langle \underline{A} \rangle$ .

#### Példa

Legyen  $\underline{A} = \langle A_1, A_2, A_3 \rangle$  és  $\underline{B} = \langle A_1, A_3, A_4 \rangle$  két halmazvektor, ahol  $A_1 = \{1, 2, 3, 4\}$ ,  $A_2 = \{a, b\}$ ,  $A_3 = \{x\}$ ,  $A_4 = \{x, y\}$ , továbbá  $\mathbf{R}\langle \underline{A} \rangle \subseteq \mathbf{D}\langle \underline{A} \rangle$  egy vektorhalmaz, ahol  $\mathbf{R}\langle \underline{A} \rangle = \{\langle 3, a, x \rangle, \langle 3, b, x \rangle, \langle 4, b, x \rangle\}$ . Ekkor nyilván  $vFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{B}) = \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}} = \{\langle 3, 4 \rangle, \{x\}\}$ , melyre láthatóan teljesül, hogy  $vFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{B}) \subseteq^* \underline{A}$ , továbbá  $sFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{B}) = \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}} = \{\langle 3, x \rangle, \langle 4, x \rangle\}$ , melyre pedig láthatóan az teljesül, hogy  $sFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{B}) \subseteq \mathbf{R}\langle \underline{A} \rangle$ . Megjegyezzük, hogy ez utóbbi teljesüléséhez szükség volt arra, hogy  $\underline{A} \upharpoonright_{\underline{B}} \subseteq \underline{B}$  fennálljon.

Végül nyilván  $vFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{A}) = \{\langle 3, 4 \rangle, \{a, b\}, \{x\}\} = \mathbf{R}^\# \langle \underline{A} \rangle$ , továbbá  $sFrame(\mathbf{R}\langle \underline{A} \rangle, \underline{A}) = \{\langle 3, a, x \rangle, \langle 3, b, x \rangle, \langle 4, b, x \rangle\} = \mathbf{R}\langle \underline{A} \rangle$ .

## Vetítések és tulajdonságai (összefoglaló)

Az alábbiakban áttekintjük az eddig bevezetett vetítési műveleteket, transzformációkat, valamint ezek alapvető tulajdonságait. (Mivel a szereplő halmazok valódi multihalmazok is lehetnek, ezért a halmazműveletek értelemszerűen multiműveletek.)

### Halmazvetítés

$A$  és  $B$  halmazok esetén az

$$A|_B \text{ vetület szintén halmaz.}$$

#### Tulajdonságok, kapcsolat a részhalmazzal

1.  $A|_B = A \cap B$ .

2.  $A|_B \subseteq A$ ,  $B|_A \subseteq B$ ,  $A|_B = B|_A$ ,  $A|_A = A$ .
3.  $B \subseteq A \models A|_B = B$ .

### Vektorvetítés

$\underline{a}$  és  $\underline{x}$  vektorok, valamint ezek alaphalmazai ( $a$  és  $x$ ) esetén az

$\underline{a}|_{\underline{x}}$ ,  $\underline{a}|_{\underline{x}}$  és  $\underline{a}|_{\underline{x}}$  vetületek szintén vektorok.

#### Tulajdonságok, kapcsolat az alvektorral

1. Az  $\underline{a}|_{\underline{x}}$ ,  $\underline{a}|_{\underline{x}}$  és  $\underline{a}|_{\underline{x}}$  vektorok alaphalmaza  $a|_x$ .
2.  $\underline{a}|_{\underline{x}} = \underline{x}|_a$ ,  $\underline{a}|_{\underline{x}} = \underline{x}|_a$ ,  $\underline{a}|_{\underline{a}} = \underline{a}$ .
3.  $\underline{a}|_x \subseteq \underline{a}$ ,  $\underline{a}|_x \subseteq \underline{x}$ ,  $\underline{a}|_x \subseteq \underline{x}$ .
4.  $\underline{x} \subseteq \underline{a} \models \underline{a}|_{\underline{x}} = \underline{x}$ .

### Halmazvektor-vetítés

$\underline{A}$  és  $\underline{B}$  halmazvektorok esetén az

$\underline{A} \parallel \underline{B}$  szintén egy halmazvektor,

melyet csak  $\mu(\underline{B}) = \mu(\underline{A})$  esetén értelmezünk.

#### Tulajdonságok, kapcsolat a résztartománnyal

1.  $\mu(\underline{A} \parallel \underline{B}) = \mu(\underline{A})$ .
2. Minden  $i \in I_{\mu(\underline{A})}$  esetén  $\underline{A} \parallel \underline{B}(i) = A_i|_{B_i}$ , ahol  $A_i = \underline{A}(i)$ , és  $B_i = \underline{B}(i)$ .
3.  $\underline{A} \parallel \underline{B} \subseteq \underline{A}$ ,  $\underline{A} \parallel \underline{A} = \underline{A}$ .
4.  $\underline{B} \subseteq \underline{A} \models \underline{B} \subseteq \underline{A}$ .

### Rekordvetítés

$\underline{A}$  és  $\underline{B}$  halmazvektorok, valamint  $\underline{r} \in \mathbf{D}(\underline{A})$  rekord (vektor) esetén az

$\underline{r}|_{\underline{B}}$  és  $\underline{r}|_{\underline{B}}$  vetületek szintén vektorok,

ahol  $B$  a  $\underline{B}$  halmazvektor alaphalmaza.

#### Tulajdonságok, kapcsolat az alrekorddal (alvektorral)

1.  $\underline{r}|_{\underline{B}}(i) = \underline{r}(\underline{A}|_{\underline{B}}(i))$ , ahol  $i \in \text{Ind}(\underline{A}|_{\underline{B}})$ .
2.  $\underline{r}|_{\underline{B}}(i) = \underline{r}(\underline{A}|_{\underline{B}}(i))$ , ahol  $i \in \text{Ind}(\underline{A}|_{\underline{B}})$ .
3.  $\underline{r}|_{\underline{B}} \in \mathbf{D}(\underline{A}|_{\underline{B}})$  és  $\underline{r}|_{\underline{B}} \in \mathbf{D}(\underline{A}|_{\underline{B}})$ .
4.  $\underline{r}|_{\underline{B}} \subseteq \underline{r}$ .

5.  $r \dagger_A = r$ .

### Vektorhalmaz-vetítés

$\underline{A}$  és  $\underline{B}$  mono-halmazvektorok, valamint  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{A} \rangle$  vektorhalmaz esetén az

$$\mathbf{R}\langle \underline{A} \rangle \parallel_B \text{ és } \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}} \text{ vetületek szintén vektorhalmazok,}$$

ahol  $B$  a  $\underline{B}$  halmazvektor alaphalmaza.

#### Tulajdonságok, kapcsolat az alvektorhalmazzal és a vektorvetítéssel

1.  $\mathbf{R}\langle \underline{A} \rangle \parallel_B$  és  $\mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}}$  akkor is lehet valódi multihalmaz, ha  $\mathbf{R}\langle \underline{A} \rangle$  monohalmaz.
2.  $\mathbf{R}\langle \underline{A} \rangle \parallel_B = \{ r \dagger_B \mid r \in \mathbf{R}\langle \underline{A} \rangle \}$ , és  $\mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}} = \{ r \dagger_{\underline{B}} \mid r \in \mathbf{R}\langle \underline{A} \rangle \}$ .
3.  $\mu(\mathbf{R}\langle \underline{A} \rangle \parallel_B) = \mu(\mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}}) = \mu(\mathbf{R}\langle \underline{A} \rangle)$ .
4.  $\mathbf{R}\langle \underline{A} \rangle \parallel_A = \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{A}} = \mathbf{R}\langle \underline{A} \rangle$ .
5.  $\mathbf{R}\langle \underline{A} \rangle \parallel_B \stackrel{\cong}{=} \mathbf{R}\langle \underline{A} \rangle$ .
6.  $\mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}} \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{A} \dagger_{\underline{B}} \rangle$ .

### Hasonlósági vetítés

$\underline{A}$  és  $\underline{B}$  mono-halmazvektorok, melyekre  $\underline{A} \dagger_{\underline{B}} \neq \emptyset$ , továbbá  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{A} \rangle$  és  $\mathbf{S}\langle \underline{B} \rangle \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{B} \rangle$  vektorhalmazok esetén az

$$\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{S}\langle \underline{B} \rangle} \text{ szintén vektorhalmaz.}$$

#### Tulajdonságok, kapcsolat a vektorhalmaz-vetítéssel

1.  $\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{S}\langle \underline{B} \rangle} = \{ r \mid r \in \mathbf{R}\langle \underline{A} \rangle, \exists s \in \mathbf{S}\langle \underline{B} \rangle : s \stackrel{\underline{A} \parallel_{\underline{B}}}{\sim} r \}$ .
2.  $\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{R}\langle \underline{A} \rangle} = \mathbf{R}\langle \underline{A} \rangle$ .
3. Ha  $\underline{A} \dagger_{\underline{B}} \neq \emptyset$  és  $\mathbf{Q} = \mathbf{R}\langle \underline{A} \rangle \parallel_{\underline{B}}$ , akkor  

$$\mathbf{R}\langle \underline{A} \rangle \#_{\mathbf{Q}} = \mathbf{R}\langle \underline{A} \rangle.$$

### Transzvertált vetítés, transzvertálás

$\underline{A}$  és  $\underline{B}$  mono-halmazvektorok, ezek  $A$  és  $B$  alaphalmazai, és  $\mathbf{R}\langle \underline{A} \rangle \stackrel{\mathbf{R}}{\subseteq} \mathbf{D}\langle \underline{A} \rangle$  vektorhalmaz esetén az

$$\mathbf{R}\langle \underline{A} \rangle \#_B \text{ és } \mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}} \text{ vetületek halmazvektorok (és nem vektorhalmazok), továbbá}$$

$$\mathbf{R}^\# \langle \underline{A} \rangle \triangleq \mathbf{R}\langle \underline{A} \rangle \#_A.$$

#### Tulajdonságok, kapcsolat a résztartománnyal

1.  $\mu(\mathbf{R}\langle \underline{A} \rangle \#_B) = \mu(\mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}}) = \mu(A|_B)$ .
2. Minden  $i \in \text{Ind}(\underline{A} \dagger_B)$  esetén

- 2.1.  $\mathbf{R}\langle \underline{A} \rangle \#_B(i) = \bigcup_{r \in \mathbf{R}\langle \underline{A} \rangle} \{r(A_i)\}$ , ahol  $A_i = \underline{A} \upharpoonright_B(i)$ , és
- 2.2.  $\mathbf{R}\langle \underline{A} \rangle \#_{\underline{B}}(i) = \bigcup_{r \in \mathbf{R}\langle \underline{A} \rangle} \{r(A_i)\}$ , ahol  $A_i = \underline{A} \upharpoonright_{\underline{B}}(i)$ .
3.  $\mathbf{R}\langle \underline{A} \rangle \#_A = \mathbf{R}\langle \underline{A} \rangle \#_{\underline{A}} \stackrel{*}{\subseteq} \underline{A}$ .
4.  $\mu(\mathbf{R}\langle \underline{A} \rangle \#_A) = \mu(\mathbf{R}\langle \underline{A} \rangle \#_{\underline{A}}) = \mu(\underline{A})$ .
5.  $\mathbf{R}\langle \underline{A} \rangle \#_B \stackrel{*}{\subseteq} \mathbf{R}\langle \underline{A} \rangle \#_A$ .
6.  $\mathbf{D}\langle \underline{A} \rangle \#_A = \underline{A}$ .

## Hasonlóságok és tulajdonságaik (összefoglaló)

Az alábbiakban áttekintjük az eddig bevezetett hasonlósági kapcsolatokat, valamint ezek alapvető tulajdonságait. Mivel az ekvivalencia is egy speciális hasonlóság, ezért az áttekintést a vektorok ekvivalenciájának bemutatásával kezdjük.

### Vektor-ekvivalencia

$\underline{a}$  és  $\underline{b}$  vektorok esetén

$$\underline{a} \equiv \underline{b}, \text{ ha } \underline{a}^{\text{norm}} = \underline{b}^{\text{norm}},$$

ahol például az  $\underline{a}^{\text{norm}}$  értelmezése:

$$\text{minden } i \in I_{\mu(\underline{a})} \text{ esetén } \underline{a}^{\text{norm}}(i) = \underline{a}[i],$$

$$\text{és így } \text{Ind}(\underline{a}^{\text{norm}}) = I_{\mu(\underline{a})}.$$

### Tulajdonságok, kapcsolat az alaphalmazzal

1.  $\underline{a} \equiv \underline{b} \models \text{Comp}(\underline{a}) = \text{Comp}(\underline{b})$ .
2.  $\underline{a} \equiv \underline{b} \models a = b$ .
3.  $\underline{a} \equiv \underline{a}$
4.  $\underline{a} \equiv \underline{b} \models \underline{b} \equiv \underline{a}$ .
5.  $\underline{a} \equiv \underline{b}, \underline{b} \equiv \underline{c} \models \underline{a} \equiv \underline{c}$ .

### Vektorhasonlóság

$\underline{a}$  és  $\underline{b}$  vektorok esetén

$$\underline{a} \sim \underline{b}, \text{ ha van olyan } \underline{c} \text{ vektor, melyre } \underline{c} \subseteq \underline{a}, \text{ és } \underline{c} \subseteq \underline{b}.$$

### Tulajdonságok, kapcsolat az alvektorral és a vektorvetítéssel

1.  $\underline{a} \sim \underline{a}$ .
2.  $\underline{a} \sim \underline{b} \models \underline{b} \sim \underline{a}$ .
3.  $\underline{b} \subseteq \underline{a} \models \underline{b} \sim \underline{a}$ .
4.  $\underline{a} \upharpoonright_b \sim \underline{a}$ .

## Rekordhasonlóság

$r$  és  $s$  rekordok, valamint  $B$  halmazvektor esetén

$$r \stackrel{B}{\sim} s, \text{ ha } r \upharpoonright_B = s \upharpoonright_B.$$

$$r \stackrel{B}{\approx} s, \text{ ha } r \upharpoonright_{\bar{B}} = s \upharpoonright_{\bar{B}}.$$

### Tulajdonságok, kapcsolat a vektorhasonlósággal

1.  $r \stackrel{B}{\sim} s$ , ha minden  $A_i \in A|_B$  esetén  $r(A_i) = s(A_i)$ .
2.  $r \stackrel{B}{\sim} s \models r \stackrel{B}{\approx} s$ .
3.  $r \stackrel{B}{\sim} s \models r \sim s$ .

## Vektorhalmaz-hasonlóság

$A$ ,  $B$  és  $C$  halmazvektorok, melyekre  $A \upharpoonright_B \neq \emptyset$  és  $(A \upharpoonright_B) \upharpoonright_C \neq \emptyset$ .  $R\langle A \rangle \subseteq D\langle A \rangle$  és  $S\langle B \rangle \subseteq D\langle B \rangle$  vektorhalmazok esetén

$$R\langle A \rangle \stackrel{C}{\approx} S\langle B \rangle,$$

ha minden  $r \in R\langle A \rangle$  és  $s \in S\langle B \rangle$  esetén  $r \stackrel{C}{\approx} s$ .

### Tulajdonságok, kapcsolat a vektorhalmaz-vetítéssel és az alvektorhalmazzal

1.  $R\langle A \rangle \stackrel{C}{\approx} R\langle A \rangle$ .
2.  $R\langle A \rangle \stackrel{C}{\approx} S\langle B \rangle \models S\langle B \rangle \stackrel{C}{\approx} R\langle A \rangle$ .
3.  $S\langle B \rangle = R\langle A \rangle|_C \models S\langle B \rangle \stackrel{C}{\approx} R\langle A \rangle$ .
4.  $S\langle B \rangle \stackrel{C}{\approx} R\langle A \rangle \models S\langle B \rangle \stackrel{C}{\approx} R\langle A \rangle$ .

# A relációalgebra alapfogalmai

## Relációséma, attribútum

Legyen  $A = \{A_1, \dots, A_n\}$  egy mono-hiperhalmaz (elemei tehát monohalmazok), valamint tekintsük elemeinek egy  $\underline{A} = \langle A_1, \dots, A_n \rangle$  elrendezését, vagyis egy  $\underline{A}$  halmazvektorát. Ekkor az  $A$ -t *attribútumhalmaznak* (tulajdonsághalmaznak), elemeit *attribútumoknak*, valamely  $a \in A_i$  elemet ( $i \in I_n$ ) az  $A_i$  *attribútum egy értékének*, továbbá  $\underline{A}$ -t *attribútumvektornak* nevezük. Ezek után az  $\underline{A}$  attribútumvektor fölötti Descartes-szorzat valamely

$$\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$$

részhalmazát az  $\underline{A}$  attribútumvektor fölötti *relációsémának* nevezzük.

Mivel az üres halmaz minden halmaznak részhalmaz, így a definíció szerint az lehet relációséma is. Az üres halmazt, mint relációsémát *triviális relációsémának* nevezzük.

*Megjegyzés*

1. Algebrailag a relációsémát *attribútumvektor fölött* értelmezhetjük csak, de fogjuk használni az „*attribútumhalmazon értelmezett relációséma*” kifejezést is. Természetesen a relációsémát ekkor is az attribútumhalmaz valamely vektorára vonatkoztatjuk.
2. A relációséma tehát egy vektorhalmaz, mégpedig egy *rektanguláris mono-vektorhalmaz*, hiszen a mono-hiperhalmazon értelmezett Descartes-szorzat is az, és a relációséma nem más, mint a Descartes-szorzat egyszerű részhalmaz.

## Relációséma keretei (értéktartományai)

Mivel egy  $\mathbf{R}(\underline{A})$  relációséma nem más, mint egy  $\underline{A}$  attribútumvektor fölött (tehát a Descartes-szorzaton keresztül) értelmezett vektorhalmaz, így rá definiálhatjuk a, valamely  $\underline{B}$  halmazvektorra vonatkozó  *$\underline{B}$ -parciális vektorkeret* ( $\underline{vFrame}(\mathbf{R}(\underline{A}), \underline{B})$ ), és  *$\underline{B}$ -parciális halmazkeret* ( $\underline{sFrame}(\mathbf{R}(\underline{A}), \underline{B})$ ) fogalmakat, illetve a *teljes vektorkeret* és *teljes halmazkeret* fogalmakat, mégpedig pontosan az előző alfejezet végén, a vektorhalmazokra bevezett módon.

Nyilván minden  $\mathbf{R}(\underline{A})$  relációséma teljes vektorkeretére és teljes halmazkeretére fennáll:

1.  $\underline{vFrame}(\mathbf{R}(\underline{A}), \underline{A}) \stackrel{*}{=} \underline{A}$ ,
2.  $\underline{vFrame}(\mathbf{D}(\underline{A}), \underline{A}) = \underline{A}$ , továbbá
3.  $\underline{sFrame}(\mathbf{R}(\underline{A}), \underline{A}) = \mathbf{R}(\underline{A})$ .

*Megjegyzés*

Egy relációséma teljes vektorkerete (teljes vektor-értéktartománya) tehát az attribútumvektorának egy résztartománya, vagyis egy olyan halmazvektor, mely az attribútumok azon részhalmazait tartalmazza, melyeknek elemei a relációséma legalább egy rekordjában előfordulnak. A fentiek szemléltetéséhez tekintsük az előző alfejezet végén szereplő példát.

## A relációséma értelmezése

Először tekintsük a Descartes-szorzatot. A  $\mathbf{D}\langle A \rangle$  Descartes-szorzat gyakorlati értelmezésénél abból indulunk ki, hogy  $\mathbf{D}\langle A \rangle$  az  $A$ -beli attribútumok mindegyikének teljes értékészletét lefedi, azt is mondhatjuk, hogy  $\mathbf{D}\langle A \rangle$  az  $n$ -dimenziós attribútum-tér minden pontját tartalmazza. Ha az attribútumokat *tulajdonságtípusoknak* tekintjük, akkor azt mondhatjuk, hogy a  $\mathbf{D}\langle A \rangle$  tér egy  $\langle a_1, a_2, \dots, a_n \rangle$  vektorral kijelölt pontja egy olyan objektumot reprezentál, mely az  $A_1 \in A$  tulajdonságtípuson (például SZIN) az  $a_1 \in A_1$  tulajdonságértéket vette fel (például „tűzpiros”), az  $A_2$  tulajdonságtípuson (például TÍPUS) az  $a_2 \in A_2$  tulajdonságértéket vette fel (például „Ferrari”), stb. Tehát  $\mathbf{D}\langle A \rangle$  valamely objektum összes, elvileg lehetséges előfordulását tartalmazza, vagyis  $\mathbf{D}\langle A \rangle$  az attribútumok értékészlete által biztosított összes objektumok halmaza. Ily módon tehát azt is mondhatjuk, hogy egy  $\mathbf{D}\langle A \rangle$  Descartes-szorzat egy *objektumtípus*, melynek egyes előfordulásait – az *objektum-előfordulásokat*, vagy csak egyszerűen az *objektumokat* – az egyes  $A_i \in A$  tulajdonságtípusok konkrét  $a_i \in A_i$  értékei, az *attribútum-előfordulások*, vagy más néven *tulajdonság-előfordulások* együttesen határoznak meg.

Az  $\mathbf{R}\langle A \rangle$  relációséma az  $A$  attribútumvektor fölötti Descartes-szorzat egy olyan részhalma, melyet a később definiálandó *függőségek*, mint  $\mathbf{D}\langle A \rangle$ -beli megszorítások jelölnek ki. (Függőségeknek majd az olyan – az attribútumhalmazok közötti – függvényszerű leképezéseket fogjuk nevezni, melyek kifejezik, az attribútumok közötti kapcsolatokat.) A relációséma a Descartes-szorzatnak egy olyan szűkítése tehát, mely az attribútumhalmaz értéktartományára által lehetővé tett, elvben előforduló összes objektumhoz képest csak a gyakorlati megfontolások alapján megadott függőségek által megengedettek tartalmazza.

Megjegyezzük még, hogy mivel az  $\mathbf{R}\langle A \rangle$  relációséma az  $A$  attribútumhalmazon definiálható összes megengedett objektumot tartalmazza tulajdonságkijelölés (minden  $A_i \in A$  értékészlete) és a később ismertetendő függőségek alapján, tehát reprezentálja ezek kijelöléséhez szükséges minimális információt, ezért  $\mathbf{R}\langle A \rangle$ -t *objektummodellnek*, vagy *objektumspecifikációnak* is tekintjük egy adatbázis logikai tervezésében (például  $\mathbf{R}\langle A \rangle = \mathbf{gépko-} \mathbf{csi}\langle \text{SZIN, TÍPUS, ÉVJÁRAT} \rangle$ ). Ebben az esetben az  $\mathbf{R}\langle A \rangle$  egy *jelöléssé*, az  $A$  attribútumhalmazon *megengedett összes objektum gyűjtőfogalmává* egyszerűsödik, amely jelölésben a fontos információt az hordozza, hogy

- mely attribútumokat tartalmaz  $A$ ,
- az egyes attribútumok milyen típusúak (mely értéktartományból vehetnek fel értékeket),
- mi az egyes attribútumok sorrendje az  $A$  attribútumvektorban,
- mi az attribútumok kapcsolata egymással (ezeket fogják a függőségek meghatározni), és
- milyen egyéb speciális tulajdonságokkal rendelkeznek még az egyes attribútumok, illetve maga a relációséma. (Ezeket majd megszorításoknak nevezzük, és fontos szerepük lesz az adatbázis egyes relációi közötti kapcsolatok leírásában, de a reláció-algebrai tárgyalásban nincs jelentőségük).

Végül tekintsünk egy gyakorlati példát. Nem valószínű például, hogy lehet fekete, vagy rózsaszínű Ferrarival találkozni, tehát míg a  $\langle \text{„rózsaszínű”, „Ferrari”, 2000} \rangle$  rekord nyilván eleme a  $\langle \text{SZIN, TÍPUS, ÉVJÁRAT} \rangle$  attribútumvektoron értelmezett Descartes-szorzatnak, addig nem eleme a gyakorlatban megengedett adatokat tartalmazó  $\mathbf{gépko-} \mathbf{csi}\langle \text{SZIN, TÍPUS, ÉVJÁRAT} \rangle$  relációsémának. (Ezt a tiltást például kifejezheti a TÍPUS attribútumot a SZIN attribútumra leképező függőség.)

## Reláció

Legyen  $\underline{A} = \langle A_1, \dots, A_n \rangle$  egy attribútumvektor, valamint  $\mathbf{R}(\underline{A})$  egy ezen értelmezett relációséma. Legyen továbbá  $R(\underline{A})$  egy multihalmaz, melyre

$$\text{Reduc}(R(\underline{A})) \subseteq \mathbf{R}(\underline{A}).$$

Ekkor az  $R(\underline{A})$ -t az  $\underline{A}$  attribútumvektor fölötti relációnak, vagy az  $\mathbf{R}(\underline{A})$  relációséma egy relációjának nevezzük. Az  $R(\underline{A})$  relációra használni fogjuk még az  $R\langle A_1, \dots, A_n \rangle$  jelölést is. Ha  $\mu(A) = n$ , akkor  $R(\underline{A})$ -t  $n$ -edfokú relációnak, és speciálisan  $\mu(A) = 2$  esetén bináris relációnak nevezzük. Láthatóan a  $\mathbf{D}(\underline{A})$  Descartes-szorzat és az  $\mathbf{R}(\underline{A})$  relációséma is reláció.

Mivel az  $R(\underline{A})$  reláció egy multihalmaz, ezért *multirelációnak* is nevezzük, és speciális esetben *valódi multirelációnak*, ha  $R(\underline{A})$  valódi multihalmaz, illetve *monorelációnak*, ha  $R(\underline{A})$  monohalmaz. A hagyományos algebrában bevezetett reláció tehát egy monoreláció, és ha  $R(\underline{A})$  egy monoreláció, akkor nyilván  $R(\underline{A}) \subseteq \mathbf{R}(\underline{A})$ .

A relációsémánál tett megfontolásokhoz hasonló okokból bevezetjük a *triviális reláció* fogalmát, mely nem más, mint az üres halmaz.

*Megjegyzés*

1. A fenti definíciót nyilván

$$R(\underline{A}) \subseteq^R \mathbf{R}(\underline{A})$$

alakban is felírhattuk volna, tehát az  $R(\underline{A})$  reláció redukált részhalmaza az  $\mathbf{R}(\underline{A})$  relációsémának, továbbá nyilván  $R(\underline{A}) \subseteq^R \mathbf{D}(\underline{A})$  is fennáll.

2. Nyilván a relációsémához hasonlóan a reláció is egy vektorhalmaz, mégpedig egy *rektanguláris multi-vektorhalmaz*.
3. A relációsémánál tett megjegyzések értelmében a reláció esetén is használni fogjuk az *attribútumvektor fölött értelmezett* kifejezés helyett az *attribútumhalmazon értelmezett* kifejezést, de természetesen a relációt ekkor is az attribútumhalmaz valamely vektorára vonatkoztatjuk.

## Reláció keretei

A relációsémához hasonló módon értelmezzük egy  $R(\underline{A})$  reláció különböző típusú értéktartományait, vagy kereteit. Az ezekre vonatkozó definíciókból láthatóan nem okoz problémát az, hogy a reláció egy multi-vektorhalmaz.

Bármely  $\underline{A}$  és  $\underline{B}$  attribútumvektor esetén egy  $\mathbf{R}(\underline{A})$  relációséma minden  $R(\underline{A})$  relációjára (ahol tehát  $R(\underline{A}) \subseteq^R \mathbf{R}(\underline{A})$ ) nyilván fennáll, hogy  $\underline{vFrame}(R(\underline{A}), \underline{B}) \subseteq^* \underline{vFrame}(\mathbf{R}(\underline{A}), \underline{B})$ .

*Megjegyzés*

Egy reláció teljes vektorkerete (teljes vektor-értéktartománya) tehát – hasonlóan a relációsémához – az attribútumvektorának egy résztartománya, vagyis egy olyan halmazvektor, mely az attribútumok azon részhalmazait tartalmazza, melyeknek elemei a reláció legalább egy rekordjában előfordulnak.

**Példa**

Tekintsük az előző alfejezet végén lévő példát, ám ezúttal a vizsgált vektorhalmaz legyen az abban szereplő  $\mathbf{R}(\underline{A})$  relációséma  $R(\underline{A}) = \{\langle 3, a, x \rangle, \langle 3, b, x \rangle, \langle 3, a, x \rangle\}$  relációja, mely láthatóan egy valódi multireláció. Ekkor nyilván  $\underline{vFrame}(R(\underline{A}), \underline{A}) = \langle \{3\}, \{a, b\}, \{x\} \rangle$  és  $\underline{vFrame}(R(\underline{A}), \underline{A}) \subseteq^* \underline{vFrame}(\mathbf{R}(\underline{A}), \underline{A})$ .

**A reláció értelmezése**

A reláció értelmezésénél abból indulunk ki, hogy a relációséma maga is egy reláció, mégpedig az összes engedélyezett különböző objektumok halmaza. Egy reláció a relációsémához képest szűkebb abban az értelemben, hogy az összes engedélyezett különböző objektumoknak csak egy részét tartalmazza, másrészt bővebb, mivel egy objektum többször is előfordulhat benne.

Ez egyrészt azt jelenti, hogy az  $\mathbf{R}(\underline{A})$  relációsémával kijelölt objektumhalmaznak csak egy töredéke kerül be valamely ténylegesen létező  $R(\underline{A})$  relációba. Például nyilván engedélyezett a  $\langle \text{„tűzpiros”, „Ferrari”, 1964} \rangle$  rekord és így eleme a **gépkocsi**(SZÍN, TÍPUS, ÉVJÁRAT) relációsémának, ám egy konkrét autószaalon valószínűleg nem tart ilyet a raktáron, így e rekord nem eleme az adatbázisában lévő *gépkocsi*(SZÍN, TÍPUS, ÉVJÁRAT) relációjának.

Másrészt a fenti  $R(\underline{A})$  relációban lehet három olyan objektum, melynek a teljes attribútumhalmazon vett leírása  $\langle \text{„tűzpiros”, „Ferrari”, „2000”} \rangle$ . Ez a három „azonos” objektum természetesen három különböző gépkocsit jelez (például az alvázszámuk is más), csakhogy az adott  $\mathbf{R}(\underline{A})$  relációséma több attribútumot nem tartalmaz, vagyis az  $A$  attribútumhalmaz által biztosított absztrakciós szinten a három objektum azonos. (Ilyen eset adódhat például egy lekérdezés során, ahol a felhasználó jelöli ki a számára fontos attribútumokat. Ekkor a származtatott relációban lehetnek azonos objektumok akkor is, ha az eredetiben nem voltak.) Az azonos objektumok egyébként  $\text{Reduc}(R(\underline{A}))$  módon nyilván kiszűrhetők.

A gyakorlati relációk további fontos tulajdonsága, hogy használatuk során állandóan változik a tartalmuk. Algebrailag így mindig más és más relációról kellene beszélnünk, attól függően, hogy éppen mely objektumok szerepelnek bennük. A gyakorlatban azonban e relációknak csak azok a tulajdonságai érdekelnek minket, amelyek e különböző reláció-előfordulásokban azonosak. Nos, ezt a minimális azonosságot szimbolizálja a relációséma.

A gyakorlati relációknak van még egy fontos tulajdonsága; attribútumai rendelkezhetnek egy speciális értékkel, az úgynevezett NULL-értékkel. Ez azt jelenti, hogy van olyan rekordja, mely egy vagy több attribútum helyén nem rendelkezik értékkel, „kitöltetlen”. Például egy adóhivatali nyilvántartó rendszerben egy új szervezet számlaszáma azért kitöltetlen, mert a számlanyitáshoz éppen az adóhivataltól kell vinni igazolást arról, hogy ott már nyilvántartásba vették. Természetesen vannak attribútumok, melyekre a NULL-érték engedélyezett, és van, amelyekre nem. Az előbbi példában a szervezet megnevezése még átmenetileg sem lehet kitöltetlen.

A relációalgebrai vizsgálatokban a NULL-értéknek nincs jelentősége, egy a sok egyéb érték között, a numerikus, vagy egyéb kifejezésekben azonban egy kifejezés értéke NULL-érték, ha valamelyik attribútuma NULL-értéket tartalmaz.

A továbbiakban a jelölések egyszerűsítése érdekében a konkrét alkalmazásokban a relációsémákat és a relációkat azonos módon jelöljük. Például *gépkocsi*(SZÍN, TÍPUS, ÉVJÁRAT) egyaránt jelöl relációsémát és relációt. Ez nem okoz félreértést, mivel a szövegkörnyezetből mindig egyértelmű, hogy éppen melyikről beszélünk.

## További származtatott fogalmak

### Rekord, mint a reláció eleme

A  $\mathbf{D}(\underline{A})$  Descartes-szorzat definíciójánál az elemeit alkotó vektorokat *rekordoknak* neveztük. Valamely  $\underline{r} \in \mathbf{D}(\underline{A})$  rekord felírható  $\underline{r} = \langle \underline{r}(A_1), \dots, \underline{r}(A_n) \rangle$  vektoros alakban, ahol minden  $A_i \in A$  esetén  $\underline{r}(A_i) \in A_i$ , és az  $\underline{r}(A_i)$  elemet az  $\underline{r}$  rekord  $A_i$  attribútumhoz tartozó komponensének nevezzük. Mivel a rekordok vektorok, így alkalmazhatjuk rá a komponenshalmaz-függvényt, tehát egy  $\underline{r}$  rekord komponenshalmaza a  $\text{Comp}(\underline{r})$  multihalmaz, melyet az  $\underline{A}$  attribútumvektor segítségével

$$\text{Comp}(\underline{r}) \triangleq \{ \underline{r}(A_i) \mid A_i \in A \} \quad (**)$$

módon is definiálhatunk.

#### Megjegyzés

Könnyen belátható, hogy a komponenshalmaz-függvény fenti (\*\*) és a vektoroknál bemutatott (\*) definíciója ekvivalens; figyeljük meg, hogy a (\*\*) definícióban az indexelő-halmazt az attribútumhalmaz helyettesíti (pontosabban az attribútumhalmazban az egyes attribútumokat indexelő-halmaz).

Már többször utaltunk arra, hogy az  $R(\underline{A})$  relációnak a multihalmaz jellege miatt lehetnek azonos rekordjai. Ezek az azonos rekordok a világ olyan (nem feltétlen azonos) egyedeit reprezentálják, melyek a tulajdonságok  $A$  halmazán azonosak, vagyis ezen az absztrakciós szinten nem különböztethetők meg egymástól.

Egy rekord komponenseinek halmaza is lehet valódi multihalmaz, ha a rekord a hozzátartozó különböző attribútumokon azonos értéket vesz fel (például a DOLGOZÓ\_NEVE, és az APJA\_NEVE attribútumokon egyaránt lehet „Kovács Béla”).

### Relációban álló attribútumértékek

Legyen  $A = \{A_1, \dots, A_n\}$  egy attribútumhalmaz,  $R(\underline{A})$  egy ezen értelmezett reláció.

Tetszőleges  $A_i, A_j \in A$  esetén valamely  $a_i \in A_i$  és  $a_j \in A_j$  attribútumértékek  $R(\underline{A})$  relációban állnak egymással, ha van olyan  $\underline{r} \in R(\underline{A})$  rekord, melyre  $\underline{r}(A_i) = a_i$  és  $\underline{r}(A_j) = a_j$ .

### Alséma, alreláció (attribútumhalmazra és attribútumvektorra relatív)

Legyen  $\mathbf{R}(\underline{A})$  egy relációséma az  $\underline{A}$  attribútumvektor fölött,  $R(\underline{A})$  ennek egy relációja (azaz  $R(\underline{A}) \subseteq \mathbf{R}(\underline{A})$ ), továbbá  $\underline{B} \subseteq \underline{A}$ . (Tehát a  $\underline{B}$  attribútumvektor egyszerű alvektora az  $\underline{A}$  attribútumvektornak. Ekkor az ezekhez tartozó  $A$  és  $B$  hiperhalmazokra nyilván fennáll  $B \subseteq A$ ).

Ekkor az  $\mathbf{R}(\underline{A})|_B = \{ \underline{r}|_B \mid \underline{r} \in \mathbf{R}(\underline{A}) \}$  relációsémát az  $\mathbf{R}(\underline{A})$   $B$ -alsémájának, és az  $R(\underline{A})|_B = \{ \underline{r}|_B \mid \underline{r} \in R(\underline{A}) \}$  relációt az  $R(\underline{A})$   $B$ -alrelációjának nevezzük. Nyilván  $\mathbf{R}(\underline{A})|_B$  egy relációséma, és  $R(\underline{A})|_B$  egy reláció a  $B$  attribútumhalmaz felett.

A fenti attribútumhalmazra relatív definíciók értelemszerű módosításával lehet definiálni egy relációséma vagy egy reláció attribútumvektorra vonatkozó alsémáját vagy alrelációját.

**Példa**

Tekintsük az  $R(\underline{A}) = \text{gépkocsi}(\langle \text{SZIN}, \text{TÍPUS}, \text{ÉVJÁRAT} \rangle)$  relációt és a  $\underline{B} = \langle \text{TÍPUS}, \text{SZIN} \rangle$  attribútumvektort. Ekkor egyrészt  $R(\underline{A})|_{\underline{B}} = \text{gépkocsi\_I}(\langle \text{SZIN}, \text{TÍPUS} \rangle)$  és valamely  $\underline{r} = \langle \text{„tűzpiros”}, \text{„Ferrari”}, 2000 \rangle$  rekord esetén  $\underline{r}|_{\underline{B}} = \langle \text{„tűzpiros”}, \text{„Ferrari”} \rangle$ , másrészt  $R(\underline{A})|_{\underline{B}} = \text{gépkocsi\_2}(\langle \text{TÍPUS}, \text{SZIN} \rangle)$  és  $\underline{r}|_{\underline{B}} = \langle \text{„Ferrari”}, \text{„tűzpiros”} \rangle$ .

**Reláció, mint tábla**

Egy  $R(\underline{A})$  relációt a gyakorlatban egy olyan  $R_{\underline{A}}$  jelű táblaként szoktunk megadni, melyben az  $\underline{r} \in R(\underline{A})$  rekordok alkotják a tábla  $R_{\underline{A}}[\underline{r}]$  módon jelölt sorait, és az összes  $\underline{r} \in R(\underline{A})$  rekord azonos  $A_i$  attribútumhoz ( $A_i \in \underline{A}$ ) tartozó  $\underline{r}(A_i)$  attribútum-értékei alkotják a tábla  $A_i$ -hez tartozó oszlopát, melyet  $R_{\underline{A}}[A_i]$  módon jelölünk. (A táblák sorainak és oszlopainak jelölésében használt szögletes zárójelek a táblák *mátrix* jellegére utalnak.)

*Megjegyzés*

Egy  $R_{\underline{A}}$  táblában (a definíció szerint) az oszlopok sorrendje kötött (a hozzájuk tartozó attribútumok  $\underline{A}$  vektorbeli sorrendjének megfelelően), ám a sorok sorrendje nem meghatározott abból következően, hogy a táblákat a halmaz jellegű relációk egy reprezentációjának tekintjük, melyben pedig az elemek sorrendje meghatározatlan.

A gyakorlatban a táblák sorainak (a relációk rekordjainak) sorrendje természetesen kötött. Érthető, hiszen a memóriában a sorok adatai valamilyen sorrendben követik egymást. Azért, hogy az adattáblákat relációként (vagyis a sorok rendezetlen halmazaként) értelmezhesük (a viszonylag egyszerű matematikai eszközök, és nyelvi leírás kedvéért) nagy árat fizetünk.

Az még a kisebbik baj, hogy úgy kell tennünk, mintha nem tudnánk egy olyan információról (a sorok sorrendjéről), amiről persze tudunk. Példa erre a rendezés műveletének használata. Mivel egy tábla elvben rendezetlen, így az SQL nyelv `SELECT` kiválasztási utasításában elvben akkor is ki kellene jelölnünk egy rendezést, ha tudjuk, hogy a tábla sorai abban a sorrendben vannak fizikailag rendezve. Természetesen a gyakorlati adatbáziskezelő szoftverek mind ismerik a „rendezett” tábla fogalmát, így a felhasználónak nem kell például minden alkalommal ugyanazt a rendezést kiadni (elvégeztetni).

Nagyobb baj az, hogy időnként valóban szükségünk lenne egy sor tényleges fizikai sor-számára. Látni fogjuk, hogy lesz olyan tábla-művelet (relációalgebrai művelet), mely egy adott táblából úgy állítja elő az eredménytáblát, hogy egyes oszlopokat elhagy belőle. Erre szükség lehet vagy egy kimutatás miatt, vagy azért, mert később egy olyan tábla-transzformációt szeretnénk elvégezni, amely lényegesen gyorsabban hajtódik végre, ha a tábla mérete kisebb. Mi van azonban akkor, ha az elhagyott oszlopok között voltak olyanok is, amelyek az egyes sorok egyértelmű megkülönböztetését biztosították? Nem az a baj, hogy ily módon a táblában azonos sorok keletkeztek (végül is ezért vezettük be a multirelációt). A problémát az okozza, ha vissza szeretnénk nyúlni az eredeti táblához, ha szeretnénk tudni, hogy valamelyik „azonos sor” melyik is valójában az eredeti táblában. E probléma sem relációalgebrailag, sem pedig (ebből következően) az SQL nyelvben nem oldható meg. A gyakorlati adatbáziskezelő szoftverek természetesen ezt a problémát is megoldják egy úgynevezett sorazonosító segítségével (az Oracle esetében ez a ROWID).

Összefoglalóan megállapíthatjuk, hogy a relációkat táblákkal reprezentáljuk a gyakorlatban. Ám, ha azt kérdezzük, hogy azokat a számítástechnikai objektumokat, melyeket táblának (rekordtípusú tömbnek) nevezünk, vajon megfelelő módon interpretáljuk-e (modelliz-

zük-e) a reláció fogalmával, akkor azt a választ kell adjuk, hogy nem! Innen közelítve a modellválasztást súlyos információvesztést láthatunk. Még a multireláció esetén is olyan fontos információ veszik el, mint a sorok rendezettsége, ha pedig a klasszikus relációalgebrai tárgyalás monorelációját tekintjük, akkor még az azonos sorok tárolásának képessége is.

Láthatjuk tehát, hogy az általunk bevezetett matematikai leíró eszköz nem tökéletes. A gyakorlati szoftverek persze a felmerült problémákat mindig megoldják, azonban egységes leíró eszköz, és így szabványos nyelvi eszköz hiányában általában különbözőképpen. Volna ugyan egy lehetőség, hogy a fenti problémákat leíró eszköz szinten korrigáljuk. Ennek az a lényege, hogy az adattáblákat ne egyszerűen vektorok (rekordok) halmazaként, hanem a hipervektorként interpretáljuk, azaz minden relációt egészítsünk ki egy sorszámozó attribútummal. Ez azonban messze vezetne, mi a továbbiakban a szokásos tárgyalásnál maradunk. Akit a felvetett probléma érdekel, gondolkodjon el ezen...

## Altábla

Az  $R\langle \underline{A} \rangle$  reláció valamely  $B \subseteq A$  attribútumhalmazhoz tartozó  $R\langle \underline{A} \rangle \llcorner_B$  alrelációjának  $R_{\underline{A}} \llcorner_B$  jelű táblája az  $R_{\underline{A}}$  táblából oly módon származtatható, hogy abból egyrészt minden  $R_{\underline{A}}[A_i]$  oszlopot ( $A_i \in A \setminus B$ ) elhagyunk, másrészt az oszlopokat átrendezzük a  $B$  vektorbeli sorrendnek megfelelően. Így értelemszerűen az  $R_{\underline{A}} \llcorner_B$  táblát az  $R_{\underline{A}}$  tábla  $\underline{B}$ -altáblájának is nevezzük.

### Példa

Legyen az  $R\langle X, Y, Z \rangle$  reláció táblája az alábbi:

$X$	$Y$	$Z$
1	2	3
6	7	8
9	7	8

Legyen  $S\langle Z, Y \rangle = R\langle X, Y, Z \rangle \llcorner_{\langle Z, Y \rangle}$ , vagyis az  $R\langle X, Y, Z \rangle$  alrelációja. Az  $S\langle Z, Y \rangle$  tábla ekkor:

$Z$	$Y$
3	2
8	7
8	7

Láthatóan az  $S\langle Z, Y \rangle$  egy valódi multireláció.

## Részséma, részreláció, résztábla

Legyen  $\mathbf{R}\langle \underline{A} \rangle$  egy relációséma az  $A$  attribútumhalmaz fölött,  $R\langle \underline{A} \rangle$  ennek egy relációja,  $R_{\underline{A}}$  pedig ez utóbbi táblája. Ekkor valamely  $\mathbf{R}'\langle \underline{A} \rangle \subseteq \mathbf{R}\langle \underline{A} \rangle$  relációsémát az  $\mathbf{R}\langle \underline{A} \rangle$  relációséma *részsémájának*, egy  $R'\langle \underline{A} \rangle \subseteq R\langle \underline{A} \rangle$  relációt az  $\mathbf{R}\langle \underline{A} \rangle$  reláció *részrelációjának*, az  $R_{\underline{A}}$  táblából az egyes sorok elhagyásával nyert  $R_B \subseteq R_{\underline{A}}$  táblát pedig az  $R_{\underline{A}}$  *résztáblájának* nevezzük.

### Megjegyzés

Míg tehát az  $R_{\underline{A}} \llcorner_B$  altáblakiválasztás az  $R_{\underline{A}}$  tábla egyes oszlopainak elhagyását, és a megmaradt oszlopok  $B$  vektornak megfelelő esetleges cseréjével nyert táblát, addig az  $R_B \subseteq R_{\underline{A}}$  résztábla-kiválasztás az  $R_{\underline{A}}$  tábla egyes sorainak elhagyását jelenti.

Az al- és résztábla-kiválasztást az SQL nyelvben a SELECT utasítás teszi lehetővé.

## Relációs adatbázismodell (adatbázisséma), adatbázis

A *relációs adatbázismodell* objektumtípusoknak relációsémájukkal adott véges halmaza. (Ezt a későbbiekben még kiegészítjük a relációsémához tartozó függőségek halmazával).

A reláció tábla-felfogásában egy adatbázis nem más, mint az adatbázismodellben szereplő objektumtípusokhoz tartozó táblák halmaza.

#### *Megjegyzés*

Természetesen valamilyen praktikus szempont mindig összefogja az egy adatbázis-modellbe tartozó relációsémákat, hiszen a gyakorlatban minden relációséma az objektumok valamely azonos leírású halmazát, azaz egy *objektumtípust* reprezentál. (Például egy vállalat adatbázisában egy-egy tábla reprezentálhatja a beszállítókat, a raktárkészletet és a kiadott rendeléseket).

A kereskedelmi adatbázis-kezelő szoftverekben a relációk multihalmazok, vagyis az ezeket reprezentáló táblákban ugyanaz a sor többször is előfordulhat. Ez persze nem azt jelenti, hogy megengedett ugyanazt az objektumot többször is bevinni a nyilvántartásba (bár ezt sem maguk a szoftverek akadályozzák meg, hanem azok a felhasználói programok, melyeket a fejlesztők írnak). Egyszerűen arról van szó, hogy a táblakiválasztási műveletek során (mint a korábbi példákban is) keletkezhetnek azonos sorok. Van ugyan lehetőség ezeknek az automatikus kiszűrésére (például az SQL-ben a `DISTINCT` kulcsszóval), azonban ezzel jelentősen lelassulnak a kiválasztások.

## Leképezések és függvények

A leképezés a reláció után a legfontosabb fogalom az algebrában, sőt az egész matematikában, bár többnyire speciális alakjaiban *függvényként*, illetve *műveletként* használjuk. Mivel a leképezés maga is reláció, így esetenként *relációleképezésnek* is nevezzük.

Mivel az adatbázis-kezelés során igen összetett algebrai szerkezeteket, úgynevezett *strukturákat* használunk, ezért a bevezetésre kerülő fogalmak bonyolultabbak, mint a hagyományos megfelelőik. Ezek nélkül azonban sem a relációalgebrát, sem a függőségeket nem tudnánk egzakt módon bemutatni.

A leképezések lehetnek egyszerűek és összetettek. Az *egyszerűek* maguk a hagyományos leképezések, melyek egyszerű (nem összetett) elemeket képeznek le egyszerű elemek halmazába. Bár ezek számunkra általában „túl egyszerűek”, a leképezések legáltalánosabb fogalmait ezeken keresztül mutatjuk be.

Az *összetett leképezések* közül részletesen a *vektorleképezésekkel* foglalkozunk, mivel ezekből könnyen származtathatjuk a hagyományos leképezéseket. A vektorleképezések vektorokat képeznek le vektorok halmazába. A leképezett elem egy  $n$ -változós vektorleképezés esetén lehet például az  $n$ -dimenziós tér egy pontja.

Az *egyéb összetett leképezések* esetén (az előbbi szemléletes hasonlatnál maradva) vagy különböző terek egy-egy pontjából alkotott vektort (tehát egy hipervektort) képezünk le, vagy egy  $n$ -dimenziós tér pontjainak valamely halmazát (tehát egy alterét, mely egy vektorhalmaz), vagy egészen általános esetben különböző terek altereinek rendezett halmazát (vagyis vektorhalmazok rendezett halmazát). Ne gondoljuk, hogy tárgyalásunkban eltúlozzuk az általánosítást! Az imént említett legáltalánosabb leképezések az adatbázis-kezelésben jól ismert és gyakran használt, úgynevezett *reláció-műveletekként* jelennek meg.

A fentiekén túl nyilván egyéb (akár még jobban összetett) leképezések is definiálhatók, sőt az általunk bevezetett összetett leképezések elemei is lehetnek összetettek.

### Egyszerű leképezés

Legyen  $X$  és  $Y$  két halmaz. Ekkor az

$$F \subseteq X \times Y$$

relációt *egyszerű leképezésnek* nevezzük, és

$$F : X \rightarrow Y$$

módon jelöljük, az  $X$  és az  $Y$  halmazokat az  $F$  *egyszerű leképezés alaphalmazainak*, az  $f \in F$  elemeket pedig az  $F$  *leképezés elemi leképezéseinek* nevezzük, és

$$f = \langle x, y \rangle, \text{ illetve}$$

$$f : x \mapsto y$$

módon is jelöljük, ahol természetesen  $x \in X$  és  $y \in Y$ .

Mivel a leképezés multihalmaz, ezért *multileképezésnek* is nevezzük, és speciális esetben *valódi multileképezésnek*, ha valódi multihalmaz, illetve *monoleképezésnek*, ha monohalmaz. A hagyományos algebrában bevezetett leképezés tehát egy monoleképezés. A továbbiakban leképezés alatt – hacsak mást nem mondunk – multileképezést értünk.

## Egyszerű leképezés vetületei

Legyen  $F : X \rightarrow Y$  egy egyszerű leképezés, és  $f$  ennek egy elemi leképezése (azaz  $f \in F$ ).

1. *Egyszerű leképezés elemi vetületei:*

1.1. Az  $f : x \mapsto y$  elemi leképezés magja:

$$\text{Dom}(f) \triangleq x, \text{ ahol } x \in X.$$

1.2. Az  $f : x \mapsto y$  elemi leképezés képe:

$$\text{Im}(f) \triangleq y, \text{ ahol } y \in Y.$$

2. *Egyszerű leképezés halmazvetületei:*

2.1. Az  $F$  egyszerű leképezés maghalmaza:

$$\text{Dom}(F) \triangleq \{ \text{Dom}(f) \mid f \in F \}.$$

2.2. Az  $F$  egyszerű leképezés képhalmaza:

$$\text{Im}(F) \triangleq \{ \text{Im}(f) \mid f \in F \}.$$

Egy leképezés maghalmazának elemeit a *leképezés magelemeinek*, képhalmazának elemeit a *leképezés képelemeinek* is nevezzük. Egy egyszerű leképezésben szereplő magelemet a leképezés *argumentumának* is nevezünk.

*Megjegyzés*

Figyeljünk fel arra, hogy a definíció szerint a leképezés halmazvetületei multihalmazok, elemi vetületei pedig a leképezés halmazvetületeinek elemei.

## Elem képhalmaza (elem-leképezés)

Legyen  $F : X \rightarrow Y$  egy egyszerű leképezés. Ha  $x \in \text{Dom}(F)$ , akkor az  $x$  magelem képhalmaza:

$$F[x] \triangleq \bigcup_{\substack{f \in F, \\ \text{Dom}(f) = x}} \{ \text{Im}(f) \}.$$

Természetesen amennyiben  $x \notin \text{Dom}(F)$ , akkor az  $F$  leképezés nincs értelmezve, és így az  $x$  magelem  $F[x]$  képe sem.

### Példa

Legyen  $X = \{1, 2, 3, 4, 5, 6, 7, 8\}$ ,  $Y = \{2, 3, 4, 9\}$ , és  $F : X \rightarrow Y$  egy egyszerű leképezés, mely minden  $X$  halmazbeli számhoz hozzárendeli annak mindazon osztóit, melyek az  $Y$  halmaz elemei. Határozzuk meg az  $F$  leképezés elemi leképezéseit, vetületeit és magelemeinek képhalmazait.

*Megoldás*

Az  $F$  leképezés, mint elemi leképezéseinek halmaza  $F = \{2 \mapsto 2, 3 \mapsto 3, 4 \mapsto 2, 4 \mapsto 4, 6 \mapsto 2, 6 \mapsto 3, 8 \mapsto 2, 8 \mapsto 4\}$ . Például az  $f = 6 \mapsto 2$  elemi leképezés magja  $\text{Dom}(f) = 6$ , és képe  $\text{Im}(f) = 2$ . Az  $F$  leképezés maghalmaza  $\text{Dom}(F) = \{2, 3, 4, 4, 6, 6, 8, 8\}$  és a képhalmaza  $\text{Im}(F) = \{2, 3, 2, 4, 2, 3, 2, 4\}$ . Végül az  $F$  leképezés magelemeinek képhalmazai a következők:  $F[2] = \{2\}$ ,  $F[3] = \{3\}$ ,  $F[4] = \{2, 4\}$ ,  $F[6] = \{2, 3\}$ ,  $F[8] = \{2, 4\}$ . Vegyük észre, hogy a példabeli leképezés halmazvetületei valódi multihalmazok.

## Egyszerű leképezés inverze

Egy  $F : X \rightarrow Y$  egyszerű leképezés inverze az  $F^{-1} : Y \rightarrow X$  egyszerű leképezés, ha

$$F^{-1} = \{ \langle y, x \rangle \mid \langle x, y \rangle \in F \}.$$

Ekkor az  $\underline{f} = \langle x, y \rangle$  elemi leképezés inverzét (ahol tehát  $\underline{f} \in F$ )  $\underline{f}^{-1}$  módon jelöljük, és természetesen  $\underline{f}^{-1} = \langle y, x \rangle$ , továbbá  $\underline{f}^{-1} \in F^{-1}$ .

A fenti definícióból következően minden leképezésnek létezik inverze, továbbá egy leképezés és annak inverze kölcsönösen egyértelmű módon meghatározzák egymást. Nyilvánvalóan fennáll továbbá, hogy egy leképezés magja az inverzének a képe és fordítva.

## A leképezés és a leképezés-vetületek értelmezése

Figyeljünk fel arra, hogy az  $F : X \rightarrow Y$  leképezés magelemei és képelemei között egyik irányban sincs egyértékű kapcsolat. Például megfelelő attribútumhalmaz esetén a  $2 \mapsto b$ , a  $4 \mapsto b$  és a  $2 \mapsto d$  elemi leképezések egyaránt lehetnek ugyanannak az  $F$  leképezésnek az elemei. Ez annak következménye, hogy maga az  $F$  leképezés egy általános reláció, vagyis  $X$ -beli és az  $Y$ -beli elemek tetszőleges párosításban alkothatnak elemi leképezést az  $F$ -ben.

Ezt úgy is mondhatnánk, hogy a leképezés fontos tulajdonsága az egyértékűség hiánya. Egy szemléletes példaként tekintsük azt a leképezést, mely az európai és az amerikai államok közötti diplomáciai kapcsolatokat írja le. Jelölje az európai államok halmazát  $X$ , az amerikaiakét pedig  $Y$ . Ekkor az  $F[\text{MAGYARORSZÁG}]$  jelöli nyilván azoknak az amerikai államoknak a halmazát, amelyekkel Magyarország diplomáciai kapcsolatban áll. Ez a halmaz pedig nyilván nem egyelemű, mint ahogyan a leképezés általában nem egyértékű. Végül természetesen az is előfordulhat, hogy egynémely európai, vagy amerikai országnak egyáltalán nincs diplomáciai kapcsolata a másik földrész egyetlen államával sem. Ez pedig azt jelenti, hogy a leképezés egyes elemekre vonatkozóan nincs értelmezve.

Összefoglalóan tehát azt mondhatjuk, hogy az  $F : X \rightarrow Y$  leképezés egy egyértelmű leképezés az  $X$  halmazról az  $Y$  halmazra abban az értelemben, hogy az  $F$  maghalmazának (vagyis az  $X$  egy részhalmazának) minden elemi rekordjához egyértelműen hozzárendeli az  $F$  képhalmazának (vagyis az  $Y$  egy részhalmazának) valamelyik részhalmazát. Látható tehát, hogy az egyértékűség hiánya nem jelenti az egyértelműség hiányát!

A leképezés-vetületek gyakorlatban legfontosabb tulajdonsága azonban az, hogy segítségével korlátozni tudjuk a leképezés alaphalmazait, hiszen egy leképezésben az alaphalmazoknak csak azon értékei vesznek részt, melyek a leképezésnek, mint relációnak valamelyik rekordjában szerepelnek. Sőt, bár ezek az értékek többféle kombinációban szerepelhetnek, ezek közül csak azok vesznek részt a leképezésben, melyek együttesen és adott sorrendben szerepelnek a leképezés, mint reláció valamelyik rekordjában.

## Egyszerű függvény

Bár már többször hivatkoztunk függvényekre (melyet esetenként transzformációnak is neveztünk), e fontos fogalmat eddig intuitív módon használtuk. A reláció és a leképezés fogalmának bevezetése azonban már lehetővé teszi a függvény pontos definícióját

Az egyszerű leképezés származtatásaként definiált egyszerű függvényeket, bár segédfogalomként gyakran használjuk (számosságfüggvény, indextranszformáció, stb.), azonban az

adatbázis-kezelés témakörének bemutatásához ennél bonyolultabb, összetett függvényekre is szükségünk lesz.

Egy  $F: X \rightarrow Y$  egyszerű leképezést *egyszerű függvénynek*, vagy csak egyszerűen *függvénynek* nevezzük, ha minden  $x \in \text{Dom}(F)$  magelem esetén

$$\mu(\text{Reduc}(F[x])) = 1,$$

és ekkor használjuk a szokásos

$$F(x)$$

jelölést az  $x$  magelem *függvényképére*, ahol tehát

$$\text{Reduc}(F[x]) = \{F(x)\}.$$

Az  $F$  függvény elemeit elemi függvényleképezéseknek *fogjuk nevezni*, és nyilván értelemszerűen alkalmazzuk a leképezésnél bevezetett fogalmakat.

#### Megjegyzés

A fenti definícióból láthatóan nem azt várjuk el egy függvénytől, hogy minden magelemt egyetlen képelembe képezzen le (ellentétben a hagyományos algebrai tárgyalással), csupán azt, hogy a magelemekhez tartozó képhalmazok redukáltjai álljanak egyetlen elemből. Tehát ha egy függvény valamely mageleméhez tartozó képhalmaz több elemet tartalmaz, akkor azok csak azonosak lehetnek.

Figyeljünk fel arra is, hogy egy függvénynek az inverze általában nem függvény.

### BIJEKTÍV FÜGGVÉNY

Egy függvényt *bijektívnek* nevezünk, ha az inverze is függvény.

#### Megjegyzés

A definíció tehát nem zárja ki azt, hogy egy bijektív függvény valódi multifüggvény lehessen. Ebből következően egy bijektív függvény, halmazmagja és halmazképe egyaránt lehet multihalmaz. (Természetesen ha egy függvény mutihalmaz, akkor nyilván a halmazmagja és a halmazképe is az, de a vektorhalmaz-vetítés tulajdonságai miatt – amint azt a leképezéseknél már említettük – lehetséges, hogy egy függvény monohalmaz, a halmazmagja és/vagy halmazképe mégis valódi multihalmaz legyen.)

A definícióból következik, hogy egy bijektív függvény esetén a halmazmag és a halmazkép redukáltjainak elemei között kölcsönösen egyértelmű hozzárendelés áll fenn.

#### Példa

Egy tombolán minden sorsolás két húzásból áll; az első urnából egy nyeremény, a másodikkól egy személy nevét húzzák ki. Tegyük fel, hogy a társaságban mindenkinek más a neve, és minden cédulán más név áll. Az egyes sorsolásokból álló leképezés mikor függvény?

#### Megoldás

Ha minden sorsolás után a húzott cédulákat visszateszik, akkor az egyes sorsolások halmaza nem biztos, hogy függvényt alkot. Ha csupán a személyek nevét teszik vissza, akkor biztos, hogy függvény, és ha egyiket se teszik vissza, akkor pedig bijektív függvény.

Legyen a nyeremények halmaza  $X = \{\text{toll, torta, sapka, bor, pumpa}\}$  a nevek halmaza  $Y = \{\text{Péter, Pál, Éva, Irén}\}$ , végül legyen az  $F: X \rightarrow Y$  leképezés, mint az egyes sorsolások halmaza  $F = \{f_1, f_2, f_3, f_4\}$ , ahol  $f_1: \text{toll} \mapsto \text{Éva}$ ,  $f_2: \text{sapka} \mapsto \text{Pál}$ ,  $f_3: \text{torta} \mapsto \text{Éva}$ ,

$f_4 : \text{bor} \mapsto \text{Péter}$ . Az  $F$  leképezés maghalmaza  $\text{Dom}(F) = \{\text{toll, torta, sapka, bor}\}$ , és képhalmaza  $\text{Im}(F) = \{\text{Éva, Pál, Éva, Péter}\}$ . Láthatóan minden magelem képhalmaza egyelemű (például az  $F$  leképezés "torta" magelemének képhalmaza  $F[\text{torta}] = \{\text{Éva}\}$ ), tehát a sorsolás-leképezés valóban függvény, de nem bijektív, mert az egyes húzások után a személyek nevét tartalmazó cédulát visszatettük (így például az  $F^{-1}[\text{Éva}] = \{\text{toll, torta}\}$  képhalmaz már nem egyelemű).

## Összetett leképezések származtatása az egyszerű leképezésekből

A fentiekben az egyszerű leképezések kapcsán áttekintettük a leképezések legfontosabb általános tulajdonságait. A következőkben összetett leképezéseket mutatunk be, legrészletesebben a vektorleképezést.

Az egyes összetett leképezések fogalmait az egyszerű leképezésnél bevezetett fogalmakból, vagy azok mintájára vezetjük be az alább vázolt gondolatmenet alapján:

Először bemutatjuk az adott leképezést, mint relációt, majd a jelölését. Ezután megállapítjuk, hogy az adott leképezésnek milyen szerkezetűek az elemi leképezései, és milyen objektumok alkotják egy elemi leképezés magját és képét. Ezt követően az elemi leképezésből származtatjuk a vizsgált leképezés halmazvetületeit. Végül értelmezzük a vizsgált leképezés magelemeinek képhalmazát, és a leképezés típusának megfelelő függvényt.

A különböző típusú leképezések azonos jellegű fogalmait az egyszerűség kedvéért azonos módon jelöljük (például a halmazvetületeket a  $\text{Dom}$  névvel). Ez nem okoz félreértést, mert a jelölésben mindig szerepel a vonatkozó leképezés (például  $\text{Dom}(F)$ ).

Az egyes összetett leképezések leírásai a továbbiakban egyre vázlatosabbak lesznek, noha elsősorban ezeket fogjuk használni. Ennek részben az az oka, hogy nem akarunk olyan fogalmakkal foglalkozni (például leképezés inverze), melyeknek az értelmezése a korábbiak alapján egyértelmű, másrészt éppen mert később részletesen foglalkozunk velük, egyéb tulajdonságaikat ráérünk megismerni.

## Vektorleképezés

Legyen  $A$  egy attribútumhalmaz,  $X, Y \subseteq A$ , továbbá legyenek  $\underline{A}$ ,  $\underline{X}$  és  $\underline{Y}$  az  $A$ ,  $X$  és  $Y$  halmazok valamely vektorai. Ekkor az

$$F \subseteq \mathbf{D}(\underline{X}) \times \mathbf{D}(\underline{Y})$$

relációt *vektorleképezésnek* nevezzük, és

$$F : \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$$

módon jelöljük, az  $X$  és az  $Y$  halmazokat ezúttal is az  $F$  *vektorleképezés alaphalmazainak*, az  $f \in F$  elemi leképezést pedig *elemi vektorleképezésnek* nevezzük, és

$$f = \langle \underline{x}, \underline{y} \rangle, \text{ illetve}$$

$$f : \underline{x} \mapsto \underline{y}$$

módon is jelöljük, ahol természetesen  $\underline{x} \in \mathbf{D}(\underline{X})$  és  $\underline{y} \in \mathbf{D}(\underline{Y})$ .

A továbbiakban az  $X$  attribútumhalmaz elemeit *magattribútumoknak*, az  $Y$  attribútumhalmaz elemeit pedig *képpattribútumoknak* is fogjuk nevezni.

*Megjegyzés*

Vegyük észre, hogy a definíció megengedi azt, hogy az  $X$  és az  $Y$  attribútumhalmazoknak legyenek közös elemeik! A matematikában nem szoktunk élni ezzel a lehetőséggel, az adatbázis-tervezésben azonban ilyen módon visszük át egy leképezés származtatásán azokat az attribútumokat, melyek egyértelműen azonosítják egy reláció rekordjait (gondoljunk például a vállalkozások adóazonosítójára).

**A vektorleképezés vetületei**

Legyen  $F : \mathbf{D}\langle X \rangle \rightarrow \mathbf{D}\langle Y \rangle$  egy vektorleképezés, és  $f$  ennek egy elemi leképezése ( $f \in F$ ).

1. *Vektorleképezés elemi vetületei:*

1.1. Az  $f : x \mapsto y$  elemi vektorleképezés magja:

$$\text{Dom}(f) \triangleq x, \text{ ahol } x \in \mathbf{D}\langle X \rangle.$$

1.2. Az  $f : x \mapsto y$  elemi vektorleképezés képe:

$$\text{Im}(f) \triangleq y, \text{ ahol } y \in \mathbf{D}\langle Y \rangle.$$

2. *Vektorleképezés halmazvetületei I.:*

Egy vektorleképezés halmazvetületeit az alábbiakban az egyszerű leképezésnél bemutatott módon az elemi vetületek segítségével definiáljuk.

2.1. Az  $F$  vektorleképezés maghalmaza:

$$\text{Dom}(F) \triangleq \{ \text{Dom}(f) \mid f \in F \}.$$

2.2. Az  $F$  vektorleképezés képhalmaza:

$$\text{Im}(F) \triangleq \{ \text{Im}(f) \mid f \in F \}.$$

3. *Vektorleképezés halmazvetületei II.:*

Egy vektorleképezés halmazvetületeit a leképezés, mint reláció parciális halmazkeretei segítségével is értelmezhetjük az alábbi módon.

3.1. Az  $F$  vektorleképezés maghalmaza:

$$\begin{aligned} \text{Dom}(F) &\triangleq s\text{Frame}(F, X), \text{ azaz} \\ \text{Dom}(F) &= F \parallel_X. \end{aligned}$$

3.2. Az  $F$  vektorleképezés képhalmaza:

$$\begin{aligned} \text{Im}(F) &\triangleq s\text{Frame}(F, Y), \text{ azaz} \\ \text{Im}(F) &= F \parallel_Y. \end{aligned}$$

4. *Vektorleképezés vektorvetületei:*

Egy vektorleképezésnek értelmezhetjük az úgynevezett vektorvetületeit is a leképezés, mint reláció parciális vektorkeretei segítségével az alábbi módon.

4.1. Az  $F$  vektorleképezés magvektora:

$$\begin{aligned} v\text{Dom}(F) &\triangleq v\text{Frame}(F, X), \text{ azaz} \\ v\text{Dom}(F) &= F \#_X. \end{aligned}$$

4.2. Az  $F$  vektorleképezés képvektora:

$$\underline{vIm}(F) \triangleq \underline{vFrame}(F, \underline{Y}), \text{ azaz}$$

$$\underline{vIm}(F) = F \# \underline{Y}.$$

Egy vektorleképezés halmazmagjának elemeit is *magelemeknek*, képmagjának elemeit *képelemeknek* nevezzük, a *vektorleképezés argumentumainak* a magelemnek, mint vektornak az egyes komponenseit, és magát a magelemet *argumentumvektornak* fogjuk nevezni.

*Megjegyzés*

Vegyük észre, hogy egy vektorleképezés halmazvetületei vektorhalmazok, a vektorvetületei pedig halmazvektorok.

A *halmazvetületek* révén megkapjuk például egy  $n$ -változós vektorleképezés magját, illetve képét alkotó pontok összességét az attribútumhalmaz által meghatározott  $n$ -dimenziós térnek a magattribútumok, illetve képattribútumok által kijelölt résztartományában. E pontok tehát *egy-egy alakzatot alkotnak a magtartományban, illetve a képtartományban*, így egy vektorleképezés tulajdonképpen nem más, mint e két alakzat pontjai közötti kapcsolatok (élek, azaz elemi leképezések) összessége.

Mivel egy vektorleképezés *vektorvetületeinek komponenshalmazai* kifejezik a *leképezés „kiterjedését”* az attribútumhalmaz egyes attribútumainak értéktartományában, ezért egy vektorleképezés magvektorát a *leképezés értelmezési tartományának*, a képvektorát pedig *értékkészletének* is szoktuk nevezni. A gyakorlatban ezek alapján választhatjuk meg a számítógépes implementációhoz szükséges változók típusait.

**Állítás**

A vektorleképezés halmazvetületeinek az elemi leképezés és a parciális halmazkeretek alapján történő definíciója ekvivalens.

*Bizonyítás*

A bizonyítás alap gondolata az, hogy a parciális halmazkeret szerinti  $F \# \underline{X}$  halmazvetület az  $F$  leképezés, mint reláció rekordjainak az  $\underline{X}$  attribútumvektor szerinti alrekordjait tartalmazza, vagyis az  $\underline{X} \vdash \underline{X}$  alrekordokat, az  $F$  leképezés elemi leképezéseinek magjai pedig definíció-szerűen éppen ezek az alrekordok.

**Az unáris, a bináris, az  $n$ -változós és az egyrétegű vektorleképezés**

Egy  $F: \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$  vektorleképezést  *$n$ -változós*nak nevezünk, ha  $\mu(\underline{X}) = n$ , és *egyrétegűnek*, ha  $\mu(\underline{Y}) = 1$ . Az  $n$ -változós vektorleképezést *bináris*nak is nevezzük, ha  $n = 2$ , és *unáris*nak, ha  $n = 1$ , azaz egyváltozós.

Az egyváltozós (tehát unáris) vektorleképezések, azaz  $\underline{X} = \langle X \rangle$  esetén az egyszerűbb kezelés érdekében elhagyjuk a vektoros jelölést, és hasonlóan járunk el az egyrétegű vektorleképezések, azaz  $\underline{Y} = \langle Y \rangle$  esetén is.

Általában, ha az alaphalmazok kevés attribútumot tartalmaznak, akkor közvetlenül ki szoktuk írni azokat a függvényjelölésben. Például  $X = \{X, Y\}$ , és  $Y = \{Z\}$ , akkor használjuk az

$$F: X \times Y \rightarrow Z$$

alakot is.

**Állítás**

A fenti egyszerűsítő jelölések és a definíciók alapján nyilvánvaló, hogy minden egyváltozós egyrétegű  $F$  vektorleképezéshez egyértelműen hozzárendelhető egy  $F'$  egyszerű leképezés, melyre fennáll, hogy

1.  $Dom(F') = Comp(\underline{vDom}(F))$ , és
2.  $Im(F') = Comp(\underline{vIm}(F))$ .

**Reláció leképezése**

A vektorleképezéseket gyakran valamely adott relációhoz rendeljük, illetve azon keresztül vezetjük be. Legyen  $A$  egy attribútumhalmaz,  $X, Y \subseteq A$ , továbbá az  $\underline{A}$ ,  $\underline{X}$  és  $\underline{Y}$  az  $A$ ,  $X$  és  $Y$  halmazok valamely vektorai. Legyen továbbá  $R = R(\underline{A})$  egy reláció az  $\underline{A}$  fölött (azaz  $R \subseteq {}^R \mathbf{D}(\underline{A})$ ). Ekkor az  $R$  reláció leképezésének nevezzük, és  $F_R$  módon jelölünk egy

$$F_R: \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$$

vektorleképezést, ha

1.  $\mu(F_R) = \mu(R)$ ,
2. létezik az  $F_R$  és az  $R$  elemi vektorainak (rekordjainak) egy kölcsönösen egyértelmű összerendelést biztosító  $i: I_{\mu(R)} \rightarrow I_{\mu(R)}$  indextranzformációja (vagyis minden  $r_k \in R$  vektorhoz kölcsönösen egyértelműen hozzárendelhető egy  $f_{i(k)} \in F_R$  vektor),
3. minden  $r_k \in R$  rekord és  $f_{i(k)} = \langle x, y \rangle$  vektor esetén fennáll (ahol tehát  $f_{i(k)} \in F_R$ ,  $x \in \mathbf{D}(\underline{X})$  és  $y \in \mathbf{D}(\underline{Y})$ ), hogy

$$x = r_k \upharpoonright_{\underline{X}}, \text{ és } y = r_k \upharpoonright_{\underline{Y}}.$$

**Megjegyzés**

Ha egy reláció valódi multihalmaz, akkor a belőle származtatott minden leképezés, valamint azok minden halmazvetületei is nyilván mind valódi multihalmazok. A halmazvetületek azonban (a vektorhalmaz-vetítés tulajdonságai miatt – amint erre a vektorhalmazok kereteinek bevezetésénél már utaltunk) akkor is lehetnek valódi multihalmazok, ha maga a reláció, vagy akár a belőle származtatott leképezés monohalmaz.

**Példa**

Legyen  $\underline{A} = \langle A, B, C, D, E \rangle$  egy attribútumvektor, ahol  $A = \{6, 7, 8\}$ ,  $B = \{3, 4, 5\}$ ,  $C = \{0, 1, 2\}$ ,  $D = \{3, 4\}$ ,  $E = \{8, 9\}$ , és tekintsük az alábbi táblával adott  $R(\underline{A})$  relációt.

Legyen továbbá  $\underline{X}, \underline{Y} \subseteq \underline{A}$  két attribútumvektor, ahol  $\underline{X} = \langle A, B \rangle$  és  $\underline{Y} = \langle C, E \rangle$ .

$R(\underline{A})$ :	A	B	C	D	E
	6	3	1	4	9
	6	3	2	3	9
	6	4	2	4	8
	7	3	1	3	8
	7	3	1	3	8
	6	3	1	3	9
	7	3	2	4	8

Határozzuk meg az  $R(\underline{A})$  reláció

$$F_R: \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$$

leképezésének vetületeit.

Az  $F_R$  vektorleképezés halmazvetületei:

$Dom(F_R):$	A	B	$Im(F_R):$	C	E	tehát
	6	3		1	9	$Dom(F_R) = \{\langle 6, 3 \rangle, \langle 6, 3 \rangle, \langle 6, 4 \rangle, \langle 7, 3 \rangle, \langle 7, 3 \rangle, \langle 6, 3 \rangle, \langle 7, 3 \rangle\}$ , és
	6	3		2	9	
	6	4		2	8	
	7	3		1	8	$Im(F_R) = \{\langle 1, 9 \rangle, \langle 2, 9 \rangle, \langle 2, 8 \rangle, \langle 1, 8 \rangle, \langle 1, 8 \rangle, \langle 1, 9 \rangle, \langle 2, 8 \rangle\}$ .
	7	3		1	8	
	6	3		1	9	
	7	3		2	8	
	7	3		2	8	

E halmazvetületek láthatóan valódi multihalmazok, mégpedig a fenti megjegyzésben említett mindkét okból következően.

Az  $F_R$  vektorleképezés vektorvetületei:

$$\underline{vDom}(F_R) = R(\underline{A})\#_{\underline{X}} = \langle \{6, 7\}, \{3, 4\} \rangle,$$

$$\underline{vIm}(F_R) = R(\underline{A})\#_{\underline{Y}} = \langle \{1, 2\}, \{8, 9\} \rangle.$$

### Elem képhalmaza (elem vektorleképezése)

Az  $F : \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$  alakú vektorleképezésekhez bevezetjük valamely  $\underline{x} \in \mathbf{D}(\underline{X})$  magelem képhalmazának és képvektorának fogalmát.

1. Az  $\underline{x}$  magelem képhalmaza az egyszerű leképezéseknél bemutatott módon bevezetve:

Ha  $\underline{x} \in Dom(F)$ , akkor

$$F[\underline{x}] \triangleq \bigsqcup_{\substack{\underline{f} \in F, \\ Dom(\underline{f}) = \underline{x}}} \{Im(\underline{f})\}.$$

2. Az  $\underline{x}$  magelem képhalmazát bevezethetjük a leképezés, mint reláció parciális halmazkerete segítségével is:

Ha  $\underline{x} \in Dom(F)$ , akkor

$$F[\underline{x}] \triangleq sFrame(F\#_{\{\underline{x}\}}, \underline{Y}), \text{ azaz}$$

$$F[\underline{x}] = (F\#_{\{\underline{x}\}})\|_{\underline{Y}}.$$

3. Értelmezhetjük az  $\underline{x}$  magelem képvektorát is a parciális vektorkeret segítségével:

Ha  $\underline{x} \in Dom(F)$ , akkor

$$\underline{E}[\underline{x}] \triangleq \underline{vFrame}(F\#_{\{\underline{x}\}}, \underline{Y}), \text{ azaz}$$

$$\underline{E}[\underline{x}] = (F\#_{\{\underline{x}\}})\#_{\underline{Y}}.$$

Természetesen (az egyszerű leképezéshez hasonlóan) ha  $\underline{x} \notin Dom(F)$ , akkor az  $F$  vektorleképezés nincs értelmezve, és így az  $\underline{x}$  magelem  $F[\underline{x}]$  képhalmaza és  $\underline{E}[\underline{x}]$  képvektora sem.

### Példa

Határozzuk meg a fenti példa esetén az  $\underline{x} = \langle 6, 3 \rangle$  magelem képhalmazát és képvektorát.

A  $\langle 6, 3 \rangle$  rekord képhalmaza:

$$F_R[\langle 6, 3 \rangle] = (F_R\#_{\{\langle 6, 3 \rangle\}})\|_{\underline{Y}}, \text{ azaz}$$

$F_R \#_{\{\langle 6, 3 \rangle\}}:$	<table><tr><th>A</th><th>B</th><th>C</th><th>E</th></tr><tr><td>6</td><td>3</td><td>1</td><td>9</td></tr><tr><td>6</td><td>3</td><td>2</td><td>9</td></tr><tr><td>6</td><td>3</td><td>1</td><td>9</td></tr></table>	A	B	C	E	6	3	1	9	6	3	2	9	6	3	1	9	$F_R[\langle 6, 3 \rangle]:$	<table><tr><th>C</th><th>E</th></tr><tr><td>1</td><td>9</td></tr><tr><td>2</td><td>9</td></tr><tr><td>1</td><td>9</td></tr></table>	C	E	1	9	2	9	1	9
A	B	C	E																								
6	3	1	9																								
6	3	2	9																								
6	3	1	9																								
C	E																										
1	9																										
2	9																										
1	9																										

tehát

$$F_R[\langle 6, 3 \rangle] = \{\langle 1, 9 \rangle, \langle 2, 9 \rangle, \langle 1, 9 \rangle\}.$$

A  $\langle 6, 3 \rangle$  rekord képvektora:

$$E_R[\langle 6, 3 \rangle] = (F_R \#_{\{\langle 6, 3 \rangle\}}) \#_Y = \langle \{1, 2\}, \{9\} \rangle.$$

### Állítás

Az elem-leképezés fenti definíciójában szereplő vetületfogalmak kifejtésével egy magelem képhalmaza és képvektora az alábbi alakban is kifejezhető (a definícióbeli jelölésekkel):

1. Valamely  $\underline{x} \in \text{Dom}(F)$  magelem képhalmaza:

$$F[\underline{x}] = \{ \underline{y} \mid \underline{y} \in \text{Im}(F), \underline{y} = \underline{f} \# \underline{x}, \underline{f} \in F, \underline{f} \# \underline{x} = \underline{x} \}.$$

2. Valamely  $\underline{x} \in \text{Dom}(F)$  magelem képvektorának  $i$  indexű komponense:

$$E[\underline{x}](i) = \bigcup_{\underline{f} \in F} \{ \underline{f}(\underline{y}(i)) \}, \text{ ahol } i \in \text{Ind}(\underline{y}), \text{ és természetesen}$$

$$\mu(E[\underline{x}]) = \mu(\underline{y}).$$

### Állítás

Az elem-leképezés fogalmának segítségével nyilván ki lehet fejezni a vektorleképezés képhalmazát és képvektorát:

1. Az  $F$  vektorleképezés képhalmaza:

$$\text{Im}(F) = \bigcup_{\underline{x} \in \text{Dom}(F)} F[\underline{x}].$$

2. Az  $F$  vektorleképezés képvektorának  $i$  indexű komponense:

$$\underline{\text{vIm}}(F)(i) = \bigcup_{\underline{x} \in \text{Dom}(F)} \underline{F}[\underline{x}](i), \text{ ahol } i \in \text{Ind}(\underline{y}), \text{ és természetesen}$$

$$\mu(F) = \mu(\underline{y}).$$

### Állítás

Egy  $F : \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$  vektorleképezés képének vetületeit az alábbi módon is definiálhatjuk, és e definíciók ekvivalensek az eredeti definícióval:

1. Az  $F$  vektorleképezés képhalmaza:

$$\text{Im}(F) \triangleq F[\text{Dom}(F)].$$

2. Az  $F$  vektorleképezés képvektora:

$$\underline{\text{vIm}}(F) \triangleq \underline{F}[\text{Dom}(F)].$$

*Bizonyítás* (Az állítás 2. részét bizonyítjuk, az 1. rész bizonyítása ezzel lényegében azonos.)

Az elem-leképezés definíciója alapján

$$E[\text{Dom}(F)] = (F \#_{\text{Dom}(F)}) \#_Y. \quad (1)$$

Mivel definíció szerint

$$\text{Dom}(F) = F \#_X, \quad (2)$$

így a hasonlósági vetítés fogalmának bevezetésénél kimondott állítás alapján

$$F \#_{Dom(F)} = F. \quad (3)$$

Ezt behelyettesítve az (1)-be, kapjuk

$$E[Dom(F)] = F \#_Y. \quad (4)$$

A vektorkép definíciója szerint

$$\underline{vIm}(F) = \underline{vFrame}(F, \underline{Y}), \quad (5)$$

ahol a vektorkeret

$$\underline{vFrame}(F, \underline{Y}) = F \#_Y. \quad (6)$$

A (4), (5) és (6) összevetéséből már adódik az állítás 2. része. ■

## Leképezés-vetületek, mint speciális részhalmazok

A fenti jelöléseket felhasználva tetszőleges  $F : \mathbf{D}\langle \underline{X} \rangle \rightarrow \mathbf{D}\langle \underline{Y} \rangle$  vektorleképezés esetén nyilvánvalóan teljesülnek az alábbi állítások:

1.  $Dom(F) \cong F$ ,
2.  $Im(F) \cong F$ ,
3.  $Dom(F) \subseteq \mathbf{D}\langle \underline{X} \rangle$ ,
4.  $Im(F) \subseteq \mathbf{D}\langle \underline{Y} \rangle$ .
5.  $\underline{vDom}(F) \subseteq^* \underline{X}$ ,
6.  $\underline{vIm}(F) \subseteq^* \underline{Y}$ ,

## Vektorfüggvény

Egy  $F : \mathbf{D}\langle \underline{X} \rangle \rightarrow \mathbf{D}\langle \underline{Y} \rangle$  alakú vektorleképezést *vektorfüggvénynek* nevezünk, ha minden  $\underline{x} \in Dom(F)$  magelem esetén

$$\mu(Reduc(F[\underline{x}])) = 1,$$

és ekkor használjuk a szokásos

$$F(\underline{x})$$

jelölést az  $\underline{x}$  magelem *függvényképére*, ahol tehát

$$Reduc(F[\underline{x}])) = \{F(\underline{x})\}.$$

### Példa

Legyen  $\underline{A} = \langle A, B, C, D, E \rangle$  egy attribútumvektor, ahol  $A = \{6, 7, 8\}$ ,  $B = \{3, 4, 5\}$ ,  $C = \{0, 1, 2\}$ ,  $D = \{3, 4\}$ ,  $E = \{8, 9\}$ , és tekintsük az alábbi táblával adott  $R\langle \underline{A} \rangle$  relációt. Legyen továbbá  $\underline{X}, \underline{Y} \subseteq \underline{A}$  két attribútumvektor, ahol  $\underline{X} = \langle A, B \rangle$  és  $\underline{Y} = \langle C, E \rangle$ .

$R\langle \underline{A} \rangle$ :	A	B	C	D	E
	6	3	1	4	9
	6	4	2	4	8
	7	3	2	3	8
	6	3	1	3	9

### Feladat

Állapítsuk meg, hogy az  $R\langle \underline{A} \rangle$  reláció  $F_R : \mathbf{D}\langle \underline{X} \rangle \rightarrow \mathbf{D}\langle \underline{Y} \rangle$  leképezése függvény-e, és határozzuk meg a vetületeit.

Mivel az  $F_R$  leképezés az elemi leképezéseinek halmazaként

$$F_R = \{ \langle 6, 3 \rangle \mapsto \langle 1, 9 \rangle, \langle 6, 4 \rangle \mapsto \langle 2, 8 \rangle, \langle 7, 3 \rangle \mapsto \langle 2, 8 \rangle, \langle 6, 3 \rangle \mapsto \langle 1, 9 \rangle \},$$

így  $F_R$  egy függvény (mégpedig valódi multifüggvény), hiszen nincs két különböző képhalmazbeli elem, melyhez ugyanaz a magelem tartozna. Látható azonban az is, hogy az  $F_R$  függvény inverze nem függvény (tehát  $F_R$  nem bijektív), hiszen a  $\langle 6, 4 \rangle$  és a  $\langle 7, 3 \rangle$  vektorokat, mint magelemeket az  $F_R$  függvény egyaránt a  $\langle 2, 8 \rangle$  vektorba, mint képelembe képezi le. Szintén e felírásból látható, hogy az  $F_R$  függvény halmazvetületei:

$Dom(F_R)$ :	$\begin{array}{cc} A & B \\ 6 & 3 \\ 6 & 4 \\ 7 & 3 \\ 6 & 3 \end{array}$	$Im(F_R)$ :	$\begin{array}{cc} C & E \\ 1 & 9 \\ 2 & 8 \\ 2 & 8 \\ 1 & 9 \end{array}$	tehát
				$Dom(F_R) = \{ \langle 6, 3 \rangle, \langle 6, 4 \rangle, \langle 7, 3 \rangle, \langle 6, 3 \rangle \}$ , és
				$Im(F_R) = \{ \langle 1, 9 \rangle, \langle 2, 8 \rangle, \langle 2, 8 \rangle, \langle 1, 9 \rangle \}$ ,

míg vektorvetületei:

$$\underline{vDom}(F_R) = R(\underline{A}) \#_{\underline{X}} = \langle \{6, 7\}, \{3, 4\} \rangle, \text{ és}$$

$$\underline{vIm}(F_R) = R(\underline{A}) \#_{\underline{Y}} = \langle \{1, 2\}, \{8, 9\} \rangle.$$

## Hipervektor-leképezés

Legyen  $A$  egy attribútumhalmaz,  $X_1, \dots, X_n, Y \subseteq A$ , továbbá legyenek  $\underline{A}, \underline{X}_1, \dots, \underline{X}_n$  és  $\underline{Y}$  az  $A, X_1, \dots, X_n$  és  $Y$  halmazok valamely vektorai. Ekkor egy

$$F \subseteq \mathbf{D}(\underline{X}_1) \times \dots \times \mathbf{D}(\underline{X}_n) \times \mathbf{D}(\underline{Y})$$

relációt *n-változós hipervektor-leképezésnek* nevezünk, és ezt a továbbiakban

$$F : \mathbf{D}(\underline{X}_1) \times \dots \times \mathbf{D}(\underline{X}_n) \rightarrow \underline{Y}$$

módon jelöljük. Az  $X_1, \dots, X_n, Y$  halmazokat itt is a *leképezés alaphalmazainak* nevezzük.

A hiper-vektor leképezés minden elemi leképezése tehát vektorok rendezett  $n$ -esét (vagyis egy  $n$ -elemű hipervektort) képez le egyetlen vektorba.

Az

$$F : \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$$

hipervektor-leképezést *unáris*, vagy *egyváltozós hipervektor-leképezésnek*, az

$$F : \mathbf{D}(\underline{X}) \times \mathbf{D}(\underline{Y}) \rightarrow \mathbf{D}(\underline{Z})$$

alakúakat pedig *bináris*, vagy *kétváltozós hipervektor-leképezéseknek* nevezzük. Láthatóan a vektorleképezések unáris hipervektor-leképezéseknek is tekinthetők.

### A hipervektor-leképezés vetületei

Legyen  $F : \mathbf{D}(\underline{X}_1) \times \dots \times \mathbf{D}(\underline{X}_n) \rightarrow \underline{Y}$  egy hipervektor-leképezés,  $f \in F$  egy elemi leképezés, a "+" pedig a vektorláncolás műveletjele.

1. *Hipervektor-leképezés elemi vetületei:*

1.1. Az  $f : \underline{x} \mapsto \underline{y}$  elemi hipervektor-leképezés magja:

$$Dom(f) \triangleq \underline{x}, \text{ ahol } \underline{x} \in \mathbf{D}(\underline{X}_1) \times \dots \times \mathbf{D}(\underline{X}_n).$$

1.2. Az  $f : \underline{x} \mapsto \underline{y}$  elemi hipervektor-leképezés képe:

$$Im(f) \triangleq \underline{y}, \text{ ahol } \underline{y} \in \mathbf{D}(\underline{Y}).$$

2. Hipervektor-leképezés halmazvetületei I.:

Halmazvetületek definiálása elemi vetület segítségével.

2.1. Az  $F$  hipervektor-leképezés maghalmaza:

$$Dom(F) \triangleq \{ Dom(f) \mid f \in F \}.$$

2.2. Az  $F$  hipervektor-leképezés képhalmaza:

$$Im(F) \triangleq \{ Im(f) \mid f \in F \}.$$

3. Hipervektor-leképezés halmazvetületei II.:

Halmazvetületek definiálása parciális halmazkeret segítségével.

3.1. Az  $F$  hipervektor-leképezés maghalmaza:

$$Dom(F) \triangleq sFrame(F, \underline{X}_1 + \dots + \underline{X}_n), \text{ azaz}$$

$$Dom(F) = F \parallel_{(\underline{X}_1 + \dots + \underline{X}_n)}.$$

3.2. Az  $F$  hipervektor-leképezés képhalmaza:

$$Im(F) \triangleq sFrame(F, \underline{Y}), \text{ azaz}$$

$$Im(F) = F \parallel_{\underline{Y}}.$$

4. Hipervektor-leképezés vektorvetületei:

Vektorvetületek definiálása parciális vektorkeret segítségével.

4.1. Az  $F$  hipervektor-leképezés magvektora:

$$\underline{vDom}(F) \triangleq \underline{vFrame}(F, \underline{X}_1 + \dots + \underline{X}_n), \text{ azaz}$$

$$\underline{vDom}(F) = F \#_{(\underline{X}_1 + \dots + \underline{X}_n)}.$$

4.2. Az  $F$  hipervektor-leképezés képvektora:

$$\underline{vIm}(F) \triangleq \underline{vFrame}(F, \underline{Y}), \text{ azaz}$$

$$\underline{vIm}(F) = F \#_{\underline{Y}}.$$

### Elem képhalmaza (elem hipervektor-leképezése)

Az  $F : \mathbf{D}(\underline{X}_1) \times \dots \times \mathbf{D}(\underline{X}_n) \rightarrow \underline{Y}$  alakú hipervektor-leképezésekhez is bevezetjük valamely  $\underline{x} \in \mathbf{D}(\underline{X}_1) \times \dots \times \mathbf{D}(\underline{X}_n)$  magelem képhalmazának és képvektorának fogalmát.

1. Magelem képhalmazának definiálása elemi vetület segítségével.

Ha  $\underline{x} \in Dom(F)$ , akkor

$$F[\underline{x}] \triangleq \bigcup_{\substack{f \in F, \\ Dom(f) = \underline{x}}} \{Im(f)\}.$$

2. Magelem képhalmazának definiálása parciális halmazkeret segítségével.

Ha  $\underline{x} \in Dom(F)$ , akkor

$$F[\underline{x}] \triangleq sFrame(F \#_{\{\underline{x}\}}, \underline{Y}), \text{ azaz}$$

$$F[\underline{x}] = (F \#_{\{\underline{x}\}}) \parallel_{\underline{Y}}.$$

3. *Magelem képvektorának* definiálása parciális vektorkeret segítségével.

Ha  $\underline{x} \in \text{Dom}(F)$ , akkor

$$E[\underline{x}] \triangleq \mathbf{vFrame}(F \upharpoonright_{\{\underline{x}\}}, \underline{Y}), \text{ azaz}$$

$$E[\underline{x}] = (F \upharpoonright_{\{\underline{x}\}}) \upharpoonright_{\underline{Y}}.$$

Természetesen (az eddigiekhez hasonlóan) ha  $\underline{x} \notin \text{Dom}(F)$ , akkor az  $F$  vektorleképezés nincs értelmezve, és így az  $\underline{x}$  magelem  $F[\underline{x}]$  képhalmaza és  $E[\underline{x}]$  képvektora sem.

Láthatóan hipervektor-leképezés esetén a magelem képhalmazainak definiálása a vektor-leképezésnél bemutatott módon történt.

## Hipervektorfüggvény

Egy  $F : \mathbf{D}\langle \underline{X}_1 \rangle \times \dots \times \mathbf{D}\langle \underline{X}_n \rangle \rightarrow \underline{Y}$  hipervektor-leképezést *n-változós hipervektorfüggvénynek* nevezünk, ha minden  $\underline{x} \in \text{Dom}(F)$  magelem esetén

$$\mu(\text{Reduc}(F[\underline{x}])) = 1,$$

és ekkor használjuk a szokásos

$$F(\underline{x})$$

jelölést az  $\underline{x}$  magelem *függvényképére*, ahol tehát

$$\text{Reduc}(F[\underline{x}]) = \{F(\underline{x})\}.$$

### Példa

Hipervektor-leképezés például karaktersorozatok (azaz vektorok) láncolása (konkatenációja). Ez nem csak leképezés, de nyilvánvalóan függvény, sőt az alaphalmazok megfelelő megválasztása esetén – mint látni fogjuk – művelet is.

## Halmazleképezés

Legyen  $A$  egy attribútumhalmaz, és  $X_1, \dots, X_n, Y \subseteq A$ . Ekkor egy

$$F \subseteq 2^{X_1} \times \dots \times 2^{X_n} \times 2^Y$$

relációt *n-változós halmazleképezésnek* nevezünk, és azt a továbbiakban

$$F : 2^{X_1} \times \dots \times 2^{X_n} \rightarrow 2^Y$$

módon jelöljük (ahol például  $2^Y$  az  $Y$  halmaz hatványhalmaza, vagyis az  $Y$  részhalmazainak halmaza). Az  $X_1, \dots, X_n, Y$  halmazokat itt is a *leképezés alaphalmazainak* nevezzük.

### Megjegyzés

A halmazleképezés minden elemi leképezése tehát halmazok rendezett  $n$ -esét (vagyis egy  $n$ -elemű halmazvektort) képez le egyetlen halmazba.

Figyeljünk fel arra, hogy egy halmazleképezés elemi leképezéseinek bármelyik argumentuma, vagy a képe lehet üres halmaz, mivel az üres halmaz minden halmaz részhalmaza.

A későbbiekben fogunk olyan halmazleképezéseket használni (az úgynevezett függőségeket), melyekben az alaphalmazok *hiperhalmazok* lesznek. Ezek kezeléséhez bevezethetnénk a *hiperhalmaz-függvényeket*, de tárgyalásunkhoz erre nem lesz szükség.

**Példa**

Halmazleképezések például az alábbiak:

- Halmazok Descartes-szorozása, mely nem csak leképezés, de függvény is, azonban nem művelet.
- A halmazalgebra alapvető műveletei nem csupán halmazleképezések, hanem függvények, sőt (megfelelő alaphalmazok esetén) algebrai értelemben vett műveletek is.

**A halmazleképezés vetületei**

Legyen  $F : 2^{X_1} \times \dots \times 2^{X_n} \rightarrow 2^Y$  egy halmazleképezés, és  $f \in F$  egy elemi leképezés.

1. *Halmazleképezés elemi vetületei:*

1.1. Az  $f : \underline{x} \mapsto y$  elemi halmazleképezés magja:

$$Dom(f) \triangleq \underline{x}, \text{ ahol } \underline{x} \in 2^{X_1} \times \dots \times 2^{X_n}.$$

1.2. Az  $f : \underline{x} \mapsto y$  elemi halmazleképezés képe:

$$Im(f) \triangleq y, \text{ ahol } y \in 2^Y.$$

2. *Halmazleképezés halmazvetületei:*

2.1. Az  $F$  halmazleképezés maghalmaza:

$$Dom(F) \triangleq \{ Dom(f) \mid f \in F \}.$$

2.2. Az  $F$  halmazleképezés képhalmaza:

$$Im(F) \triangleq \{ Im(f) \mid f \in F \}.$$

Halmazleképezés vetületeihez a parciális keretek nyilván nem alkalmazhatóak.

**Egyszerű halmazleképezés**

Egy  $F$  halmazleképezés egyszerű, ha minden  $f \in F$  esetén

$$\mu(Im(f)) = 1.$$

**Elem képhalmaza (elem halmazleképezése)**

Az  $F : 2^{X_1} \times \dots \times 2^{X_n} \rightarrow 2^Y$  halmazleképezés valamely  $\underline{x} \in 2^{X_1} \times \dots \times 2^{X_n}$  magelemének képhalmaza  $\underline{x} \in Dom(F)$  esetén

$$F[\underline{x}] \triangleq \bigcup_{\substack{f \in F, \\ Dom(f) = \underline{x}}} \{ Im(f) \}.$$

Természetesen (az eddigiekhez hasonlóan) ha  $\underline{x} \notin Dom(F)$ , akkor az  $F$  halmazleképezés nincs értelmezve, és így az  $\underline{x}$  magelem  $F[\underline{x}]$  képhalmaza sem.

**Halmazfüggvény**

Egy  $F : 2^{X_1} \times \dots \times 2^{X_n} \rightarrow 2^Y$  halmazleképezést *n-változós halmazfüggvénynek* nevezünk, ha minden  $\underline{x} \in Dom(F)$  magelem esetén

$$\mu(Reduc(F[\underline{x}])) = 1,$$

és ekkor használjuk a szokásos

$$F(\underline{x})$$

jelölést az  $\underline{x}$  magelem *függvényképére*, ahol tehát

$$\text{Reduc}(F[\underline{x}]) = \{F(\underline{x})\}.$$

### Példa

Tekintsük a halmazok egyesítését. Legyen  $X = \{a, b\}$ ,  $Y = \{1, 2\}$  és  $Z = \{a, b, 1, 2\}$  végül legyen az  $F: 2^X \times 2^Y \rightarrow 2^Z$  halmazleképezés, mint elemi leképezéseinek halmaza  $F = \{\underline{f}_i \mid i \in I_{16}\}$ , ahol  $\underline{f}_1: \langle \emptyset, \emptyset \rangle \mapsto \emptyset$ , ...,  $\underline{f}_k: \langle \{a\}, \{2\} \rangle \mapsto \{a, 2\}$ , ...,  $\underline{f}_m: \langle \{a, b\}, \{1\} \rangle \mapsto \{a, b, 1\}$ , ...,  $\underline{f}_{16}: \langle \{a, b\}, \{1, 2\} \rangle \mapsto \{a, b, 1, 2\}$ .

Az  $F$  leképezés maghalmaza  $\text{Dom}(F) = \{\langle \emptyset, \emptyset \rangle, \dots, \langle \{a\}, \{2\} \rangle, \dots, \langle \{a, b\}, \{1\} \rangle, \dots, \langle \{a, b\}, \{1, 2\} \rangle\}$ , és képhalmaza  $\text{Im}(F) = \{\emptyset, \dots, \{a, 2\}, \dots, \{a, b, 1\}, \dots, \{a, b, 1, 2\}\}$ . Végül például az  $F$  leképezés  $\langle \{b\}, \emptyset \rangle$  magelemének képhalmaza  $F[\langle \{b\}, \emptyset \rangle] = \{\{b\}\}$ . Láthatóan minden magelemhez egyelemű képhalmaz tartozik, vagyis ez a halmazleképezés egy függvény.

## Vektorhalmaz-leképezés

Legyen  $A$  egy attribútumhalmaz,  $X_1, \dots, X_n, Y \subseteq A$ , továbbá  $\underline{A}, \underline{X}_1, \dots, \underline{X}_n$  és  $\underline{Y}$  az  $A, X_1, \dots, X_n$  és  $Y$  halmazok valamely vektorai. Ekkor az

$$F \subseteq 2^{\mathbf{D}(\underline{X}_1)} \times \dots \times 2^{\mathbf{D}(\underline{X}_n)} \times 2^{\mathbf{D}(\underline{Y})}$$

relációt *n-változós vektorhalmaz-leképezésnek* nevezzük, melyet a továbbiakban

$$F: 2^{\mathbf{D}(\underline{X}_1)} \times \dots \times 2^{\mathbf{D}(\underline{X}_n)} \rightarrow 2^{\mathbf{D}(\underline{Y})}$$

módon jelölünk. Az  $X_1, \dots, X_n, Y$  halmazokat itt is a *leképezés alaphalmazainak* nevezzük.

A vektorhalmaz-leképezés minden elemi leképezése tehát vektorhalmazok rendezett  $n$ -esét (vagyis egy  $n$ -elemű halmazvektort) képez le egyetlen vektorhalmazba.

A vektorhalmaz-leképezés segítségével valósíthatók meg a relációk leképezései, majd a későbbiekben a relációkra vonatkozó transzformációk és műveletek, vagyis a legtöbbet éppen ezzel a típusú leképezéssel fogunk foglalkozni.

### A vektorhalmaz-leképezés vetületei

Legyen  $F: 2^{\mathbf{D}(\underline{X}_1)} \times \dots \times 2^{\mathbf{D}(\underline{X}_n)} \rightarrow 2^{\mathbf{D}(\underline{Y})}$  egy vektorhalmaz-leképezés, és  $\underline{f} \in F$ .

1. *Vektorhalmaz-leképezés elemi vetületei:*

1.1. Az  $\underline{f}: \underline{x} \mapsto y$  elemi vektorhalmaz-leképezés *magja*:

$$\text{Dom}(\underline{f}) \triangleq \underline{x}, \text{ ahol } \underline{x} \in 2^{\mathbf{D}(\underline{X}_1)} \times \dots \times 2^{\mathbf{D}(\underline{X}_n)}.$$

1.2. Az  $\underline{f}: \underline{x} \mapsto y$  elemi vektorhalmaz-leképezés *képe*:

$$\text{Im}(\underline{f}) \triangleq y, \text{ ahol } y \in 2^{\mathbf{D}(\underline{Y})}.$$

2. *Vektorhalmaz-leképezés halmazvetületei:*

2.1. Az  $F$  vektorhalmaz-leképezés *maghalmaza*:

$$Dom(F) \triangleq \{ Dom(f) \mid f \in F \}.$$

2.2. Az  $F$  vektorhalmaz-leképezés képhalmaza:

$$Im(F) \triangleq \{ Im(f) \mid f \in F \}.$$

A parciális keretek nyilván a vektorhalmaz-leképezés vetületeihez sem alkalmazhatóak.

### Elem képhalmaza (elem vektorhalmaz-leképezése)

Az  $F : 2^{D(X_1)} \times \dots \times 2^{D(X_n)} \rightarrow 2^{D(Y)}$  vektorhalmaz-leképezés valamely  $\underline{x} \in 2^{D(X_1)} \times \dots \times 2^{D(X_n)}$  *magelemének képhalmaza*  $\underline{x} \in Dom(F)$  esetén ezúttal is

$$F[\underline{x}] \triangleq \bigcup_{\substack{f \in F, \\ Dom(f) = \underline{x}}} \{ Im(f) \}.$$

Természetesen (mint eddig is) ha  $\underline{x} \notin Dom(F)$ , akkor az  $F$  halmazleképezés nincs értelmezve, és így az  $\underline{x}$  magelem  $F[\underline{x}]$  képhalmaza sem.

### Vektorhalmazfüggvény

Egy  $F : 2^{D(X_1)} \times \dots \times 2^{D(X_n)} \rightarrow 2^{D(Y)}$  vektorhalmaz-leképezést  $n$ -változós vektorhalmazfüggvénynek nevezünk, ha minden  $\underline{x} \in Dom(F)$  magelem esetén

$$\mu(Reduc(F[\underline{x}])) = 1,$$

és ekkor használjuk a szokásos

$$F(\underline{x})$$

jelölést az  $\underline{x}$  magelem *függvényképére*, ahol tehát

$$Reduc(F[\underline{x}]) = \{ F(\underline{x}) \}.$$

### Példa

Vektorhalmaz-leképezés például egy relációnak az alábbiakban bemutatott (a relációalgebrai műveleteknél tárgyalandó) dekompozíciója (felbontása), mely persze nem függvény.

$R(\underline{A})$ :	$U$	$P$	$R$	$Az R(\underline{A})$ dekompozíciója	$R(\underline{A}) \parallel_{\langle U, P \rangle}$ :	$U$	$P$	$R(\underline{A}) \parallel_{\langle U, R \rangle}$ :	$U$	$R$
	3	1	5	$az \langle U, P \rangle$ és $az \langle U, R \rangle$		3	1		3	5
	3	3	6	vektorok szerint		3	3		3	6
	4	2	5	$\Rightarrow$		4	2		4	5
	3	3	6			3	3		3	6

# Műveletek és struktúrák

## Művelet

A gyakorlatban igen sűrűn használunk különböző matematikai műveleteket, mégis ritkán tudatosul az, hogy milyen tevékenységet is jelent egy művelet elvégzése valamely halmazon, illetve, hogy melyek azok az objektumok, amelyeken valamely adott művelet egyáltalán elvégezhető. (Például számokról szólva általában a természetes számokra gondolunk, pedig az aritmetikai műveleteink többsége azokon nem is értelmezhető, vagy például a hétköznapi gondolkodásban nemigen tartjuk számon, hogy a szorzás műveletét másképpen végezzük egész számok, mint törtszámok körében.)

Vizsgálataink során unáris és bináris (egy és két operandussal rendelkező, azaz egy- és kétváltozós) műveletekkel fogunk foglalkozni. Unáris műveletre tekintünk példaként az előjelváltást ( $-x$ ) és az invertálást ( $1/x$ ), bináris műveletre pedig a kivonást és az osztást.

Figyeljünk fel arra, hogy nyilván nem tudunk előjelet váltani a természetes (nemnegatív egész) számok halmazán (a nulla kivételével), és a kivonás sem végezhető el itt korlátlanul. Hasonlóképpen nem tudjuk értelmezni az invertálást az egész számok halmazán, gondjaink vannak az osztással is, és persze egyáltalán nem tudjuk értelmezni ezeket a műveleteket például az ábécé betűire. Tekintsük ezekkel szemben például a láncolás (összefűzés, konkatenáció) műveletét, melyet viszont csak karakterek és karaktersorozatok esetén értelmezünk. További példaként említhetjük a hatványozást, mely a természetes számokon korlátlanul elvégezhető, míg az egész számokon csak akkor, ha a hatványkitevő pozitív szám (hiszen az egész számok negatív kitevőjű hatványa már 1-nél kisebb eredményt ad).

A fentiek alapján foglalkozunk össze „igényeinket” a műveletekkel szemben.

1. *A művelet legyen értelmezve az adott halmazon.*
  - 1.1. Ez egyrészt azt jelenti, hogy az adott halmaz egyetlen eleme se legyen kizárva a művelet operandusai közül. (Például a valós számok halmazából a nullát el kell hagyni, hogy az osztás művelete lehessen ennek a halmaznak.)
  - 1.2. Másrészt azt is jelenti, hogy ha a művelet több operandussal rendelkezik, akkor ezek az operandusok ugyanabból a halmazból származzanak, azaz minden operandusnak ugyanaz legyen az értékkészlete.
2. *A művelet legyen zárt az adott halmazon.* Eszerint a művelet eredménye tetszőleges operandus(ok) esetén legyen eleme az adott halmaznak. (Ezt az unáris műveletek közül sem az „előjelváltás a természetes számokon”, sem az „invertálás az egész számokon”, vagy a „gyökvonás a racionális számokon”, a bináris műveletek közül sem a „kivonás a természetes számokon”, sem az „osztás az egész számokon” nem teljesíti.)

## Néhány megjegyzés műveletek definíciójáról

Amint már említettük, a művelet fogalmát a függvények fogalmából fogjuk származtatni. Ennek során azonban célszerű lesz néhány egyszerűsítő feltevést alkalmazni a gyakorlati igényekkel összhangban. Tekintsük példaképpen az alábbiakban bemutatásra kerülő unáris egyszerű művelet definícióját.

Legyen  $\underline{A} = \langle X, Y \rangle$  egy attribútumvektor. Egy ezen értelmezett  $F : X \rightarrow Y$  egyszerű függvény egy *unáris egyszerű művelet* a  $Dom(F)$  halmazon, ha  $Im(F) \subseteq Dom(F)$ . Mivel a gyakorlatban alkalmazott műveletek esetén nem csupán  $Im(F) = Dom(F)$  teljesül, hanem még  $Dom(F) = Y = X$  is, ezért célszerű a definíciót ennek figyelembevételével egyszerűsíteni, vagyis a művelet-definíciókra alábbi ajánlások tehetők:

1. legyen  $Im(F) = X$ , vagyis használjuk az  $F : X \rightarrow X$  függvény alakot, továbbá
2. alkalmazzuk a  $Dom(F) = X$  feltételt (tehát a műveletet értelmezzük a teljes  $X$  halmazon).

Többváltozós műveletek esetén célszerű feltételezni, hogy a művelet operandusai (változói) ugyanabból a halmazból származnak. Például egy egyszerű  $n$ -változós műveletet az általános tárgyalásban az  $\underline{A}$  attribútumvektoron értelmezett  $F : \mathbf{D}(\underline{X}) \rightarrow Y$   $n$ -változós függvényből származtatnánk, ahol természetesen  $X \subseteq A$ ,  $Y \in A$ ,  $\underline{X} = X \upharpoonright_{\underline{A}}$ , és  $\mu(X) = n$ , továbbá  $\mu(A)$  és  $n$  egymástól független értékek. Miután a gyakorlatban a műveleteknek az egyes attribútumokra vetített értelmezési tartománya úgyszólván azonos, ezért célszerű a

$$\text{minden } i, j \in I_n \text{ esetén legyen } Comp(Dom(F)|_{\{X_i\}}) = Comp(Dom(F)|_{\{X_j\}})$$

feltételt helyettesíteni a

$$\text{minden } i, j \in I_n \text{ esetén legyen } X_i = X_j$$

feltétellel, és így a  $\mathbf{D}(\underline{X})$  helyett célszerűbb az  $X^n$  hatványt használni a függvénymegadásban, ahol minden  $i \in I_n$  esetén  $X_i = X$ . Ennek értelmében

3. az  $n$ -változós műveletet az  $F : X^n \rightarrow X$   $n$ -változós függvényből származtatjuk, ahol tehát már figyelembe vettük az 1. pontban mondottakat.

A továbbiakban a művelet-definíciókat már a fenti három pontnak megfelelően fogjuk megfogalmazni. Megjegyezzük, hogy a továbbiakban kizárólag egy- és kétváltozós (azaz unáris és bináris) műveletekkel fogunk foglalkozni.

## Egyszerű művelet (unáris és bináris műveletek), infix jelölés

Az egyszerű műveleteket az egyszerű függvényekből származtatjuk.

1. Valamely  $X$  halmazon értelmezett

$$F : X \rightarrow X$$

függvény egy *unáris művelet* az  $X$  halmazon, ha

- 1.1. értelmezve van a teljes  $X$  halmazon, azaz

$$Dom(F) = X, \text{ és}$$

- 1.2. zárt az  $X$  halmazon, azaz

$$Im(F) \subseteq X,$$

ahol ez utóbbi feltétel következik az  $F$  függvény  $X \rightarrow X$  alakjából.

2. Az  $n$ -változós egyszerű függvény segítségével általánosított műveletet  *$n$ -áris műveletnek* nevezzük, ám mivel a gyakorlatban az  $n > 2$  eset nem fordul elő (nagyon ritkán használunk *ternális*, *kvadrális*, stb. műveleteket), ezért az alábbi definícióban csak az  $n = 2$  esettel, az úgynevezett *bináris művelettel* foglalkozunk. Ennek során az unáris művelet gyakorlati definíciójából fogunk kiindulni.

Valamely  $X$  halmazon értelmezett

$$F : X \times X \rightarrow X$$

kétváltozós függvény egy *bináris művelet az  $X$  halmazon*, ha

2.1. értelmezve van a teljes  $X$  halmazon, azaz

$$\text{Dom}(F) = X \times X, \text{ és}$$

2.2. zárt az  $X$  halmazon, azaz

$$\text{Im}(F) \subseteq X,$$

ahol ez utóbbi feltétel következik az  $F$  függvény  $X \times X \rightarrow X$  alakjából.

A bináris műveleteket a szokásos, úgynevezett *infix* alakban fogjuk használni. Tehát valamely  $a, b, c \in X$  esetén az eddigi  $c = F(a, b)$  alak helyett a  $c = a F b$  alakot fogunk írni.

## Összetett műveletek

Az összetett műveleteket az összetett függvényekből származtatjuk az egyszerű műveletnél bemutatott módon. Nyilván ezeknél is alapkövetelmény, hogy a függvények alaphalmazai azonosak legyenek.

## Struktúra

Az előzőekben a műveletet olyan speciális függvényként értelmeztük, melyben a megkötés a művelet argumentumait adó alaphalmazokra vonatkozott. Művelet és alaphalmaz, halmaz és a rajta értelmezett műveletek, ezt a kettősséget fejezi ki a struktúra, mint összetett fogalom, melyet egy halmaz alkot a rajta értelmezett bizonyos tulajdonságú műveletekkel együtt.

A struktúrát azért igen fontos fogalom, mivel a világ dolgai számunkra mindig valamilyen tevékenységen keresztül jelennek meg, és – más oldalról nézve – maguk a tevékenységek sem értelmezhetők azon konkrét dolgok nélkül, melyekre hatnak. A csupán algebrai tulajdonságai által definiált absztrakt struktúrák – más néven algebrák – tanulmányozása azért hasznos, mert áttekinthető formában ismerhetünk meg olyan összefüggéseket, melyek eredeti „természetes” közegükben csak nagyon nehezen, vagy sohasem lettek volna felismerhetők. Valamely természeti jelenségnek – mint például egy adatbázisnak (!) – az algebrai eszközök nélkül való tanulmányozása hasonló ahhoz, mint amikor egy idegen városban akarunk eligazodni térkép nélkül, vagy amikor az időben akarunk tájékozódni dátumok nélkül.

## Egyszerű struktúra

Legyen  $H$  egy halmaz,  $M$  pedig műveletek halmaza. Ekkor egy

$$S = \langle H, M \rangle$$

párost *egyszerű algebrai struktúrának*, vagy csak *egyszerű struktúrának* nevezünk, ha minden  $m \in M$  művelet egyszerű művelet, értelmezve van, továbbá zárt a  $H$  halmazon, azaz

1.  $\text{Dom}(m) \subseteq H$ , és
2.  $\text{Im}(m) \subseteq \text{Dom}(m)$ .

## Struktúrák tulajdonságai

Mint láttuk, a struktúra tulajdonképpen egy művelettel/műveletekkel ellátott halmaz. Ha az alaphalmaz egyszerű (azaz csak egyszerű elemet tartalmaz), akkor *egyszerű struktúráról* beszélünk, egyébként *összetett struktúráról*.

Mivel a struktúrák tulajdonságait alapvetően a rajtuk értelmezett műveletek határozzák meg, ezért az alábbiakban áttekintjük a műveletek fő típusait. E tulajdonságokat a gyakorlati igényeknek és a szokásoknak megfelelően bináris műveletekre fogjuk megfogalmazni az alábbiakban.

Legyen  $S = \langle H, \circ \rangle$  egy egyszerű struktúra, ahol  $H$  egy halmaz, " $\circ$ " pedig egy rajta értelmezett bináris egyszerű művelet.

### ASSZOCIATIVITÁS

A " $\circ$ " művelet *asszociatív* a  $H$  halmazon, ha minden  $a, b, c \in H$  esetén

$$(a \circ b) \circ c = a \circ (b \circ c).$$

Az asszociatív műveleteknél a zárójelezés el is hagyható és egyszerűen az  $a \circ b \circ c$  alakot használjuk, a műveletsorozat végrehajtását pedig balról-jobbra értelmezzük.

Az asszociativitás jelentőségét az adja, hogy az ilyen típusú műveletek esetén  $a \circ b \circ c \dots$  műveletsorozat láncszerűen hajtható végre, a benne szereplő műveleteket balról-jobbra irányban egymásután elvégezve.

Az olyan struktúrát, melynek van asszociatív művelete *félcsoportnak* nevezzük. Félcsoportot alkot például a természetes számok halmaza az összeadásra, a szorzásra nézve, de például a kivonásra, vagy a hatványozásra már nem. Megemlítjük, hogy például a mátrixok azonban még a szorzásra vonatkozóan sem alkotnak félcsoportot (ha amúgy a szorzás értelmezhető is – például azonos rendű, valós számokat tartalmazó négyzetes mátrixok esetén). Fontos félcsoportot alkotnak egy nyelv szavai a láncolás (összefűzés, konkatenáció) műveletére nézve.

### KOMMUTATIVITÁS

A " $\circ$ " művelet *kommutatív* a  $H$  halmazon, ha minden  $a, b \in H$  esetén

$$a \circ b = b \circ a.$$

A kommutativitás különösen az aritmetikai műveleteknél fontos (sok algoritmusban, többek között az egyenletmegoldásoknál kihasználjuk), de például mátrixok szorzására a kommutativitás sem teljesül.

### INVERTÁLHATÓSÁG

A " $\circ$ " művelet *balról invertálható* a  $H$  halmazon, ha minden  $a, b \in H$  esetén egyetlen olyan  $x \in H$  elem létezik, melyre

$$a \circ x = b,$$

és *jobbról invertálható*, ha minden  $a, b \in H$  esetén egyetlen olyan  $y \in H$  elem létezik, melyre

$$y \circ a = b.$$

Végül a „ $\circ$ ” művelet *invertálható* a  $H$  halmazon, ha jobbról és balról egyaránt invertálható ugyanitt.

Az invertálható művelettel rendelkező struktúrákat *csoportoknak* nevezzük. Az invertálhatóság igen egyszerűvé teszi egy struktúra egyenleteinek megoldását, ugyanis ennek révén van lehetőség egy változó explicit kifejezésére.

Persze nagyon sok, számunkra fontos struktúra műveletei nem invertálhatóak. Invertálható például az összeadás az egész számokon (inverz művelete a kivonás), a szorzás a racionális számokon (inverz művelete az osztás), sőt még a négyzetes mátrixok szorzása is invertálható. Nem invertálható azonban sem az összeadás, sem a szorzás a természetes számok halmazán.

### EGYSÉGELEM

A „ $\circ$ ” műveletnek a  $H$  halmazon valamely  $e \in H$  a *jobboldali egységelem*e, ha tetszőleges  $a \in H$  esetén

$$a \circ e = a,$$

a *baloldali egységelem*e, ha bármely  $a \in H$  esetén

$$e \circ a = a.$$

Végül természetesen az  $e \in H$  a „ $\circ$ ” művelet *egységelem*e a  $H$  halmazon, ha jobboldali és baloldali egységelem is egyidejűleg.

Egységelem a nulla a természetes, vagy akár a valós számokon értelmezett összeadásnak (vagy az 1 ugyanezen halmazokon értelmezett szorzásnak), de már a háromdimenziós térben a vektorösszeadásnak nem ez az egységelem, hanem a  $\langle 0, 0, 0 \rangle$  vektor. Az aritmetikai kivonásban pedig csupán a jobboldali egységelem a nulla.

Az algebraiban sok műveletnek egységelem az üres halmaz. A halmazegyesítésnek egységelem, a halmazkivonásnak pedig jobboldali egységelem. Különlegességgént megemlítjük, hogy valamely halmaznak az üres halmazzal való Descartes-szorzata önmagát adja (ha nem is a „műveletvégzés” szabályaiból adódóan, hanem csupán definíció szerinti kiterjesztésként), így az üres halmaz a Descartes-szorzás egységelemének tekinthető. (Egyébként a Descartes-szorzás algebrai értelemben – mint láttuk – nem művelet.)

Az egységelem tulajdonságait mutatják be az alábbi állítások.

### Állítás

1. Ha egy struktúrának létezik valamely műveletre vonatkozóan jobboldali és baloldali egységelem is, akkor a kettő megegyezik.
2. Egy műveletnek legfeljebb egy egységelem lehet egy struktúrában.

### Bizonyítás

Érdekességgéppen a 2. állítást bebizonyítjuk. A bizonyítás indirekt, tehát abból indul ki, hogy létezik két egységelem, majd bebizonyítjuk, hogy a kettő azonos.

Tekintsünk egy  $\langle A, \circ \rangle$  struktúrát, és legyen ennek  $e_1$  és  $e_2$  egységelem (nyilván  $e_1, e_2 \in A$ ). Az egységelem definíciója szerint ekkor minden  $a \in A$  esetén fennállnak az alábbi egyenlőségek:

$$a \circ e_1 = a, \tag{1}$$

$$e_1 \circ a = a, \tag{2}$$

$$a \circ e_2 = a, \quad (3)$$

$$e_2 \circ a = a. \quad (4)$$

Ha a fenti egyenlőségek tetszőleges  $a \in A$  esetén fennállnak, akkor nyilván teljesülnek  $a = e_1$  és  $a = e_2$  esetén is. Végezzük el az  $a = e_1$  behelyettesítést a (3) egyenlőségben, és az  $a = e_2$  behelyettesítést a (2) egyenlőségben. Ekkor az alábbi egyenlőségeket kapjuk:

$$(3) \Rightarrow e_1 \circ e_2 = e_1, \quad (5)$$

$$(2) \Rightarrow e_1 \circ e_2 = e_2. \quad (6)$$

A műveletek általános szabályai szerint (minthogy a művelet fogalmát a függvény fogalmából származtattuk) a művelet eredménye egyértelmű, tehát (5)-ből és (6)-ból következik

$$e_1 = e_2,$$

amit bizonyítani akartunk. ■

## Összetett struktúrák

Az összetett struktúrák legalább egy összetett műveletet tartalmaznak. Mivel a következő alfejezetek (és fejezetek) összetett műveletekkel foglalkoznak, itt csak annyit jegyünk meg, hogy a gyakorlatban sokszor műveletnek neveznek olyan leképezést, amelyik esetleg még csak nem is függvény (lásd például majd a relációműveleteknél a dekompozíciót). Ilyen esetekben a művelet fogalmát nem algebrailag értelmezzük, hanem, mint valami elvégezhető számítássorozatot, algoritmust.

## Osztályozás

Az osztályozás (algebrában elterjedt nevén particionálás) művelete során egy halmazt elemeinek valamilyen tulajdonsága alapján diszjunkt (tehát közös elemet nem tartalmazó) részhalmazaira bontunk.

Például már egy kisgyermek által megoldható az a feladat, hogy a Lego építőelemeit színeik szerint osztályozza. Ekkor egy-egy osztályba kerülnek a piros, a kék, a sárga, stb. színű elemek függetlenül attól, hogy milyen formájúak. Már ennek az egyszerű példának a kapcsán is bemutatathatóak az osztályozás legfontosabb tulajdonságai:

1. minden osztály a teljes elemhalmaz egy részhalmaza,
2. az egyes osztályokban lévő elemek összességében kiadják az eredeti elemhalmazt (tehát minden elem valamelyik osztályban van),
3. bármely két osztály diszjunkt,
4. minden osztályban teljesül a felosztás alapját képező kritérium (esetünkben az azonos szín),
5. bármely osztálybeli bármelyik elemmel kibővítünk egy másik osztályt, a kibővített osztályra már nem teljesül az osztályozási kritérium.

Ahhoz, hogy ez a jól áttekinthető struktúra létrejöjjön két dologra volt szükségünk. Egyrészt elemek egy megadott halmazára, továbbá az elemek egy olyan tulajdonságára, mely szerint egyezőségük, vagy különbözőségük eldönthető, hiszen ennek alapján kerülnek azonos, vagy különböző osztályba.

Az osztályozás igen fontos művelet lesz számunkra. Segítségével egyrészt a gyakorlatban gyakran előforduló osztályozási feladatokat tudunk algebrailag megoldani, másrészt tématerületünk legfontosabb fogalmait hídként fogja összekötni.

## Alapfogalmak

### Ekvivalencia-reláció

Legyen  $A$  egy halmaz, és  $\varepsilon_A \subseteq A \times A$  egy reláció, ahol  $\times$  a halmazok közötti Descartes-szorzás jele. Ekkor az  $A$ -beli párosokból álló  $\varepsilon_A$  relációt *az  $A$  halmaz fölötti ekvivalencia-relációnak* nevezzük, ha

*reflexív*, vagyis az  $A$  halmaz minden eleme önmagával „relációban áll”, azaz

minden  $a_k \in A$  esetén  $\langle a_k, a_k \rangle \in \varepsilon_A$ ,

*szimmetrikus*, vagyis a „relációban állás” független a sorrendtől, azaz

minden  $a_k, a_l \in A$  esetén, ha  $\langle a_k, a_l \rangle \in \varepsilon_A$ , akkor  $\langle a_l, a_k \rangle \in \varepsilon_A$ , és

*transzitiv*, vagyis a „relációban állás” „öröklődik”, azaz

minden  $a_k, a_l, a_m \in A$  esetén, ha  $\langle a_k, a_l \rangle \in \varepsilon_A$  és  $\langle a_l, a_m \rangle \in \varepsilon_A$ , akkor  $\langle a_k, a_m \rangle \in \varepsilon_A$ .

#### Megjegyzés

Az  $\varepsilon_A$  ekvivalencia-reláció „öröklí” az  $A$  halmaz „mono-”, illetve „multi-” jellegét, azaz csak akkor lesz  $\varepsilon_A$  monohalmaz, ha az  $A$  halmaz is az.

**Példa**

Ekvivalencia-reláció például az „azonos színű” kapcsolat a bevezetőben említett Lego elemek halmazán.

Könnyen belátható, hogy ez valóban *reflexív* (hiszen minden elem azonos színű önmagával), *szimmetrikus* (hiszen ha egy  $a$  elem azonos színű egy  $b$  elemmel, akkor a  $b$  elem is azonos színű az  $a$  elemmel), és *transzitiv* (hiszen ha egy  $a$  elem azonos színű egy  $b$  elemmel, a  $b$  azonos színű a  $c$ -vel, akkor az  $a$  elem azonos színű a  $c$ -vel is).

Általában bármilyen egyenlőség ekvivalencia-reláció. Nem teljesül viszont a szimmetria a „ $\leq$ ” összehasonlításra, a „ $<$ ” összehasonlításra pedig még a reflexivitás sem.

**Halmazfelosztás, particionálás**

Egy  $A$  halmaz *halmazfelosztása*, vagy egyszerűen csak *felosztása* egy olyan

$$\Phi[A]$$

módon jelölt halmaz, melynek

1. elemei az  $A$  nemüres részhalmazai, azaz

$$\text{minden } A_i \in \Phi[A] \text{ esetén } A_i \subseteq A, \text{ és } A_i \neq \emptyset,$$

2. elemi halmazait egyesítve az  $A$  halmazt kapjuk, azaz

$$\bigsqcup_{A_i \in \Phi[A]} A_i = A,$$

3. elemi halmazai diszjunktak, azaz

$$\text{minden } A_i, A_j \in \Phi[A] \text{ és } A_i \neq A_j \text{ esetén } A_i \cap A_j = \emptyset.$$

Ekkor a  $\Phi[A]$  halmazfelosztás elemi halmazait *osztályoknak* nevezzük. A halmazfelosztást a szakirodalomban még *particionálásnak* is nevezik.

**Megjegyzés**

Elemi algebrai eszközökkel bizonyítható (ám ezt most nem tesszük meg) a következő igen fontos állítás: *Minden ekvivalencia-relációhoz kölcsönösen egyértelmű módon hozzárendelhető egy halmazfelosztás.* (Gondoljunk az előző példára.)

A fenti definíció akkor is helyes, ha  $A$  multihalmaz. A multiunióknak itt tehát nem az a „szerepe”, hogy a művelet eredményeként az azonos elemek a különböző osztályokból többszöröződhessenek (hiszen az azonos elemek nyilván ugyanabban az osztályban vannak), hanem egyszerűen az, hogy a multihalmazok között az egyesítés oly módon legyen elvégezhető, hogy az elemtöbbszöröződések megjelenjenek az eredményhalmazban is. Erre pedig csak a multiunió alkalmas, ugyanis a monounió nem alkalmazható multihalmazokra, a redukáló egyesítés pedig eltüntetné a többszörös elemelőfordulásokat.

**Osztálykritérium**

A gyakorlatban nem előre döntjük el, hogy egy felosztási feladatban mely elemek kerüljenek egy osztályba, hanem megfogalmazunk egy összehasonlító feltételt (ez az osztálykritérium), amelyet ha teljesít két vizsgált elem, akkor egy osztályba kerülnek, egyébként nem.

1. *Halmazfelosztás osztálykritériuma*

Legyen egy  $A$  halmaz valamely felosztása  $\Phi[A]$ . Ekkor egy  $T_A : A \times A \rightarrow \{\text{HAMIS}, \text{IGAZ}\}$  logikai függvényt a  $\Phi[A]$  *halmazfelosztás osztálykritériumának* nevezzük, ha

a) az egy osztályban levő elempárosokra IGAZ logikai értéket ad, azaz

$$\forall a_k, a_l \in A : (\exists A_i \in \Phi[A] : a_k, a_l \in A_i) \models T_A(a_k, a_l) = \text{IGAZ}, \text{ és}$$

b) a különböző osztályban levő elempárosokra HAMIS logikai értéket ad, azaz

$$\forall a_k, a_l \in A : (\nexists A_i \in \Phi[A] : a_k, a_l \in A_i) \models T_A(a_k, a_l) = \text{HAMIS}.$$

2. *Halmazfelosztás kiterjesztett osztálykritériuma*

Az osztálykritériumot, mint logikai függvényt kiterjesztjük az  $A$  halmaz részhalmazaira is a következőképpen: a  $T_A$  logikai függvény az  $A$  halmaz valamely  $B$  részhalmazára

c) IGAZ értéket ad, ha a  $B$  minden elempárjára teljesül a  $T_A$  logikai függvény, azaz

$$\forall B \subseteq A : T_A(B) \triangleq \text{IGAZ}, \text{ ha } \forall a_k, a_l \in B : T_A(a_k, a_l) = \text{IGAZ}, \text{ és}$$

d) HAMIS értéket ad, ha van olyan elempárja a  $B$ -nek, melyre nem teljesül a  $T_A$  logikai függvény, azaz

$$\forall B \subseteq A : T_A(B) \triangleq \text{HAMIS}, \text{ ha } \exists a_k, a_l \in B : T_A(a_k, a_l) = \text{HAMIS}.$$

*Megjegyzés*

A gyakorlatban az ekvivalencia-relációkat az osztálykritériumokon keresztül definiáljuk. Azt ugyanis, hogy az ekvivalencia-reláció a Descartes-szorzatbeli elempárosok mely részhalmaza, mindig valamilyen tulajdonsággal adjuk meg. Ezt nevezzük osztálykritériumnak.

Az osztálykritériumot közvetlenül is megfogalmazhatjuk ekvivalencia-relációra: Legyen  $\varepsilon$  egy, az  $A$  halmaz fölötti ekvivalencia-reláció. Ekkor a  $T_\varepsilon : A \times A \rightarrow \{\text{HAMIS}, \text{IGAZ}\}$  logikai függvényt az  $\varepsilon$  *ekvivalencia-reláció osztálykritériumának* nevezzük, ha

- minden  $a_k, a_l \in A$  esetén, ha  $\langle a_k, a_l \rangle \in \varepsilon$ , akkor  $T_\varepsilon(a_k, a_l) = \text{IGAZ}$ , és
- minden  $a_k, a_l \in A$  esetén, ha  $\langle a_k, a_l \rangle \notin \varepsilon$ , akkor  $T_\varepsilon(a_k, a_l) = \text{HAMIS}$ .

Végül figyeljünk fel arra, hogy az osztálykritériumnak az  $A$  halmaz egy  $B$  részhalmazára vonatkozó  $T_A(B)$  kiterjesztése algebrailag nem származtatható az osztálykritérium-függvény definíciójából. Az mindössze egy „gyorsírási” jelölés, melynek segítségével a későbbiekben majd tömörebben tudjuk megfogalmazni a gyakorlati osztálykritériumot.

Megjegyezzük még, hogy az osztálykritériumot a gyakorlatban mindig „pozitív módon”, az „egy osztályban való tartózkodásra” fogalmazzuk meg. Nem mondjuk ki, hanem természetesen tekintjük, hogy különböző osztályok elemei között nem teljesül.

## Néhány megjegyzés a halmazfelosztás és az ekvivalencia-reláció kapcsolatáról

Gondoljuk el az alábbi egyszerű kísérletet. Tekintsük ismét a Lego elemeknek a színeik alapján való osztályozását, de most a halmazfelosztás algebrai definíciója szerint. Ekkor tehát minden osztály teljesíti az „azonos színű elemek” követelményt. Vegyünk ki ezek után egy elemet valamelyik osztályból és tegyük át egy másikba. A kibővített osztály vajon teljesíti-e az „azonos színű elemek” követelményt? Lehet, hogy igen! Hogy miért? Vizsgáljuk meg a halmazfelosztás definícióját, és azt fogjuk találni, hogy az nem zárja ki azt, hogy legyen például két „piros elemek” osztály!

Egész más a helyzet, ha ekvivalencia-relációval, vagy (ami ezzel lényegében egyenértékű) valamilyen osztálykritériummal végezzük az osztálybasorolást. Az ekvivalencia-reláció esetén a tranzitivitás garantálja azt, hogy például minden piros színű elem egyetlen osztályba kerüljön, és az osztálykritériumot is úgy definiáltuk, hogy különböző osztálybeli elemekre az ne teljesüljön.

A fentiekből az következne, hogy a halmazfelosztást követő megjegyzés nem igaz? Nem volna tehát igaz az, hogy minden halmazfelosztáshoz egyértelműen hozzárendelhető egy ekvivalencia-reláció? De igen, csak nem biztos, hogy az éppen a vizsgált ekvivalencia-reláció!

Számozzuk meg például a Lego elemeket, és mondjuk azt, hogy egy osztályt alkot az "1" jelű, egy másik osztályt a "3", az "5" és a "11" jelű, egy harmadik osztályt pedig a "2" és a "10" jelű. Ezek után legyen az ekvivalencia-reláció a következő:  $\varepsilon = \{ \langle 1, 1 \rangle, \langle 3, 3 \rangle, \langle 5, 5 \rangle, \langle 11, 11 \rangle, \langle 3, 5 \rangle, \langle 5, 3 \rangle, \langle 3, 11 \rangle, \langle 11, 3 \rangle, \langle 5, 11 \rangle, \langle 11, 5 \rangle, \langle 2, 2 \rangle, \langle 10, 10 \rangle, \langle 2, 10 \rangle, \langle 10, 2 \rangle \}$ . Ekvivalencia-reláció-e az  $\varepsilon$  vektorhalmaz? Nyilván az. Az  $\{1, 2, 3, 5, 10, 11\}$  halmaz elemeivel jelölt Lego elemek  $\{\{1\}, \{3, 5, 11\}, \{2, 10\}\}$  osztályozása egy halmazfelosztás? Láthatóan igen! E kettő egyértelműen meghatározza egymást? Nyilvánvalóan! Lehet-e két csupa piros elemet tartalmazó osztály? Akár mind a három!

A halmazfelosztás és az ekvivalencia-reláció tehát ekvivalens fogalmak. A továbbiakban mi a halmazfelosztásokkal fogunk foglalkozni, mivel ez a tevékenység kötődik azokhoz a táblákhoz, amelyekkel a korábbi tárgyalás során már találkoztunk, és amelyekből a fizikai adatbázisok felépülnek. E halmazfelosztás (osztályozás) végrehajtása szempontjából láthatóan igen fontos, hogy mi az a tulajdonság, amelynek révén az osztálybasorolást elvégezzük. A továbbiakban ezért olyan osztályozásokat fogunk bemutatni, melyeknél e tulajdonságot (az osztálykritériumot) a definíció részének tekintjük. Ezáltal a halmazfelosztási tulajdonságok a definícióbeli háromról kibővülnek azzal a kettővel, melyeket már a bevezető példában (két utolsóként) bemutatunk.

## Gyakorlati osztályozások

### Halmazfelosztás osztálykritérium szerint

Legyen  $A$  egy halmaz, és  $T = T_A$  egy, az  $A$  halmazon értelmezett osztálykritérium. Ekkor az  $A$  halmazfelosztása, vagy egyszerűen csak felosztása a  $T$  osztálykritérium szerint egy olyan

$$\Phi[A|T]$$

módon jelölt halmaz,

1. melynek elemei az  $A$  részhalmazai, azaz

$$\text{minden } A_i \in \Phi[A|T] \text{ esetén } A_i \subseteq A,$$

2. melynek elemi halmazait egyesítve az  $A$  halmazt kapjuk, azaz

$$\bigsqcup_{A_i \in \Phi[A|T]} A_i = A,$$

3. melynek elemi halmazai diszjunktak, azaz

$$\text{minden } A_i, A_j \in \Phi[A|T] \text{ és } A_i \neq A_j \text{ esetén } A_i \cap A_j = \emptyset,$$

4. melynek elemi halmazán teljesül a  $T$  osztálykritérium, azaz minden  $A_i \in \Phi[A|T]$  esetén  $T(A_i) = \text{IGAZ}$ , és
5. melynek bármely elemi halmazát kibővítve egy másik elemi halmazból származó elemmel, az így kapott halmazra már nem teljesül a  $T$  osztálykritérium, azaz minden  $A_i, A_j \in \Phi[A|T]$ ,  $A_i \neq A_j$  és  $a \in A_i$  esetén  $T(A_j \cup \{a\}) = \text{HAMIS}$ .
- Ekkor a  $\Phi[A|T]$  halmazfelosztás elemi halmazait *osztályoknak* nevezzük.

#### Megjegyzés

Láthatóan a halmazfelosztásnál adott 1-3. követelmények itt kiegészültek (a bevezető példából már jól ismert) 4-5. követelményekkel, ezáltal biztosítva, hogy minden, az osztálykritériumnak megfelelő tulajdonság-előfordulás csak egy osztályban teljesüljön (például ne legyen két „piros” osztály).

#### Példa

Tekintsük az  $I_8 = \{1, 2, 3, 4, 5, 6, 7, 8\}$  halmazt, és ennek  $\Phi[I_8|T]$  felosztását a

$$T = ( \forall b \in \Phi[I_8|T], \forall a, b \in B : ( T(a, b) = \text{IGAZ} \Leftrightarrow a \bmod 2 = b \bmod 2 ) )$$

osztálykritérium szerint, vagyis egy osztályba kerüljenek az  $I_8$  halmaz azonos paritású elemei. Határozzuk meg a  $\Phi[I_8|T]$  felosztást. A megoldás nyilván:

$$\Phi[I_8|T] = \{ \{1, 3, 5, 7\}, \{2, 4, 6, 8\} \}.$$

### Sémafelosztás és reláció-felosztás osztálykritérium szerint

Legyen  $A$  egy attribútumhalmaz,  $\mathbf{R} = \mathbf{R}(\underline{A})$  egy relációséma, és  $T = T_{\mathbf{R}}$  egy, az  $\mathbf{R}$  relációsémán, mint a rekordok halmazán értelmezett osztálykritérium. Ekkor az  $\mathbf{R}$  relációséma  $T$  osztálykritérium szerinti sémafelosztása nem más, mint az  $\mathbf{R}$  relációsémának, mint halmaznak a  $T$  osztálykritérium szerinti

$$\Phi[\mathbf{R}|T]$$

halmazfelosztása.

A fentiekkel azonos módon értelmezhetjük az  $R = R(\underline{A})$  relációnak a  $\Phi[R|T]$  reláció felosztását is, csak akkor a  $T = T_R$  osztálykritérium az  $R$  reláción van értelmezve.

#### Megjegyzés

Mivel a sémafelosztás egy relációsémán, mint rekordjainak halmazán elvégzett halmazfelosztás, így a halmazfelosztás tulajdonságainál elmondottak értelmében a sémafelosztás „örökli” a relációséma monohalmaz-jellegét.

A reláció-felosztást tehát multihalmaz-felosztásként értelmezzük (maga a reláció-felosztás már lehet multihalmaz is, azaz egy osztályban több azonos rekord is lehet).

A sémafelosztások és reláció-felosztások nyilván ugyanazokkal az algebrai tulajdonságokkal rendelkeznek, mint amelyeket a halmazfelosztásnál már bemutattunk.

#### Példa

Tekintsük az  $\underline{A} = \langle X, Y, Z \rangle$  attribútumvektoron értelmezett, alábbi táblájával adott  $R = R(\underline{A})$  relációt, és az ezen értelmezett

$$T_R = ( \forall \underline{r}, \underline{s} \in R(\underline{A}) : ( \underline{r}(X) = \underline{s}(X) \models \underline{r}(Y) = \underline{s}(Y) ) )$$

osztálykritériumot. Határozzuk meg a  $\Phi[R|T_R]$  reláció felosztást.

$R: \begin{array}{c} XYZ \\ 2\ 3\ 5 \\ 1\ 4\ 7 \\ 2\ 3\ 6 \\ 3\ 4\ 5 \\ 1\ 4\ 7 \end{array}$	$\Phi[R T_R]$	$R_1: \begin{array}{c} XYZ \\ 2\ 3\ 5 \\ 2\ 3\ 6 \end{array}$	$R_2: \begin{array}{c} XYZ \\ 1\ 4\ 7 \\ 1\ 4\ 7 \end{array}$	$R_3: \begin{array}{c} XYZ \\ 3\ 4\ 5 \end{array}$
	$\Rightarrow$			

Tehát

$$\Phi[R|T_R] = \{R_1\langle \underline{A} \rangle, R_2\langle \underline{A} \rangle, R_3\langle \underline{A} \rangle\}.$$

Figyeljünk fel arra, hogy a fenti logikai feltétel tulajdonképpen egy  $X \rightarrow Y$  függvénykapcsolatot határoz meg. Az ilyen jellegű függvényeket fogjuk a következő fejezet központi fogalmaként függőségeknek nevezni.

### Sémafelosztás és reláció-felosztás attribútumhalmaz szerint

Egy  $A$  attribútumhalmazon értelmezett  $\mathbf{R} = \mathbf{R}\langle \underline{A} \rangle$  relációsémának valamely  $B \subseteq A$  attribútumhalmaz szerinti sémafelosztása nem más, mint az  $\mathbf{R}$  relációsémának egy olyan sémafelosztása, ahol egy osztályba azok a rekordok kerülnek, melyeknek a  $B$ -beli attribútumokon vett értékei megegyeznek, azaz

$$\Phi[\mathbf{R}|B] \triangleq \Phi[\mathbf{R}|T_{\mathbf{R}}^B],$$

melyben az osztálykritérium tehát:

$$T_{\mathbf{R}}^B = (\text{minden } \mathbf{R}_i \in \Phi[\mathbf{R}|B], \text{ } \underline{r}, \underline{t} \in \mathbf{R}_i, \text{ és } A_k \in B \text{ esetén } \underline{r}(A_k) = \underline{t}(A_k)), \quad (*)$$

illetve tömörebben leírva (felhasználva a hasonló rekordok 1. definícióját)

$$T_{\mathbf{R}}^B = (\forall \mathbf{R}_i \in \Phi[\mathbf{R}|B], \text{ } \underline{r}, \underline{t} \in \mathbf{R}_i: \underline{r} \stackrel{B}{\sim} \underline{t}).$$

A  $B$  attribútumhalmaz elemeit e sémafelosztás *osztályképző-attribútumainak* nevezzük.

A  $\Phi[\mathbf{R}|B]$  sémafelosztás *osztálytag-attribútum halmazának* nevezzük a legbővebb olyan  $B^* \subseteq A$  attribútumhalmazt, melyre e felosztás minden osztályában még teljesül az összes rekord hasonlósága, azaz

$$\forall \mathbf{R}_i \in \Phi[\mathbf{R}|B], \text{ } \underline{r}, \underline{t} \in \mathbf{R}_i: \underline{r} \stackrel{B^*}{\sim} \underline{t}.$$

Nyilván az osztálytag-attribútumoknak részalmazát alkotják az osztályképző-attribútumok, azaz  $B \subseteq B^*$ .

A fentiekkel azonos módon itt is értelmezhetjük az  $R = R\langle \underline{A} \rangle$  relációnak a  $B$  attribútumhalmaz szerinti  $\Phi[R|B]$  reláció felosztását.

#### Megjegyzés

Könnyen belátható, hogy a  $T_{\mathbf{R}}^B$  logikai függvényt megadó (\*) kifejezés kielégíti az osztálykritérium definíciójában szereplő követelményeket.

Az attribútumhalmaz szerinti sémafelosztást (illetve reláció-felosztást) tehát a sémafelosztás (illetve reláció-felosztás) általános, osztálykritérium szerinti felosztásából származtatuk, így az ott tett megjegyzések nyilván itt is fennállnak.

**Példa**

Tekintsük az előző példában szereplő  $\underline{A} = \langle X, Y, Z \rangle$  attribútumvektort, az abban táblázatával adott  $R = R(\underline{A})$  relációt, továbbá a  $B = \{X, Y\}$  attribútumhalmazt, és határozzuk meg a  $\Phi[R|B]$  reláció felosztást.

Jól láthatóan ezúttal is az előző felosztást kapjuk, tehát

$$\Phi[R|B] = \{R_1(\underline{A}), R_2(\underline{A}), R_3(\underline{A})\}.$$

Megjegyezzük, hogy természetesen ez az egyezés most az előző példában szereplő osztálykritérium speciális megadásából következik. A fordított irányú kapcsolat azonban nyilván fennáll, azaz minden attribútumhalmazhoz megadható olyan osztálykritérium, mely vele azonos felosztást eredményez.

**Elemi-sémafelosztás, egységfelosztás**

Végül bevezetünk két speciális sémafelosztást, melyek különösen a következő fejezetben fognak jelentős szerepet kapni.

Legyen  $\mathbf{R} = \mathbf{R}(\underline{A})$  egy relációséma. Ennek egy sémafelosztását *elemi-sémafelosztásnak* nevezzük, ha minden sémaosztálya egyelemű, és minden sémaosztály az  $\mathbf{R}$  egy-egy rekordját tartalmazza, azaz

$$\Phi^e[\mathbf{R}] \triangleq \{ \{r\} \mid r \in \mathbf{R} \}, \text{ és}$$

*egységfelosztásnak* nevezzük, ha annak csak egyetlen sémaosztálya van, és az maga a relációséma, tehát

$$\Phi^E[\mathbf{R}] \triangleq \mathbf{R}.$$

## Relációalgebrai műveletek

Az alábbiakban – a multiműveletek és redukáló műveletek alkalmazásaként, esetenként kiterjesztéseként – bemutatjuk a relációk gyakorlatban megvalósított műveleteit. E műveletek segítségével egy, vagy két relációból egy újat tudunk létrehozni. Ezek tehát reláció-transzformációk, melyek az úgynevezett *paraméter-relációkat* leképezik (általában) egy úgynevezett *eredmény-relációba*. Azokat a műveleteket, melyek egy paraméter-relációt tartalmaznak *unárisnak*, amelyek kettőt, azokat *binárisnak* nevezzük.

Ezek egyik típusa (redukálás, metszetképzés, különbségképzés, egyesítés, Descartes-szorítás, osztás) a relációkon, mint halmazokon végez alapvetően halmazalgebrai műveleteket, egy másik típusba tartozók (projekció, dekompozíció, szelekció) a reláción, mint táblán hajtanak végre valamilyen (sor, vagy oszlop) kiválasztási transzformációt. Harmadik típust (természetes és általános összekapcsolás) azok a relációs műveletek alkotják, amelyek olyan, összetett relációs műveleteket végeznek, melyek az előző két típus tulajdonságát ötvözik, azaz két relációból úgy hoznak létre egy harmadikat, hogy a kiinduló tábláknak csak bizonyos feltételeket kielégítő sorait veszik bele a műveletbe. Ez utóbbi műveletek már rendkívül összetett kiválasztási és lekérdezési lehetőségeket biztosítanak, a gyakorlati adatbázis-kezelő (például az SQL nyelvet megvalósító) szoftverek alapvető eszközkészletét alkotják. Végül a negyedik típust az úgynevezett osztályozó műveletek alkotják, melyek a paraméter-reláció bizonyos tulajdonságú rekordjait helyettesítik egyetlen reprezentáns rekorddal, miközben bizonyos, jellemzően aritmetikai függvényeket határoznak meg.

E műveletek közül érdemes külön kiemelni a multiunió a projekció és a dekompozíció műveletét, mint *multihalmazképző műveletet*. Ez alatt azt értjük, hogy e műveletek eredménye (ellentétben a metszettel, a különbség képzéssel, vagy például a szelekcióval) akkor is lehet multihalmaz, ha a művelet paraméterei monohalmazok.

A gyakorlatban azért fontos számon tartanunk egy műveletről, hogy multihalmazképző művelet-e, mert az adatbázisok konzisztenciájának (ellentmondás-mentességének) fenntartása érdekében a lehető legalacsonyabb szinten szeretjük tartani az adatredundanciát (adatismétlődést), márpedig ez egyértelműen növeli azt. A teljes irredundanciát természetesen egyéb szempontok miatt általában nem tudjuk biztosítani (sőt egyes esetekben – például normalizálások esetén – bizonyos típusú redundanciát mi magunk növelünk), mégis legalább azt elvárnánk, hogy egy adattáblában ugyanaz a rekord (sor) ne ismétlődjön. Adatfelvitel esetén (mint már említettük) ezt a fejlesztők által írt programok megakadályozzák, azonban bizonyos adatlekérdezési tevékenységek – mégpedig éppen a multihalmazképző művelet – során a keletkezett táblák között lehetnek multihalmazok. Mit tegyünk ezekkel?

Az egyik lehetőség, hogy megakadályozzuk az ilyen táblák létrejöttét azáltal, hogy a táblák keletkezésekor megtiltjuk a sorok többszöröződését. Erre az SQL-ben például a SELECT parancs DISTINCT paramétere ad lehetőséget. Ennek használata azonban rendkívül lelassítja a műveletvégzést, mivel minden egyes új sor generálásakor össze kell azt hasonlítani az összes már előállított sorral. Ám, ha benne hagyjuk a sorismétlődéseket, akkor ez hibát is okozhat, vagy megtévesztheti a felhasználót még akkor is, ha egy rendezés révén az azonos sorok egymás alá kerülnek (mivel esetleg nem veszi észre)!

A másik lehetőség, hogy meghagyjuk a feldolgozás során a sorismétlődéseket, és csak akkor töröljük, amikor az ismétlődés már feldolgozási hibát okozna (például leszámlálás esetén), vagy amikor a végső felhasználói listát állítjuk elő. Igaz a többlet sorok némileg

megnövelik a műveletvégzési időt, de közel sem annyira, mintha minden elemi művelet után törölnénk őket. Ráadásul a különböző műveletek eredménytáblái általában kisebbek az operandusok tábláinál, és így a többszörös elemek kiszűrése már eleve gyorsabb, mint azokban. (Persze van kivétel, például a táblák egyesítése, vagy szorzata, ahol az eredménytábla a nagyobb, ám ezeket a műveleteket elég ritkán használjuk.)

A fentiek tanulsága tehát az, hogy ne féljünk a sor-többszörözéstől, ám tartsuk számon, hogy mikor léphet fel: multihalmazképző műveletek esetén. Ezek közül a legfontosabb a projekció, mivel ezt szinte minden lekérdezésben használjuk. E műveletnél esetenként nem is az lesz a legnagyobb gondunk, hogy azonos sorok keletkeznek, hanem az, hogy nem tudjuk azonosítani a keletkezett sorokat az eredeti tábla soraival. Ennek érdekében olyan sorazonosító adatra kell majd hivatkoznunk, melyet sem az algebrai leírás, sem az SQL nem tartalmaz, azonban a gyakorlatban minden adatbázis-kezelő szoftver használ. Az Oracle-ben ezt a sorazonosítót ROWID-nek nevezik.

Az alábbi tárgyalás során a relációalgebrai műveleteket aszerint fogjuk csoportosítani, hogy a paraméter-relációk adattábláin milyen jellegű változtatásokat hajtanak végre. Eszerint beszélünk szerkezetartó, szerkezetmódosító és osztályozó műveletekről.

Végül még két megjegyzés. 1.) A „relációalgebrai műveletek” nem mindegyike művelet algebrai értelemben, ezeket elvégezhető számítássorozatként, algoritmusként értelmezzük. 2.) Az egyes műveletek definiálásakor nem foglalkozunk az úgynevezett NULL-értékekkel, ezek a III.rész gyakorlati tárgyalásában fognak majd szerepelni.

## Szerkezetartó műveletek

Az ebbe a csoportba tartozó relációs műveletek a paraméter-relációk szerkezetét nem változtatják meg, a bináris műveletek esetén azonban úgynevezett *kompatibilitási követelményeket* támasztanak azokkal szemben.

Ez a programozási gyakorlatban egyrészt azt jelenti, hogy a paraméter-relációkban meg kell egyezni az attribútumok számának, valamint a paraméter-relációk attribútumvektoraiban páronként az egyes attribútumok neveinek és adattípusainak is (mivel ez az attribútumvektor „öröklődik” át az eredményrelációba). Az esetleges név-eltérést az átnevező függvénnyel, a típuseltérést pedig az adatkonvertáló függvény alkalmazásával lehet kiküszöbölni. Az alábbiakban feltételezzük, hogy ezek a kompatibilitási követelmények teljesülnek.

### Reláció átnevezése, alias név, másodlagos attribútumnév

A relációkat esetenként úgynevezett *alias-névvel* látjuk el. Ez nem jelenti azt, hogy a relációt reprezentáló tábla „fizikailag tárolt” nevét megváltoztatjuk. Egy alias-név érvényességi köre általában nem is terjed túl egy reláció-műveleten, és a legfontosabb funkciója az egy műveletben szereplő paraméter-relációk érthetőbb megnevezése, vagy jobb megkülönböztetése. Hasonló okokból látjuk el egy reláció egyes attribútumneveit úgynevezett másodlagos attribútumnévvel. Ebben az esetben is a reláció „fizikai” attribútumneve nem változik meg, csak éppen az adott műveleteken belül lehetőség van e másodlagos néven történő hivatkozásra. Az alábbiakban ezeket az átnevezéseket formálisan is definiáljuk.

Legyen  $R(\underline{A})$  egy reláció az  $A$  attribútumhalmaz felett. Ennek  $S(\underline{B})$  alakra való átnevezését a  $Ren_{S(\underline{B})}(R(\underline{A}))$  *átnevezőfüggvény* végzi el. Ha csak a reláció nevét akarjuk megváltoz-

tatni, akkor a  $Ren_S(R)$ , ha csak az attribútumokét, akkor a  $Ren_{R(\underline{B})}(R(\underline{A}))$  alakot használjuk. A  $Ren$  függvény tehát mindössze egy átjelölést végez el.

Ha egy  $R\langle A_1, \dots, A_n \rangle$  relációban valamely  $A_i \in I_n$  attribútumot akarunk átnevezni  $B_i$  alakra, akkor használhatjuk az „AS” átnevező-operátort is  $A_i \text{ AS } B_i$  módon. Ha csak az attribútumokat, vagy azok közül is csak néhányat akarunk átnevezni, akkor az átnevező-operátor használata egyszerűbb.

## Reláció redukálása

Legyen  $R\langle \underline{A} \rangle$  egy reláció az  $A$  attribútumhalmaz felett. Ha biztosítani akarjuk, hogy  $R\langle \underline{A} \rangle$  ne tartalmazzon rekordismétlést, akkor alkalmazzuk rá a multihalmazoknál bevezetett redukáló függvényt. A  $Reduc(R\langle \underline{A} \rangle)$  már nyilván monoreláció.

## Relációk metszete, különbsége és egyesítése

Legyen  $R\langle \underline{A} \rangle$  és  $S\langle \underline{A} \rangle$  két reláció ugyanazon  $A$  attribútumhalmaz felett. (Ha  $S = S\langle \underline{B} \rangle$  és  $B \neq A$ , de  $\mu(B) = \mu(A)$ , akkor műveletvégzés előtt  $S = Ren_{S\langle \underline{A} \rangle}(S\langle \underline{B} \rangle)$  módon át kell nevezni az attribútumokat. A gyakorlatban természetesen oda kell figyelni arra is, hogy az összetartozó  $A_i$  és  $B_i$  attribútumok azonos típusúak legyenek (lásd kompatibilitási követelmények).

Ekkor értelmezhetjük az  $R\langle \underline{A} \rangle \cap S\langle \underline{A} \rangle$  relációmetszetet, mint multihalmazok redukált metszetét, az  $R\langle \underline{A} \rangle \setminus S\langle \underline{A} \rangle$  relációkülönbséget, mint multihalmazok redukált különbségét, valamint az  $R\langle \underline{A} \rangle \cup S\langle \underline{A} \rangle$ , illetve az  $R\langle \underline{A} \rangle \bowtie S\langle \underline{A} \rangle$  relációegyesítéseket, mint multihalmazok redukált egyesítését, illetve multiunióját.

### Példa

Metszet	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \\ 1 & 2 \end{array}$	$\cap$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 7 & 8 \\ 1 & 2 \end{array}$	$=$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \end{array}$	
Különbség	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \\ 1 & 2 \end{array}$	$\setminus$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 7 & 8 \end{array}$	$=$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \end{array}$	
Redukált egyesítés	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \end{array}$	$\cup$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 5 & 6 \end{array}$	$\cup$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{array}$	
Multiunió	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \end{array}$	$\bowtie$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 5 & 6 \end{array}$	$\bowtie$	$\begin{array}{c c} X & Y \\ \hline 1 & 2 \\ 3 & 4 \\ 1 & 2 \\ 5 & 6 \\ 1 & 2 \end{array}$	Láthatóan a multiunió a monorelációkból ezúttal valódi multirelációt állított elő.

A multiunió kivételével a többi halmazművelet láthatóan kiszűrte a rekordismétlődéseket.

## Szelekció (kiválasztás, szűrés)

Egy  $A$  attribútumhalmaz felett értelmezett  $R(\underline{A})$  relációnak ugyanezen  $A$  halmazon értelmezett  $T_A$  logikai függvényre vonatkozó *szelekciója* (más néven *kiválasztása*, vagy *szűrése*) az alábbi reláció:

$$\sigma_{T_A}(R(\underline{A})) \triangleq \{ \underline{r} \mid \underline{r} \in R(\underline{A}), T_A \}.$$

*Megjegyzés*

A  $\sigma_{T_A}(R(\underline{A}))$  szelekció tehát a  $T_A$  logikai feltételnek megfelelő sorokat választja ki az  $R_A$  táblából (illetőleg a nem megfelelőket kiszűri).

**Példa**

$R\langle X, Y \rangle$ :	<table><tr><th><math>X</math></th><th><math>Y</math></th></tr><tr><td>1</td><td>2</td></tr><tr><td>8</td><td>7</td></tr><tr><td>1</td><td>2</td></tr></table>	$X$	$Y$	1	2	8	7	1	2	$\sigma_{(Y>X)}(R\langle X, Y \rangle)$ :	<table><tr><th><math>X</math></th><th><math>Y</math></th></tr><tr><td>1</td><td>2</td></tr><tr><td>1</td><td>2</td></tr></table>	$X$	$Y$	1	2	1	2	A szelekció csak akkor eredményezhet valódi multirelációt, ha az operandusa is az.
$X$	$Y$																	
1	2																	
8	7																	
1	2																	
$X$	$Y$																	
1	2																	
1	2																	

## Szerkezetmódosító műveletek

A szerkezetmódosító műveletek esetén az eredmény-reláció felépítése eltér a paraméter-reláció(k)étól, ilyen módon *algebrai értelemben nem is tekinthetők műveletnek*. Ezek közül a legegyszerűbb a projekció, mely voltaképpen a paraméter-reláció kijelölt attribútumait (a tábla kijelölt oszlopait) tartja meg, esetleg rendezi át. Jelentőségét az adja, hogy az egyik legtipikusabb kiválasztási feladat, a listakészítés ezen alapszik. Mivel multihalmazt hoz létre, így alkalmazásakor fokozott jelentősége van a redukálási lehetőségnek.

### Projekció (vetítés)

Valamely  $R(\underline{A})$  reláció egy  $B \subseteq A$  attribútumhalmazra vonatkozó *projekciója* (más néven *vetítése*) az  $R(\underline{A})$  reláció  $\underline{B}$ -alrelációja, azaz

$$\pi_{\underline{B}}(R(\underline{A})) \triangleq R(\underline{A})|_{\underline{B}}.$$

*Megjegyzés*

A projekció tehát az  $R_A$  táblából kiválasztja a  $B$ -beli oszlopokat, mégpedig a  $\underline{B}$ -beli sorrendnek megfelelően.

**Példa**

$R\langle X, Y, Z\rangle:$	<table> <tr><th><math>X</math></th><th><math>Y</math></th><th><math>Z</math></th></tr> <tr><td>1</td><td>2</td><td>5</td></tr> <tr><td>3</td><td>4</td><td>6</td></tr> <tr><td>1</td><td>2</td><td>7</td></tr> <tr><td>1</td><td>2</td><td>8</td></tr> </table>	$X$	$Y$	$Z$	1	2	5	3	4	6	1	2	7	1	2	8	$\pi_{(Y, X)}(R\langle X, Y, Z\rangle):$	<table> <tr><th><math>Y</math></th><th><math>X</math></th></tr> <tr><td>2</td><td>1</td></tr> <tr><td>4</td><td>3</td></tr> <tr><td>2</td><td>1</td></tr> <tr><td>2</td><td>1</td></tr> </table>	$Y$	$X$	2	1	4	3	2	1	2	1	$Reduc(\pi_{(Y, X)}(R\langle X, Y, Z\rangle)):$	<table> <tr><th><math>Y</math></th><th><math>X</math></th></tr> <tr><td>2</td><td>1</td></tr> <tr><td>4</td><td>3</td></tr> </table>	$Y$	$X$	2	1	4	3
$X$	$Y$	$Z$																																		
1	2	5																																		
3	4	6																																		
1	2	7																																		
1	2	8																																		
$Y$	$X$																																			
2	1																																			
4	3																																			
2	1																																			
2	1																																			
$Y$	$X$																																			
2	1																																			
4	3																																			

E példában a projekció is valódi multirelációt állított elő a monorelációból. Ezt azonban szükség esetén redukálhatjuk (például listakészítéskor) a *Reduc* függvény segítségével.

## Dekompozíció

Legyen  $A$  egy attribútumhalmaz, továbbá  $B = \{\underline{B}_1, \underline{B}_2, \dots, \underline{B}_n\}$  egy attribútumvektor-halmaz, ahol minden  $\underline{B}_i \in B$  esetén  $B_i \subseteq A$  (tehát bármely  $\underline{B}_i, \underline{B}_j \in B$  esetén  $B_i \cap B_j \neq \emptyset$  megengedett), és

$$\bigcup_{\underline{B}_i \in B} B_i = A.$$

Ekkor valamely  $R(\underline{A})$  relációnak a  $B$  attribútumvektor-halmazra vonatkozó *dekompozíciója*

$$R(\underline{A})/B \triangleq \{ R_i \mid R_i = R(\underline{A})|_{\underline{B}_i}, \underline{B}_i \in B \},$$

ahol természetesen minden  $\underline{B}_i \in B$  esetén  $B_i$  a  $\underline{B}_i$  vektor alaphalmaza. (A dekompozíció tehát projekciók halmaza.)

### Példa

$$R(\underline{A}):$$

$P$	$R$	$S$	$T$	$U$
1	5	9	4	3
3	6	8	4	3
2	5	7	5	4
3	6	8	4	3
1	5	9	5	3

Legyen  $\underline{A} = \langle P, R, S, T, U \rangle$ ,  $B = \{\underline{B}_1, \underline{B}_2, \underline{B}_3\}$ , ahol  $\underline{B}_1 = \langle U, P, R \rangle$ ,  $\underline{B}_2 = \langle P, R, S \rangle$ ,  $\underline{B}_3 = \langle S, T, U \rangle$ , továbbá  $R(\underline{A})$  a baloldali táblázattal adott reláció.

Határozzuk meg az  $R(\underline{A})/B$  dekompozíciót.

Ekkor  $R(\underline{A})/B = \{R(\underline{A})|_{\underline{B}_1}, R(\underline{A})|_{\underline{B}_2}, R(\underline{A})|_{\underline{B}_3}\}$ , ahol

$R(\underline{A}) _{\underline{B}_1}: U \ P \ R$	$R(\underline{A}) _{\underline{B}_2}: P \ R \ S$	$R(\underline{A}) _{\underline{B}_3}: S \ T \ U$
3 1 5	1 5 9	9 4 3
3 3 6	3 6 8	8 4 3
4 2 5	2 5 7	7 5 4
3 3 6	3 6 8	8 4 3
3 1 5	1 5 9	9 5 3

### Megjegyzés

A fenti példához hasonlóan a gyakorlatban a dekompozíciót „átfedéssel” valósítjuk meg, tehát a származtatott relációknak vannak közös attribútumaik. E közös attribútumok használatának célja általában az, hogy az egyes rekordok azonosíthatósága fennmaradjon.

Figyeljünk fel arra, hogy a származtatott relációk rekordjainak száma azonos, és megegyezik az eredeti reláció rekordjainak számával. (Ha a relációkat hipervektorokként definiáltuk volna, akkor ennél erősebb állítás is kimondható lenne.) Látható továbbá, hogy a dekompozíció szintén képes valódi multirelációt előállítani monorelációból.

## Descartes-szorzás (szorzás)

Valamely  $R(\underline{A})$  és  $S(\underline{B})$  relációk *Descartes-szorzata* (vagy egyszerűen csak *szorzata*):

$$R(\underline{A}) \times S(\underline{B}) \triangleq \{ r \mid r \in \mathbf{D}(\underline{C}), \underline{C} = \underline{A} \dot{+} \underline{B} \},$$

ahol  $\dot{+}$  a pontozott láncolás műveletjele.

### Megjegyzés

Nyilvánvalóan a relációk Descartes-szorzatának táblája annyi sorból fog állni, mint a paraméter-relációk tábláiban lévő sorok szorzata, azaz

$$\mu(R(\underline{A}) \times S(\underline{B})) = \mu(R(\underline{A})) * \mu(S(\underline{B})).$$

A pontozott láncolásnál bevezetett minősített névhasználatból a relációalgebrai gyakorlatban annyiban szoktunk eltérni, hogy minősített elemneveket csak a közös elemek esetén használunk.

Egy reláció önmagával való szorzásakor – szintén az attribútumok megkülönböztethetősége érdekében – az egyik operandust egy alias-névvel helyettesítjük.

**Példa**

$$R\langle X, Y \rangle: \begin{array}{cc} X & Y \\ \hline 1 & 2 \\ 3 & 4 \end{array} \times S\langle Y, U, V \rangle: \begin{array}{ccc} Y & U & V \\ \hline 5 & 6 & 7 \\ 8 & 9 & 10 \\ 11 & 12 & 13 \end{array} = \begin{array}{ccccc} X & R.Y & S.Y & U & V \\ \hline 1 & 2 & 5 & 6 & 7 \\ 1 & 2 & 8 & 9 & 10 \\ 1 & 2 & 11 & 12 & 13 \\ 3 & 4 & 5 & 6 & 7 \\ 3 & 4 & 8 & 9 & 10 \\ 3 & 4 & 11 & 12 & 13 \end{array}$$

**Példa**

$$R\langle X, Y \rangle: \begin{array}{cc} X & Y \\ \hline 1 & 2 \\ 1 & 2 \end{array} \times S\langle Y, Z \rangle: \begin{array}{cc} Y & Z \\ \hline 2 & 3 \\ 4 & 5 \\ 4 & 5 \end{array} = \begin{array}{cccc} X & R.Y & S.Y & Z \\ \hline 1 & 2 & 2 & 3 \\ 1 & 2 & 4 & 5 \\ 1 & 2 & 4 & 5 \\ 1 & 2 & 2 & 3 \\ 1 & 2 & 4 & 5 \\ 1 & 2 & 4 & 5 \end{array}$$

Láthatóan a Descartes-szorzás is csak akkor eredményez valódi multirelációt, ha az operandus-relációk legalább egyike is az.

**Relációk osztása (osztás)**

Valamely  $R\langle \underline{A} \rangle$  reláció osztható egy  $S\langle \underline{B} \rangle$  relációval, ha van olyan  $T\langle \underline{C} \rangle$  reláció, melyre

1.  $C = A \setminus B$ , és
2.  $S\langle \underline{B} \rangle \times T\langle \underline{C} \rangle = R\langle \underline{A} \rangle$ .

Ekkor az  $R\langle \underline{A} \rangle$  és az  $S\langle \underline{B} \rangle$  relációk *hányadosa*:

$$R\langle \underline{A} \rangle \div S\langle \underline{B} \rangle \triangleq T\langle \underline{C} \rangle.$$

Könnyen belátható, hogy minden  $r \in R\langle \underline{A} \rangle$  esetén  $r \upharpoonright_{(A \setminus B)} \in R\langle \underline{A} \rangle \div S\langle \underline{B} \rangle$ .

**Példa**

$$R\langle X, Y, U, V \rangle: \begin{array}{cccc} X & Y & U & V \\ \hline 1 & 2 & 2 & 3 \\ 1 & 2 & 4 & 5 \\ 1 & 2 & 2 & 3 \\ 1 & 2 & 4 & 5 \end{array} \div S\langle U, V \rangle: \begin{array}{cc} U & V \\ \hline 2 & 3 \\ 4 & 5 \end{array} = \begin{array}{cc} X & Y \\ \hline 1 & 2 \\ 1 & 2 \end{array}$$

Láthatóan a hányados csak akkor eredményezhet valódi multirelációt, ha az osztó is az. Az alábbi példából viszont az látható, hogy az osztó multireláció jellege önmagában még nem elég ahhoz, hogy a hányados is az legyen.

**Példa**

$$R\langle X, Y, U, V \rangle: \begin{array}{c|cccc} X & Y & U & V \\ \hline 1 & 2 & 2 & 3 \\ 1 & 2 & 4 & 5 \\ 1 & 2 & 2 & 3 \\ 1 & 2 & 4 & 5 \end{array} \quad \div \quad S\langle Y, U, V \rangle: \begin{array}{c|ccc} Y & U & V \\ \hline 2 & 2 & 3 \\ 2 & 4 & 5 \\ 2 & 2 & 3 \\ 2 & 4 & 5 \end{array} = \begin{array}{c|c} X \\ \hline 1 \end{array}$$

**Természetes összekapcsolás (Natural Join)**

Legyen  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  relációk esetén  $A \cap B = X$ , és  $X \neq \emptyset$  (vagyis legyenek az  $A$  és  $B$  halmazoknak közös attribútumaik, és ezeket ne különböztessük meg aszerint, hogy melyik halmazból vettük). Ekkor  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  relációk *természetes összekapcsolása*:

$$R\langle \underline{A} \rangle \bowtie S\langle \underline{B} \rangle \triangleq \{ r \mid r \in \mathbf{D}\langle \underline{C} \rangle, \underline{C} = \underline{A} \overset{\cup}{+} \underline{B} \},$$

ahol  $\overset{\cup}{+}$  az egyesítő láncolás műveletének jele.

*Megjegyzés*

A természetes összekapcsolás táblájában tehát az  $R\langle \underline{A} \rangle$  és az  $S\langle \underline{B} \rangle$  tábláinak azon sorai vesznek részt, melyek a közös ( $X$ -beli) attribútumokon azonos értékeket tartalmaznak.

Ebben az esetben tehát nem használunk minősített neveket, mivel a természetes összekapcsolás lényegét alkotják az azonos attribútumok.

**Példa**

$$R\langle X, Y, Z \rangle: \begin{array}{c|ccc} X & Y & Z \\ \hline 1 & 2 & 3 \\ 6 & 7 & 8 \\ 9 & 7 & 8 \end{array} \quad \bowtie \quad S\langle Z, Y, U \rangle: \begin{array}{c|ccc} Z & Y & U \\ \hline 3 & 2 & 4 \\ 3 & 2 & 5 \\ 8 & 7 & 9 \end{array} = \begin{array}{c|cccc} X & Y & Z & U \\ \hline 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 5 \\ 6 & 7 & 8 & 9 \\ 9 & 7 & 8 & 9 \end{array}$$

**Példa**

$$R\langle X, Y \rangle: \begin{array}{c|cc} X & Y \\ \hline 1 & 2 \\ 1 & 2 \end{array} \quad \bowtie \quad S\langle Z, Y \rangle: \begin{array}{c|cc} Z & Y \\ \hline 3 & 2 \\ 5 & 4 \\ 5 & 4 \end{array} = \begin{array}{c|ccc} X & Y & Z \\ \hline 1 & 2 & 3 \\ 1 & 2 & 3 \end{array}$$

A Descartes-szorzáshoz hasonlóan itt is csak akkor lesz az eredmény valódi multireláció, ha az operandus-relációk legalább egyike is az.

**Általános összekapcsolás (Théta összekapcsolás, Join)**

Legyen  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  reláció,  $T_{A \cup B}$  pedig egy logikai függvény az  $A \cup B$  attribútumhalmazon (ahol  $\cup$  a pontozott egyesítés műveletének jele). Ekkor az  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  relációknak a  $T_{A \cup B}$  logikai függvényre vonatkozó *általános összekapcsolása*:

$$R\langle \underline{A} \rangle \bowtie_{T_{A \cup B}} S\langle \underline{B} \rangle \triangleq \{ r \mid r \in \mathbf{D}\langle \underline{C} \rangle, \underline{C} = \underline{A} \dot{+} \underline{B}, T_{A \cup B} \}.$$

*Megjegyzés*

Az általános összekapcsolás a természetes összekapcsolásban előírt szigorú attribútum-egyeztési feltétel helyett tetszőleges kapcsolati feltételt megenged a táblák sorainak egyesíté-

séhez. Mivel a különböző relációk esetlegesen egyező nevű attribútumai különböző feltételekben szerepelhetnek, azért természetesen minősített attribútumneveket (ennek érdekében pedig pontozott egyesítést) használunk.

### Példa

Tekintsük az előző példánál bemutatott  $R = R\langle X, Y, Z \rangle$  és az  $S = S\langle Y, Z, U \rangle$  relációk tábláit, valamint a  $T_1 = (X < U)$  és a  $T_2 = (X < U) \wedge (R.Y \neq S.Y)$  logikai függvényeket.

Ekkor az  $R \bowtie_{T_1} S$  reláció táblája:

és az  $R \bowtie_{T_2} S$  reláció táblája:

$X$	$R.Y$	$R.Z$	$S.Y$	$S.Z$	$U$
1	2	3	2	3	4
1	2	3	2	3	5
1	2	3	7	8	10
6	7	8	7	8	10
9	7	8	7	8	10

$X$	$R.Y$	$R.Z$	$S.Y$	$S.Z$	$U$
1	2	3	7	8	10

### Példa

$R\langle X, Y \rangle$ :	<table><tr><th><math>X</math></th><th><math>Y</math></th></tr><tr><td>1</td><td>2</td></tr><tr><td>1</td><td>2</td></tr></table>	$X$	$Y$	1	2	1	2	$\bowtie_{R.Y < S.Y}$	$S\langle Y, Z \rangle$ :	<table><tr><th><math>Y</math></th><th><math>Z</math></th></tr><tr><td>2</td><td>3</td></tr><tr><td>4</td><td>5</td></tr><tr><td>4</td><td>5</td></tr></table>	$Y$	$Z$	2	3	4	5	4	5	=	<table><tr><th><math>X</math></th><th><math>R.Y</math></th><th><math>S.Y</math></th><th><math>Z</math></th></tr><tr><td>1</td><td>2</td><td>4</td><td>5</td></tr><tr><td>1</td><td>2</td><td>4</td><td>5</td></tr><tr><td>1</td><td>2</td><td>4</td><td>5</td></tr><tr><td>1</td><td>2</td><td>4</td><td>5</td></tr></table>	$X$	$R.Y$	$S.Y$	$Z$	1	2	4	5	1	2	4	5	1	2	4	5	1	2	4	5
$X$	$Y$																																							
1	2																																							
1	2																																							
$Y$	$Z$																																							
2	3																																							
4	5																																							
4	5																																							
$X$	$R.Y$	$S.Y$	$Z$																																					
1	2	4	5																																					
1	2	4	5																																					
1	2	4	5																																					
1	2	4	5																																					

Az eredmény nyilván itt csak akkor lesz valódi multireláció, ha az operandus-relációk legalább egyike is az volt.

## Relációműveletek kapcsolata

A természetes és az általános összekapcsolás olyan összetett relációs művelet, mely felépíthető Descartes-szorzásból és szelekcióból. A Descartes-szorzás hátránya, hogy az eredménytábla rettenetesen megnőhet, mivel mérete (sorainak száma) az operandus-táblák méreteinek szorzata. Célszerű ezért egyrészt a különböző szelekciós műveleteket (ha csak lehet) még az operandus-táblákon elvégezni, másrészt pedig Descartes-szorzás helyett természetes, vagy általános összekapcsolással a műveletből már eleve kizárni azokat az operandus-táblabeli sorokat, melyek egy kapcsolati feltételnek nem felelnek meg. (Descartes-szorzás használata esetén e feltételeknek nem megfelelő sorokat egy utólagos szelekcióval törölnénk, persze már egy jóval nagyobb táblából.)

A gyakorlatban még fontosabb a természetes összekapcsolás helyettesítése általánossal. Az előbbinek a veszélye abban rejlik, hogy a kapcsolat implicit (nem látható), egyszerűen a két tábla bizonyos oszlopainak azonos jelölésén (nevén) alapszik. Így utólag, pusztán „ránézésre” esetleg nem is vesszük észre, hogy a két tábla hogy is kapcsolódik. Súlyos problémát okoz, ha a kapcsolatban résztvevő valamelyik tábla egyik oszlopát átnevezzük, mivel ekkor a természetes összekapcsolás már nem is „működik”. Hasonlóan hibát okoz, ha nem akarunk az összes egyező nevű attribútum mentén kapcsolatot teremteni (például azért, mert az azonos nevű attribútumok egy része az egyik táblában mást jelent, mint a másikban). Ha két táblát össze kell kapcsolnunk, akkor tehát használjunk általános összekapcsolást. Ezt „támogatja” az SQL azáltal, hogy a természetes összekapcsolást nem is valósítja meg. A relációműveletek közötti legfontosabb kapcsolatok az alábbiak.

1. Az általános összekapcsolás és a Descartes-szorzat kapcsolata

Tetszőleges  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  relációk és  $T_{A \cup B}$  logikai függvény esetén:

$$R\langle \underline{A} \rangle \bowtie_{T_{A \cup B}} S\langle \underline{B} \rangle = \sigma_{T_{A \cup B}}(R\langle \underline{A} \rangle \times S\langle \underline{B} \rangle), \text{ és így nyilván}$$

$$(R\langle \underline{A} \rangle \bowtie_{T_{A \cup B}} S\langle \underline{B} \rangle) \subseteq (R\langle \underline{A} \rangle \times S\langle \underline{B} \rangle).$$

2. A természetes összekapcsolás és a Descartes-szorzat kapcsolata

Legyen  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  relációk esetén  $A \cap B \neq \emptyset$ . Ekkor

$$R\langle \underline{A} \rangle \bowtie S\langle \underline{B} \rangle = \sigma_{T_{A \cup B}}(R\langle \underline{A} \rangle \times S\langle \underline{B} \rangle).$$

3. Az általános és a természetes összekapcsolás kapcsolata

Legyen  $R\langle \underline{A} \rangle$  és  $S\langle \underline{B} \rangle$  relációk esetén  $A \cap B \neq \emptyset$ . Ekkor

$$R\langle \underline{A} \rangle \bowtie_{T_{A \cup B}} S\langle \underline{B} \rangle = R\langle \underline{A} \rangle \bowtie S\langle \underline{B} \rangle.$$

A fentiekben a  $T_{A \cup B}$  logikai függvény például lehet a következő alakú:

$$T_{A \cup B} = (R.C_1 = S.C_1) \wedge (R.C_2 = S.C_2) \wedge \dots \wedge (R.C_n = S.C_n), \text{ melyben}$$

$$R.C_1, R.C_2, \dots, R.C_n, S.C_1, S.C_2, \dots, S.C_n \in A \cup B.$$

## Osztályozó műveletek

A szerkezetmódosító műveletekhez hasonlóan az osztályozó műveletek – melyek az algebrai osztályozáson, pontosabban az attribútum szerinti sémafelosztáson alapulnak – szintén módosítják a paraméter-reláció szerkezetét, ám ezt egészen másként teszik. Míg a korábban bemutatott szerkezetmódosító műveletek az eredmény-reláció szerkezetét a paraméter-reláció adataitól függetlenül, kizárólag a művelet jellegéből adódóan határozták meg, addig az osztályozó műveletek végrehajtásának módja alapvetően a paraméter-reláció adataitól függ. Ha ugyanis valamely kiválasztott attribútum mentén az egyes rekordok tartalmazznak különböző értékeket, akkor kialakulhat egy többosztályos felosztás, egyébként pedig nem.

Az osztályozó műveleteket, ha valamilyen aritmetikai művelet elvégzésére használjuk, nevezhetjük *relációaritmetikai műveleteknek* is. Nem tartoznak a hagyományos relációalgebrai műveletek közé, mégis célszerű őket bevezetni, mivel igen gyakran van szükségünk olyan kiválasztási, vagy lekérdezési tevékenységre, melynek során egy táblában bizonyos tulajdonságú sorok valamely attribútumait összegezzük, átlagoljuk, stb.

Az osztályozó műveletekben szereplő függvényeket az adatbázis-kezelő szoftverek tárgyalásánál *osztályfüggvényeknek*, *többsoros függvényeknek*, vagy *csoportfüggvényeknek* nevezzük, jelezve ezzel azt, hogy egyrészt a műveletek jellegéből adódóan tetszőleges számú operandusuk lehet, másrészt, hogy ezek az operandusok jellemzően egy adott tábla több sorának (rekordjának) azonos oszlopához (attribútumához) tartoznak.

A függvények másik csoportját alkotják az úgynevezett *egysoros függvények*, melyeket másnéven unáris, vagy egyoperandusú műveleteknek is hívunk. Ilyenek a különböző adatkonvertáló függvények, az előjel-, vagy az abszolútérték-függvények, stb. Ezek tetszőleges matematikai kifejezés tagjai lehetnek, a táblák szerkezetét nem befolyásolják, így – bár használni fogjuk őket – részletes tárgyalásukra nincs szükség.

Az osztályfüggvények alkalmazásához jelentek meg az SQL nyelv résztábla-kiválasztást lehetővé tevő SELECT utasításában a GROUP BY és a HAVING paraméterek. Ezek hasz-

nálatának megtanulását nehezíti, hogy egy összetett SELECT utasításban az osztályfüggvények és ezek a paraméterek általában igen távol kerülnek egymástól, és a kapcsolatuk, az alkalmazásuk módja sem nyilvánvaló.

Az osztályozó műveletek felírásánál a korábban bevezetett attribútum szerinti sémafelosztás jelölésétől egy kissé el fogunk térni éppen azért, hogy a fenti SQL utasításokkal minél jobban összhangban lehessünk. Mielőtt azonban megadnánk az osztályozó műveletek általános alakját, nézzünk egy példát egy gyakorlati osztályozási feladatra és megoldásának gondolatmenetére!

## Bevezető példa

Legyen az *eladás*(SZALON, ÉV, BEVÉTEL, MÁRKA) reláció táblája az alábbi:

SZALON	ÉV	BEVÉTEL	MÁRKA
Pipacs	1999	20000	Jaguar
Pipacs	1999	700	Mercedes
Pipacs	2000	14000	Jaguar
Pipacs	2000	2000	Mercedes
Pipacs	2001	10300	Jaguar
Pipacs	2001	5000	Mercedes
Szekér	1999	15000	Porsche
Szekér	1999	8700	Opel
Szekér	2000	12500	Porsche
Szekér	2000	22300	Opel
Szekér	2001	10500	Porsche
Szekér	2001	28000	Opel

### Feladat

Készítsünk kimutatást, mely szalononkénti és évenkénti osztályozásban tartalmazza azon szalonok bevételeit (ezer euro-ban), melyeknek éves bevétele meghaladja a 20000-et!

### Megoldás

A megoldást nyilván az alábbi táblájú *kimutatás*(SZALON, ÉV, ÖSSZBEVÉTEL) eredményreláció adja:

SZALON	ÉV	ÖSSZBEVÉTEL
Pipacs	1999	20700
Szekér	1999	23700
Szekér	2000	34800
Szekér	2001	38500

Hogyan juthatunk el ehhez a megoldáshoz algebrai úton?

### A megoldás módszere

Először kiválasztjuk a feladat relációjának (táblájának) a megoldás szempontjából fontos alrelációját (altábláját) annak érdekében, hogy kisebb legyen az elvégzendő feladat. A fentiekben például az eladott gépkocsik márkája érdektelen. Az első lépés tehát az *alreláció-kiválasztás* (altábla-kiválasztás). Második lépésként a SZALON és az ÉV attribútumok értékei szerint részrelációkra, vagyis rekord-osztályokra (résztáblákra, azaz sor-osztályokra) bontjuk a kapott alrelációt (*osztályozási* tevékenység), majd harmadik lépésként az így kapott részrelációk rekordjainak BEVÉTEL attribútumán részrelációként kiszámítjuk az ÖSSZ-

BEVÉTEL értéket előállító *Összegzés* osztályfüggvényt. Tehát az osztályozás után minden részrelációt egyetlen rekorddal (úgynevezett *reprezentáns-rekorddal*) kell helyettesíteni (ezt *faktorizációnak* nevezzük). Végül a negyedik lépés a reprezentáns-rekordok megszürése megadott osztálysűrő feltétellel (ez az *osztályvizsgálat*). Az alábbiakban először elvileg tekintjük át ezeket a lépéseket, majd a gyakorlatban is elvégezzük őket.

### 1. lépés (Alreláció-kiválasztás)

Az *alreláció-kiválasztás* egy vetítési művelet (projekció), melynek során a vizsgált relációból (a paraméter-relációból) elhagyjuk mindazon attribútum-értékeket, melyekre a feladat megoldása során, vagy végeredménye szempontjából nincs szükségünk (tehát az altábla-kiválasztás oszlopelhagyást jelent a paraméter-táblából).

### 2. lépés (Osztályozás)

Az *osztályozás* célja egy halmaznak valamilyen szempontból azonos tulajdonságú elemei szerinti diszjunkt (egymást át nem fedő) részhalmazokra bontása. Eredményeként a halmaz olyan osztályozása jön létre, melyben az egy osztályon belül levő elemek (és csak azok) egyenlők az osztályozási tulajdonság szerint. Speciális esetben előfordulhat, hogy minden osztály egyelemű (ezt triviális osztályozásnak nevezzük). A gyakorlatban ezzel ekvivalens, ha egyáltalán nem adunk meg osztályozási tulajdonságot. Esetünkben a halmaz elemei (a paraméter-reláció rekordjai) egy osztályba kerülnek, ha a SZALON és az ÉV attribútum-értékekben megegyeznek.

### 3. lépés (Faktorizáció)

A *faktorizáció* során az osztályokra (diszjunkt részhalmazokra) bontott halmazban az egyes osztályokat helyettesítjük egy úgynevezett *reprezentáns elemmel*. Ezeket azért nevezzük reprezentáns elemeknek, mert azt a tulajdonságot reprezentálják, melyben az általuk kiváltott osztály elemei megegyeznek. Feltehető a kérdés, hogy a relációalgebrában a reprezentáns elemek, mint rekordok milyen attribútumokkal rendelkeznek. A faktorizáció szempontjából a paraméter-reláció attribútumai az alábbi négy típusba sorolhatók:

- *Osztálytag-attribútumok*  
Azok az attribútumok, melyekre vonatkozóan az egy osztálybeli rekordok azonosak.
- *Osztályképző-attribútumok*  
Az osztálytag-attribútumoknak egy olyan részhalmaza, mely még alkalmas a szükséges osztályozás elvégzésére. Célszerű az ilyen tulajdonságú attribútumok minimális halmazát használni osztályozásra.
- *Osztályfüggvény-paraméter attribútum*  
Az alkalmazott osztályfüggvény ebből, mint paraméterből állítja elő az osztályfüggvény értéket, az úgynevezett *osztályfüggvény-attribútumot*.
- *Osztályidegen attribútumok*  
Ezek az eredeti reláció azon attribútumai, melyek nem osztálytagok (tehát egy osztályon belül is különböző értékeket vehetnek fel). Ezeket az attribútumokat (mivel nem reprezentálják az osztály elemeit) nem tartalmazzák a reprezentáns rekordok. Nyilván az osztályfüggvény-paraméter attribútum is osztályidegen.

A reprezentáns rekordokból álló eredmény-reláció tehát csak a paraméter-reláció *osztályképző-attribútumait* és az *osztályfüggvény-attribútumot* tartalmazhatja. Mást nem, hiszen a paraméter-relációból származtattuk (és legalább egy attribútuma biztos van, különben üres). De vajon kell-e tartalmaznia mind a kétféle attribútumot?

- *1. eset: Hiányoznak az eredményből az osztályképző-attribútumok.*  
Mivel az osztályozást általában éppen az azonos tulajdonságú osztályok valamilyen osztályfüggvény-értékének meghatározása miatt végezzük, az eredmény-relációból hiányozhatnak az osztályképző-attribútumok. Használatuk mégis célszerű, mivel ezek nélkül igen nehézkes az osztályfüggvény értékeinek értelmezése (hacsak nem egyetlen érték az osztályozó művelet eredménye).
- *2. eset: Hiányzik az eredményből az osztályfüggvény-attribútum.*  
Ha valamilyen (például az osztályfüggvényre vonatkozó) osztálysűrővel megszürt reprezentáns rekordok osztálytag-attribútumai egy részének a listázására van szükségünk, akkor előfordul, hogy e listából kihagyjuk az osztályfüggvény értékét.  
Ha azonban nem használunk osztálysűrőt, akkor az eredmény-relációt osztályozó művelet nélkül is megkaphatjuk (SELECT utasítás esetén a megfelelő oszlopokat tartalmazó altábla kiválasztásával és a DISTINCT paraméter használatával).

Megjegyezzük, hogy triviális osztályozás esetén a faktorizáció azt eredményezi, hogy a paraméter-reláció minden rekordját önmaga reprezentálja. Tehát például, ha nem adunk meg osztályozási tulajdonságot, akkor a faktorizáció eredménye az eredeti paraméter-reláció lesz.

#### 4. lépés (Osztályvizsgálat)

Az osztályvizsgálat során a faktorizáció eredményeként kapott reprezentáns-rekordok közül azokat tartjuk meg, melyek kielégítik a – valamilyen logikai függvényrel megadott – osztálysűrő feltételt. Ezt röviden csak osztálysűrőnek nevezzük. Ez az osztálysűrő gyakran tartalmazza magát az osztályfüggvényt. Speciális esetként előfordulhat, hogy nem adunk meg osztálysűrőt. Ekkor teljesen elmarad az osztályvizsgálat, és nyilván a faktorizáció eredményeként kapott minden reprezentáns-rekord bekerül az eredményrelációba.

#### A megoldás gyakorlati lépései

Az alábbiakban bemutatjuk, hogy az osztályozó művelet végrehajtásának egyes lépéseit milyen módon alkalmazzuk a gyakorlatban. Ennek megfelelően az alábbi leírásban reláció helyett táblát, attribútum helyett oszlopot és rekord helyett sort fogunk mondani.

##### 1. lépés (Altábla-kiválasztás)

Először létrehozuk az *eladás* tábla *átmeneti* altábláját, mely csak a számunkra szükséges osztálytag-oszlopokat és az osztályfüggvény-paraméter oszlopot tartalmazza.

Az alábbiakban látható ennek az *átmeneti*(SZALON, ÉV, BEVÉTEL) relációnak a táblája.

SZALON	ÉV	BEVÉTEL
Pipacs	1999	20000
Pipacs	1999	700
Pipacs	2000	14000
Pipacs	2000	2000
Pipacs	2001	10300
Pipacs	2001	5000
Szekér	1999	15000
Szekér	1999	8700
Szekér	2000	12500
Szekér	2000	22300
Szekér	2001	10500
Szekér	2001	28000

## 2. lépés (Osztályozás)

A ⟨SZALON, ÉV⟩ osztályozás eredménye:

osztályok	SZALON	ÉV	BEVÉTEL
1.	Pipacs	1999	20000
	Pipacs	1999	700
2.	Pipacs	2000	14000
	Pipacs	2000	2000
3.	Pipacs	2001	10300
	Pipacs	2001	5000
4.	Szekér	1999	15000
	Szekér	1999	8700
5.	Szekér	2000	12500
	Szekér	2000	22300
6.	Szekér	2001	10500
	Szekér	2001	28000

az ⟨ÉV, SZALON⟩ osztályozás eredménye:

osztályok	SZALON	ÉV	BEVÉTEL
1.	Pipacs	1999	20000
	Pipacs	1999	700
2.	Szekér	1999	15000
	Szekér	1999	8700
3.	Pipacs	2000	14000
	Pipacs	2000	2000
4.	Szekér	2000	12500
	Szekér	2000	22300
5.	Pipacs	2001	10300
	Pipacs	2001	5000
6.	Szekér	2001	10500
	Szekér	2001	28000

A feladat-kijelölés szerint az osztályképző-oszlopok vektora ⟨SZALON, ÉV⟩. Az osztályozás szempontjából nem lényeges, hogy azt az oszlopok milyen sorrendje szerint végezzük. Láthatóan a kétféle osztályozás eredménytáblájában ugyanazok az osztályok jönnek létre, csak a sorrendjük más. Mivel azonban a relációkat vektorhalmazoknak tekintettük, a rekordok (vagyis a táblabeli sorok) sorrendjének nincs elvi jelentősége.

## 3. lépés (Faktorizáció)

Az osztályhelyettesítő reprezentáns-sorok a kiválasztott osztályképző-oszlopok értékein kívül az egyes osztályokhoz tartozó osztályfüggvény értéket tartalmazzák.

Legyen az eredménytábla neve *kimutatás*, az eredményvektor pedig ⟨SZALON, ÉV, ÖSSZBEVÉTEL⟩. Ekkor a faktorizáció eredménye:

SZALON	ÉV	ÖSSZBEVÉTEL
Pipacs	1999	20700
Pipacs	2000	16000
Pipacs	2001	15300
Szekér	1999	23700
Szekér	2000	34800
Szekér	2001	38500

## 4. lépés (Osztályvizsgálat)

SZALON	ÉV	ÖSSZBEVÉTEL
Pipacs	1999	20700
Szekér	1999	23700
Szekér	2000	34800
Szekér	2001	38500

A feladat osztályszűrő feltétele szerint csak azon szalonok bevételeire van szükségünk, melyeknek éves bevétele meghaladja a 20000-et. Ez a fenti tábla nem megfelelő sorainak kiszűrését jelenti. Így már valóban a korábban felírt eredmény-táblát kaptuk.

## Az osztályozó műveletek általános alakja

Az osztályozó művelet is egy reláció-transzformáció, mely egy paraméter-relációt leképez egy eredmény-relációba.

Az osztályozó műveleteket oly módon definiáljuk, hogy egyetlen összetartozó szerkezet tartalmazza

- magát az osztályfüggvényt (paramétereként egy attribútummal),
- a művelet paraméter-relációját (azt a relációt, amelyre alkalmazzuk az osztályozó műveletet, és amelynek egy attribútuma az osztályfüggvény paramétere),
- azoknak az attribútumoknak a vektorát, melyekre vonatkozóan az osztályképzést elvégezzük,
- valamint azokat a feltételeket, melyeknek eleget tevő osztályok a művelet eredményében szerepelhetnek.

Az osztályozó műveletek általános alakja:

$$OP(p)[R(\underline{A}) \mid \underline{B}] \downarrow_T \Rightarrow \underline{C},$$

ahol

- $OP$ : az osztályfüggvény neve  
(megjegyezzük, hogy az Oracle-rendszerben az osztályfüggvényt többsoros-függvénynek nevezik),
- $R(\underline{A})$ : a *paraméter-reláció* (melyre a műveletet alkalmazzuk),
- $\underline{B}$ : az osztályképző-attribútumok vektora, ahol  $B \subseteq B^*$ , melyben  $B^*$  az osztálytag-attribútumok halmaza (nyilván  $B^* \subseteq A$ ),
- $p$ : a *paraméter-attribútum*, vagyis az  $OP$  művelet paramétere, ahol  $p \in P$ , melyben  $P = A \setminus B^*$ , vagy  $p$  egy  $P$ -beli attribútumkifejezés,
- $\underline{C}$ : az *eredmény-reláció attribútumvektora*, ahol  $C \subseteq B \cup \{OP\}$ , melyben az  $OP$  az osztályfüggvény eredményeit tartalmazó *osztályfüggvény-attribútum* (ez az osztályfüggvény neve), végül
- $T$ : az *osztályszűrő*, mely egy logikai függvény (osztálykritérium) a  $C$  attribútum-halmaz felett (az előtte szereplő „ $\downarrow$ ” szimbólum azt jelzi, hogy az osztályszűrő a teljes előtte álló kifejezés, vagyis az osztályfüggvény eredményére vonatkozik).

### Megjegyzés

- Az eredmény-relációból hiányozhatnak az osztályképző attribútumok, vagy hiányozhat az osztályfüggvény-attribútum, de legalább egy attribútumnak szerepelnie kell.
- Ha az eredmény-reláció attribútumvektorában szerepel az osztályfüggvény-attribútum, akkor csak az osztályfüggvény neve kerülhet be, de a paramétere már nem.
- Ha az osztályszűrő hivatkozik az osztályfüggvényre, akkor ebben az esetben is csak annak neve kerülhet be, de a paramétere már nem.
- Az osztályképző-attribútumok vektora opcionális paraméter.  
Hiánya esetén az eredmény-reláció minden rekordja osztály-reprezentáns rekordként viselkedik. Ekkor az osztályozó műveletek alakja:

$$OP(p)[R(\underline{A})] \downarrow_T \Rightarrow \underline{C}.$$

- A  $T$  osztálysűrítő opcionális paraméter.  
Hiánya esetén nincs megszorítás az osztályképzés eredmény-relációjára.  
Ekkor az osztályozó műveletek alakja:

$$OP(p)[R(\underline{A}) \mid \underline{B}] \Rightarrow \underline{C}.$$

### A bevezető példa folytatása

Írjuk fel az alfejezet bevezető példáját az osztályozó művelet fent bevezetett jelölésével!

*Megoldás*

A feladat paraméterei tehát:

$$\begin{aligned}\underline{A} &= \langle \text{szalon}, \text{év}, \text{bevétel}, \text{márka} \rangle, \\ R(\underline{A}) &= \text{eladás}(\text{szalon}, \text{év}, \text{bevétel}, \text{márka}), \\ \underline{B} &= \langle \text{szalon}, \text{év} \rangle, \\ p &= \text{bevétel}, \\ OP &= \text{Sum}, \\ \underline{C} &= \langle \text{szalon}, \text{év}, \text{Sum} \rangle, \\ T &= (\text{Sum} > 20000)\end{aligned}$$

Igy a megoldás:

$$\begin{aligned}\text{kimutatás}(\text{SZALON}, \text{ÉV}, \text{Sum}) &= \\ \text{Sum}(\text{BEVÉTEL}) & \\ [ \text{eladás}(\text{SZALON}, \text{ÉV}, \text{BEVÉTEL}, \text{MÁRKA}) \mid \langle \text{SZALON}, \text{ÉV} \rangle & \\ ]_{(\text{Sum} > 20000)} \Rightarrow \langle \text{SZALON}, \text{ÉV}, \text{Sum} \rangle . &\end{aligned}$$

Ha figyelembe vesszük, hogy a bevezető példában az osztályfüggvény eredmény-attribútumát „ÖSSZBEVÉTEL” módon jelöltük, akkor egy átnevezőfüggvényt is alkalmaznunk kell.

Ekkor a megoldás:

$$\begin{aligned}\text{kimutatás}(\text{SZALON}, \text{ÉV}, \text{ÖSSZBEVÉTEL}) &= \\ Ren_{\text{kimutatás}(\text{SZALON}, \text{ÉV}, \text{ÖSSZBEVÉTEL})} & \\ (\text{Sum}(\text{BEVÉTEL}) & \\ [ \text{eladás}(\text{SZALON}, \text{ÉV}, \text{BEVÉTEL}, \text{MÁRKA}) \mid \langle \text{SZALON}, \text{ÉV} \rangle & \\ ]_{(\text{Sum} > 20000)} \Rightarrow \langle \text{SZALON}, \text{ÉV}, \text{Sum} \rangle & \\ ) . &\end{aligned}$$

Ha átnevezőfüggvény helyett az „AS” átnevező-operátort használjuk, akkor

$$\begin{aligned}\text{kimutatás}(\text{SZALON}, \text{ÉV}, \text{ÖSSZBEVÉTEL}) &= \\ \text{Sum}(\text{BEVÉTEL}) & \\ [ \text{eladás}(\text{SZALON}, \text{ÉV}, \text{BEVÉTEL}, \text{MÁRKA}) \mid \langle \text{SZALON}, \text{ÉV} \rangle & \\ ]_{(\text{Sum} > 20000)} \Rightarrow \langle \text{SZALON}, \text{ÉV}, \text{Sum AS ÖSSZBEVÉTEL} \rangle . &\end{aligned}$$

Miután ez az alak egyszerűbb és áttekinthetőbb, ezért általában ezt használjuk.

## Osztályozó műveletek kompozíciója

Lehetőség van arra, hogy egy osztályképzés során több osztályfüggvényt is alkalmazzunk. Ennek természetesen az a feltétele, hogy az osztályképzés azonos legyen. Alkalmazására tipikus példa, amikor átlagot és szórást kell számítani. Az osztályozó műveletek kompozíciójának értelmezése:

$$(OP_1(p)[R(\underline{A}) \mid \underline{B}] \downarrow_{T_1} \Rightarrow \underline{C}_1) \otimes (OP_2(p)[R(\underline{A}) \mid \underline{B}] \downarrow_{T_2} \Rightarrow \underline{C}_2) = \\ (OP_1(p), OP_2(p)[R(\underline{A}) \mid \underline{B}] \downarrow_T \Rightarrow \underline{C}),$$

ahol

$\otimes$  jelöli a műveletkompozíciót,

$T = T_1 \wedge T_2$ , melyben  $\wedge$  a logikai és művelet jele, és

$\underline{C}$  pedig a  $\underline{C}_1$  és  $\underline{C}_2$  attribútumvektorokból összeállított olyan vektor, melyben mindkettő attribútumai szerepelnek, de az azonosak csak egyszer (tehát  $C = C_1 \cup C_2$ ).

### Példa

Írjuk fel az előző példát oly módon, hogy az egyes osztályokra az összegén kívül az átlagot is meghatározzuk!

*Megoldás*

$$\text{kimutatás}(\langle \text{SZALON}, \text{ÉV}, \text{ÖSSZBEVÉTEL}, \text{ÁTLAG} \rangle = \\ \text{Sum}(\text{BEVÉTEL}), \text{Avg}(\text{BEVÉTEL}) \\ [eladás(\langle \text{SZALON}, \text{ÉV}, \text{BEVÉTEL}, \text{MÁRKA} \rangle \mid \langle \text{SZALON}, \text{ÉV} \rangle \\ ] \downarrow_{(Sum > 20000)} \Rightarrow \langle \text{SZALON}, \text{ÉV}, \text{Sum AS ÖSSZBEVÉTEL}, \text{Avg AS ÁTLAG} \rangle.$$

Eredményként ekkor az alábbi táblát kapjuk:

SZALON	ÉV	ÖSSZBEVÉTEL	ÁTLAG
Pipacs	1999	20700	10350
Szekér	1999	23700	11850
Szekér	2000	34800	17400
Szekér	2001	38500	19250

## Egyszerű lekérdezések

Az alábbiakban néhány olyan mintapéldát mutatunk be, melyek ugyan egyszerű alapfeladatokat oldanak meg, mégis jól szemléltetik a relációalgebrai műveletek használatát.

### 1. Példa

Legyen  $\text{film}(\langle \text{CÍM}, \text{ÉV}, \text{HOSSZ}, \text{STÚDIÓ} \rangle)$  egy reláció.

*Feladat*

Melyek a Fox stúdióban készült, legalább 100 perc hosszúságú filmek, és ezek, mikor készültek?

## a.) Megoldás

$$\pi_{\text{CÍM, ÉV}}(\sigma_{\text{HOSSZ} \geq 100}(\text{film}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle) \cap \sigma_{\text{STÚDIÓ} = \text{'Fox'}}(\text{film}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle)) .$$

## b.) Megoldás

$$\pi_{\text{CÍM, ÉV}}(\sigma_{(\text{HOSSZ} \geq 100) \wedge (\text{STÚDIÓ} = \text{'Fox'})}(\text{film}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle)) .$$

## Megjegyzés

A két egyenértékű megoldás közül nyilván a második a hatékonyabb, hiszen gyorsabb egy táblán egy összetett feltételt vizsgálni, mint metszet művelettel két tábla közös sorait keresni (amely két táblán még külön is kell egy-egy kiválasztást végezni).

## 2. Példa

Legyen  $\text{film1}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle$ , és  
 $\text{film2}\langle \text{CÍM, ÉV, SZÍNÉSZ} \rangle$  két reláció.

## Feladat

Kik azok a színészek, akik legalább 100 perc hosszú filmekben szerepelnek

## a.) Megoldás

$$\pi_{\text{SZÍNÉSZ}}(\sigma_{\text{HOSSZ} \geq 100}(\text{film1}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle) \bowtie \text{film2}\langle \text{CÍM, ÉV, SZÍNÉSZ} \rangle)) .$$

## b.) Megoldás

$$\pi_{\text{SZÍNÉSZ}}(\sigma_{\text{HOSSZ} \geq 100}(\text{film1}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle) \bowtie \text{film2}\langle \text{CÍM, ÉV, SZÍNÉSZ} \rangle)) .$$

## Megjegyzés

Ezúttal is a második megoldás a hatékonyabb, mivel a természetes összekapcsolást így kisebb (a  $\text{HOSSZ} \geq 100$  feltételű kiválasztással csökkentett méretű) táblával kell elvégezni.

## 3. Példa

Legyen  $\text{szállító}\langle \text{CÉG, CÍM, TELEFON, ÜGYINTÉZŐ, TERMÉK, ÁR} \rangle$ , és  
 $\text{rendelés}\langle \text{CÉG, TERMÉK, MENNYISÉG} \rangle$  két reláció.

## Feladat

Kik azok az ügyintézők, és mi a telefonszámuk, akiknek a cégétől 100 eFt-nál nagyobb értékben vásároltak valamilyen árut?

## Megoldás

$$\begin{aligned} & \text{Sum}(\text{MENNYISÉG} * \text{ÁR}) \\ & [ \text{rendelés}\langle \text{CÉG, TERMÉK, MENNYISÉG} \rangle \bowtie \\ & \quad \text{szállító}\langle \text{CÉG, CÍM, TELEFON, ÜGYINTÉZŐ, TERMÉK, ÁR} \rangle \mid \langle \text{CÉG, TERMÉK} \rangle \\ & ] \Big|_{(\text{Sum} > 100000)} \Rightarrow \langle \text{ÜGYINTÉZŐ, TELEFON} \rangle . \end{aligned}$$

## Megjegyzés

Az eredmény olyan osztálytag-attribútumokból áll, melyek egyike sem osztályképző.

## 4. Példa

Legyen egy adatbázis modellje:

$\text{film1}\langle \text{CÍM, ÉV, HOSSZ, STÚDIÓ} \rangle$ ,  
 $\text{film2}\langle \text{CÍM, ÉV, SZÍNÉSZ} \rangle$ , és  
 $\text{színészek}\langle \text{SZÍNÉSZ, ÉV, KOR, SZEREPSZÁM} \rangle$ .

**Feladat**

Listázza ki a legalább 100 perces filmek színészeinek nevét és korát a film készítésekor!

**a.) Megoldás**

$$\pi_{\text{SZÍNÉSZ}, \text{színészek.KOR} - (\text{színészek.ÉV} - \text{film1.ÉV}) \text{ AS KORA}}$$

$$(\sigma_{\text{HOSSZ} \geq 100}(\text{film1}\langle \text{CÍM}, \text{ÉV}, \text{HOSSZ}, \text{STÚDIÓ} \rangle)) \bowtie$$

$$\text{film2}\langle \text{CÍM}, \text{ÉV}, \text{SZÍNÉSZ} \rangle \bowtie$$

$$\text{színészek}\langle \text{SZÍNÉSZ}, \text{ÉV}, \text{KOR}, \text{SZEREPSZÁM} \rangle,$$

$$\text{ahol } T = (\text{film2.SZÍNÉSZ} = \text{színészek.SZÍNÉSZ}).$$

**Megjegyzés**

A *film2.SZÍNÉSZ*, valamint a *színészek.SZÍNÉSZ* kifejezésekben a korábban már bevezetett úgynevezett *pontozott jelölést*, azaz *minősített nevet* alkalmaztuk.

A fenti példa jól mutatja azt az esetet, amikor nem szabad természetes összekapcsolást használni. Ha a második összekapcsolásban természetes összekapcsolást használnánk, akkor automatikusan létrejönne a *film2.ÉV = színészek.ÉV* kapcsolat is, holott ez helytelen lenne. A *film2* relációban ugyanis az *ÉV* nyilván a film megjelenésének éve, míg a *színészek* relációban az *ÉV* a színészek aktuális évbéli adatait tartalmazzák. Az alábbiakban megmutatjuk a megoldást arra az esetre, ha teljesen elhagyjuk a természetes összekapcsolást. (Ez már csak azért is indokolt, mivel a gyakorlati adatbázis-kezelő rendszerek nem is valósítják meg a természetes összekapcsolást, csak az általános összekapcsolást.)

**b.) Megoldás**

$$\pi_B(\sigma_{T_1}(\text{film1}\langle \text{CÍM}, \text{ÉV}, \text{HOSSZ}, \text{STÚDIÓ} \rangle)) \bowtie_{T_2}$$

$$\text{film2}\langle \text{CÍM}, \text{ÉV}, \text{SZÍNÉSZ} \rangle \bowtie_{T_3}$$

$$\text{színészek}\langle \text{SZÍNÉSZ}, \text{ÉV}, \text{KOR}, \text{SZEREPSZÁM} \rangle),$$

ahol

$$B = \langle \text{film2.SZÍNÉSZ}, \text{színészek.KOR} - (\text{színészek.ÉV} - \text{film1.ÉV}) \text{ AS KORA} \rangle,$$

$$T_1 = (\text{film1.HOSSZ} \geq 100),$$

$$T_2 = (\text{film1.CÍM} = \text{film2.CÍM}) \wedge (\text{film1.ÉV} = \text{film2.ÉV}),$$

$$T_3 = (\text{film2.SZÍNÉSZ} = \text{színészek.SZÍNÉSZ}).$$

**5. Példa**

Tekintsük az előző példa adatbázis modelljét.

**Feladat**

Adja meg a legalább 100 perces filmek színészeinek átlagéletkorát a film készítésekor!

**Megoldás**

$$\text{Avg(KORA)} [\pi_{\text{SZÍNÉSZ}, \text{színészek.KOR} - (\text{színészek.ÉV} - \text{film1.ÉV}) \text{ AS KORA}}$$

$$(\sigma_{\text{HOSSZ} \geq 100}(\text{film1}\langle \text{CÍM}, \text{ÉV}, \text{HOSSZ}, \text{STÚDIÓ} \rangle)) \bowtie$$

$$\text{film2}\langle \text{CÍM}, \text{ÉV}, \text{SZÍNÉSZ} \rangle \bowtie$$

$$\text{színészek}\langle \text{SZÍNÉSZ}, \text{ÉV}, \text{KOR}, \text{SZEREPSZÁM} \rangle)$$

$$] \Rightarrow \langle \text{Avg AS ÁTLAGÉLETKOR} \rangle,$$

ahol  $T = (\text{film2.SZÍNÉSZ} = \text{színészek.SZÍNÉSZ})$ .

#### Megjegyzés

Ez egy olyan speciális esete az osztályozó műveleteknek, melyben

- hiányzik az osztályképző-attribútum vektor (nincs rá szükség, mivel a teljes paraméter-reláción kiszámítjuk az osztályfüggvényt),
- hiányzik az osztályszűrő (nincs rá szükség, mivel a paraméter-relációban szereplő kiválasztás felétele – a  $T$  logikai függvény – már elvégezte a megfelelő szűrést),
- az eredményben csak az osztályfüggvény-attribútum szerepel (nincs szükség értelmező osztályképző-attribútumokra, mivel a feladat csupán egyetlen szám meghatározása volt).

### 6. Példa

Legyen egy adatbázis modellje:

$kézirat \langle \text{CÍM, SZERZŐ, TÉMA} \rangle$ ,  
 $könyv \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle$ ,  
 $könyvkiadó \langle \text{KIADÓ, ORSZÁG, POSTA_CÍM, TELEFON} \rangle$ .

#### Feladat

Listázza ki a krimi-kiadók nevét és postai címét az ország megjelölésével. Az elkészült lista a  $\langle \text{KIADÓ, CÍME, ORSZÁG} \rangle$  fejléccet tartalmazza. Megjegyezzük, hogy egy kiadót "krimi-kiadónak" nevezünk, ha legalább egy krimi kiadott.

#### Megoldás

$\pi_B(\sigma_T(kézirat \langle \text{CÍM, SZERZŐ, TÉMA} \rangle) \bowtie$   
 $könyv \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle \bowtie$   
 $könyvkiadó \langle \text{KIADÓ, ORSZÁG, POSTA_CÍM, TELEFON} \rangle)$ ,

ahol

$B = \langle könyvkiadó.KIADÓ, könyvkiadó.POSTA_CÍM \text{ AS CÍME, könyvkiadó.ORSZÁG} \rangle$ ,  
 $T = (kézirat.TÉMA = \text{„krimi”})$ .

### 7. Példa

Tekintsük az előző példa adatbázis modelljét.

#### Feladat

Listázza ki azon szerzőket, akiknek az 1990 és 2000 között kiadott könyveinek átlagára legalább 1000,-Ft. A lista tartalmazza a szerzők nevein kívül a könyveik átlagárát is a fenti időszakban. A lista fejléce legyen  $\langle \text{SZERZŐ, ÁTLAGÁR} \rangle$  alakú.

#### Megoldás

$Avg(\text{ÁR}) [\sigma_{T_1}(könyv \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle) \mid \langle \text{SZERZŐ} \rangle$   
 $] \downarrow_{T_2} \Rightarrow \langle \text{SZERZŐ, Avg AS ÁTLAGÁR} \rangle$ ,

ahol  $T_1 = (könyv.ÉV \in \{1990, \dots, 2000\})$ ,  
 $T_2 = (Avg \geq 1000)$ .

## Egy összetett mintapéllda

A relációalgebra alfejezetének lezárásaként tekintsünk egy összetett mintafeladatot, mely bemutatja az eddig tárgyalt szerkezetmódosító és osztályozó műveleteket.

### Péllda

Legyen egy adatbázis modellje a következő:

$k\acute{e}zirat\langle C\acute{I}M, SZERZ\acute{O}, T\acute{E}MA\rangle$ ,  
 $k\acute{o}nyv\langle C\acute{I}M, SZERZ\acute{O}, \acute{E}V, DARAB, KIAD\acute{O}, \acute{A}R\rangle$ ,  
 $k\acute{o}nyvkiad\acute{o}\langle KIAD\acute{O}, ORSZ\acute{A}G, POSTA\_C\acute{I}M, TELEFON\rangle$ .

Listázza ki az 1 Millió \$ éves bevételt meghaladó krimi-kiadók nevét, az ország nevét, amelyben működnek és éves bevételüket 1990-től 2000-ig éves bontásban! Az elkészült lista fejléce  $\langle KIAD\acute{O}, ORSZ\acute{A}G, \acute{E}V, \acute{E}VES\ BEV\acute{E}TEL\rangle$  legyen, és a listában ne legyenek azonos sorok!

### Megjegyzés

- Egy kiadó akkor krimi-kiadó, ha a vizsgált időszakban legalább egy krimi kiadott.
- Az éves árbevételt egy adott év kiadott könyveinek darabszáma és ára határozza meg.

### Megoldás

A megoldáshoz a feladatot részfeladatokra bontjuk.

#### 1. lépés (A lista reláció specifikálása)

A *lista* reláció specifikálásához két követelményt kell első lépésként figyelembe venni. Egyrészt a fejléc felépítésére vonatkozót, másrészt azt, hogy a *lista* ne tartalmazzon azonos sorokat. Ezek rögzítése már csak azért is célszerű, mivel a továbbiak ezt már nem fogják befolyásolni. Persze azért a megoldás végén célszerű meggondolni, hogy a *lista* előállítása során vajon valóban keletkezhetnek-e azonos sorok, mert ha nem, akkor a redukáló függvény alkalmazására természetesen nincs szükség (mint tudjuk ez jelentősen lassíthatja a *lista* előállítását).

$reduk\acute{a}lt\_lista\langle KIAD\acute{O}, ORSZ\acute{A}G, \acute{E}V, \acute{E}VES\ BEV\acute{E}TEL\rangle =$

$$Reduc(lista\langle KIAD\acute{O}, ORSZ\acute{A}G, \acute{E}V, \acute{E}VES\ BEV\acute{E}TEL\rangle). \quad (1)$$

#### 2. lépés (A lista reláció elemzése és esetleges felbontása)

A *lista* relációval kapcsolatban az első kérdés, hogy vajon mely paraméter-relációkra van szükség a megvalósításához. Az attribútumvektor alapján megállapíthatjuk, hogy

- a KIADÓ attribútumot a *könyv* és a *könyvkiadó* relációkból nyerhetjük,
- az ORSZÁG attribútumot a *könyvkiadó* relációból,
- az ÉV attribútumot a *könyv* relációból, és
- az ÉVES BEVÉTEL származtatott attribútumot pedig szintén a *könyv* relációból.

Ezek után célszerű a *Lista* relációt az alábbi módon felbontani két rész-relációra:

$lista\langle KIAD\acute{O}, ORSZ\acute{A}G, \acute{E}V, \acute{E}VES\ BEV\acute{E}TEL\rangle =$

$$R_1\langle KIAD\acute{O}, \acute{E}V, \acute{E}VES\ BEV\acute{E}TEL\rangle \bowtie_{T_1} R_2\langle KIAD\acute{O}, ORSZ\acute{A}G\rangle, \quad (2)$$

ahol

az  $R_1\langle KIAD\acute{O}, \acute{E}V, \acute{E}VES\ BEV\acute{E}TEL\rangle$  reláció a *könyv* relációból van származtatva,  
 az  $R_2\langle KIAD\acute{O}, ORSZ\acute{A}G\rangle$  reláció pedig a *könyvkiadó* relációból,

a  $T_1$  logikai függvény összekapcsolja az  $R_1$  és az  $R_2$  relációkat a következőképpen:

$$T_1 = (R_1.KIADÓ = R_2.KIADÓ) . \quad (3)$$

### 3. lépés (A rész-relációk elemzése és esetleges felbontása)

#### 3.1. lépés (Az $R_1$ reláció elemzése)

Tulajdonképpen az  $R_1$  reláció valósítja meg a feladat szerinti osztályképzés eredményét, mégpedig oly módon, hogy az általános

$$OP(p)[R(\underline{A}) \mid \underline{B}] \downarrow_T \Rightarrow \underline{C}$$

alakban

$OP = Sum$  az osztályfüggvény (ugyanis összegezni kell),  
 $R(\underline{A}) = R_3(\langle KIADÓ, ÉV, DARAB, ÁR \rangle)$  paraméter-reláció a *könyv* relációból származtatva,  
 $\underline{B} = \langle KIADÓ, ÉV \rangle$  az osztályképző-attribútumok vektora,  
 $p = DARAB * ÁR$  az osztályozó művelet paramétere, amely egy attribútumkifejezés,  
 $\underline{C} = \langle KIADÓ, ÉV, Sum \text{ AS } ÉVES BEVÉTEL \rangle$  az eredmény-reláció attribútumvektora, és  
 $T = (Sum > 1000000)$  az osztályszűrő.

A fentiek alapján tehát az  $R_1$  reláció származtatása:

$$\begin{aligned} R_1(\langle KIADÓ, ÉV, ÉVES BEVÉTEL \rangle) = \\ Sum(DARAB * ÁR) \\ [R_3(\langle KIADÓ, ÉV, DARAB, ÁR \rangle) \mid \langle KIADÓ, ÉV \rangle] \downarrow_{(Sum > 1000000)} \Rightarrow \langle KIADÓ, ÉV, Sum \text{ AS } ÉVES BEVÉTEL \rangle . \end{aligned} \quad (4)$$

#### 3.1.a. lépés (Az $R_3$ reláció elemzése)

Az  $R_3$  relációnak a *könyv* relációból való származtatása egy kiválasztás (melyben a kiválasztó  $T_2$  logikai függvény meghatározása lesz a következő részfeladat), másrészt a  $Sum$  osztályozó művelethez szükséges attribútumok vetítése. Tehát

$$\begin{aligned} R_3(\langle KIADÓ, ÉV, DARAB, ÁR \rangle) = \\ \pi_{KIADÓ, ÉV, DARAB, ÁR}(\sigma_{T_2}(könyv(\langle CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR \rangle))) . \end{aligned} \quad (5)$$

#### 3.1.b. lépés (A $T_2$ logikai függvény elemzése)

A  $T_2$  logikai függvény feladata azon kiadók kiválasztása, melyek bizonyos összetett feltételeknek megfelelnek. Ezek egyrészt vonatkoznak a *kézirat* relációra (a TÉMA attribútum értéke legyen „krimi”), másrészt a *könyv* relációra (az ÉV attribútum értéke legyen 1999 és 2000 közötti). További részfeladat e feltételeknek megfelelő  $R_4$  reláció meghatározása, melynek attribútumhalmaza mind a *kézirat*, mind a *könyv* attribútumait fogja tartalmazni. Így

$$\begin{aligned} T_2 = (könyv.KIADÓ \in \\ \pi_{KIADÓ}(R_4(\langle CÍM, SZERZŐ, TÉMA, ÉV, DARAB, KIADÓ, ÁR \rangle))) . \end{aligned} \quad (6)$$

#### 3.1.c. lépés (Az $R_4$ reláció elemzése)

Az előbb beláttuk, hogy  $R_4$  egy összetett reláció, a *kézirat* relációra és a *könyv* relációra vonatkozó feltételekkel, és természetesen egy olyan kapcsolattal ezek között, melyet a CÍM, SZERZŐ párosok által kijelölt könyvek biztosítanak. (Az nyilvánvaló, hogy egy könyvet

csak e páros tagjainak egyidejű használatával tudunk egyértelműen kijelölni.) Bontsuk fel tehát ennek megfelelően az  $R_4$  relációt:

$$\begin{aligned} R_4\langle \text{CÍM, SZERZŐ, TÉMA, ÉV, DARAB, KIADÓ, ÁR} \rangle = \\ R_5\langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle \bowtie_{T_3} R_6\langle \text{CÍM, SZERZŐ, TÉMA} \rangle, \end{aligned} \quad (7)$$

ahol

az  $R_5\langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle$  relációt a *könyv* relációból származtattuk,  
az  $R_6\langle \text{CÍM, SZERZŐ, TÉMA} \rangle$  relációt pedig a *kézirat* relációból,  
a  $T_3$  logikai függvény összekapcsolja az  $R_5$  és az  $R_6$  relációkat a következőképpen:

$$T_3 = (R_5.\text{CÍM} = R_6.\text{CÍM}) \wedge (R_5.\text{SZERZŐ} = R_6.\text{SZERZŐ}). \quad (8)$$

### 3.1.d.1. lépés (Az $R_5$ reláció elemzése)

Az  $R_5$  relációnak a *könyv* relációból való származtatása szintén egy kiválasztás, melyben a kiválasztó  $T_4$  logikai függvény a kiválasztott könyv megjelenési évére vonatkozó feltétel.

Tehát

$$\begin{aligned} R_5\langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle = \\ \sigma_{T_4}(\text{könyv}\langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle), \end{aligned} \quad (9)$$

$$T_4 = (\text{könyv.ÉV} \in \{1990, \dots, 2000\}). \quad (10)$$

### 3.1.d.2. lépés (Az $R_6$ reláció elemzése)

Az  $R_6$  relációnak a *kézirat* relációból való származtatása az a kiválasztás, melyben a kiválasztó  $T_5$  logikai függvény a kiválasztott könyv témájára vonatkozó feltétel. Tehát

$$\begin{aligned} R_6\langle \text{CÍM, SZERZŐ, TÉMA} \rangle = \\ \sigma_{T_5}(\text{kézirat}\langle \text{CÍM, SZERZŐ, TÉMA} \rangle), \end{aligned} \quad (11)$$

$$T_5 = (\text{kézirat.TÉMA} = \text{„krimi"}). \quad (12)$$

### 3.2. lépés (Az $R_2$ reláció elemzése)

Az  $R_2$  relációra az ORSZÁG attribútum megszerzése érdekében van szükségünk, azonban ahhoz, hogy összeköthető legyen a  $T_1$  logikai függvény segítségével az  $R_1$  relációval, szükség van még a KIADÓ attribútumra is. Ennek megfelelően egy olyan kiválasztást (projekciót) használunk, mely *könyvkiadó* relációból ezt a kettőt tartja meg, azaz

$$\begin{aligned} R_2\langle \text{KIADÓ, ORSZÁG} \rangle = \\ \pi_{\text{KIADÓ, ORSZÁG}}(\text{könyvkiadó}\langle \text{KIADÓ, ORSZÁG, POSTA CÍM, TELEFON} \rangle). \end{aligned} \quad (13)$$

### 4. lépés (A megoldás)

A feladat teljes algebrai megoldása az előzőekben felírt (1), ..., (13) összefüggések együttese. Az alábbiakban ezeket foglaljuk össze. A későbbiekben megmutatjuk a megoldást SQL nyelven is annak érdekében, hogy összehasonlíthassuk a két eszközt.

*redukált\_lista* $\langle \text{KIADÓ, ORSZÁG, ÉV, ÉVES BEVÉTEL} \rangle =$

$$\text{Reduc}(\text{lista}\langle \text{KIADÓ, ORSZÁG, ÉV, ÉVES BEVÉTEL} \rangle) \quad (1)$$

*lista* $\langle \text{KIADÓ, ORSZÁG, ÉV, ÉVES BEVÉTEL} \rangle =$

$$R_1\langle \text{KIADÓ, ÉV, ÉVES BEVÉTEL} \rangle \bowtie_{T_1} R_2\langle \text{KIADÓ, ORSZÁG} \rangle \quad (2)$$

$$T_1 = (R_1.\text{KIADÓ} = R_2.\text{KIADÓ}) \quad (3)$$

$$\begin{aligned}
 R_1 \langle \text{KIADÓ, ÉV, ÉVES BEVÉTEL} \rangle = \\
 \text{Sum}(\text{DARAB} * \text{ÁR}) \\
 [R_3 \langle \text{KIADÓ, ÉV, DARAB, ÁR} \rangle \mid \langle \text{KIADÓ, ÉV} \rangle \\
 ] \downarrow_{(\text{Sum} > 1000000)} \Rightarrow \langle \text{KIADÓ, ÉV, Sum AS ÉVES BEVÉTEL} \rangle
 \end{aligned} \quad (4)$$

$$\begin{aligned}
 R_3 \langle \text{KIADÓ, ÉV, DARAB, ÁR} \rangle = \\
 \pi_{\text{KIADÓ, ÉV, DARAB, ÁR}}(\sigma_{T_2}(\text{könyv} \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle))
 \end{aligned} \quad (5)$$

$$\begin{aligned}
 T_2 = (\text{könyv.KIADÓ} \in \\
 \pi_{\text{KIADÓ}}(R_4 \langle \text{CÍM, SZERZŐ, TÉMA, ÉV, DARAB, KIADÓ, ÁR} \rangle))
 \end{aligned} \quad (6)$$

$$\begin{aligned}
 R_4 \langle \text{CÍM, SZERZŐ, TÉMA, ÉV, DARAB, KIADÓ, ÁR} \rangle = \\
 R_5 \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle \bowtie_{T_3} R_6 \langle \text{CÍM, SZERZŐ, TÉMA} \rangle
 \end{aligned} \quad (7)$$

$$T_3 = (R_5.\text{CÍM} = R_6.\text{CÍM}) \wedge (R_5.\text{SZERZŐ} = R_6.\text{SZERZŐ}) \quad (8)$$

$$\begin{aligned}
 R_5 \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle = \\
 \sigma_{T_4}(\text{könyv} \langle \text{CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR} \rangle)
 \end{aligned} \quad (9)$$

$$T_4 = (\text{könyv.ÉV} \in \{1990, \dots, 2000\}) \quad (10)$$

$$\begin{aligned}
 R_6 \langle \text{CÍM, SZERZŐ, TÉMA} \rangle = \\
 \sigma_{T_5}(\text{kézirat} \langle \text{CÍM, SZERZŐ, TÉMA} \rangle)
 \end{aligned} \quad (11)$$

$$T_5 = (\text{kézirat.TÉMA} = \text{„krimi”}) \quad (12)$$

$$\begin{aligned}
 R_2 \langle \text{KIADÓ, ORSZÁG} \rangle = \\
 \pi_{\text{KIADÓ, ORSZÁG}}(\text{könyvkiadó} \langle \text{KIADÓ, ORSZÁG, POSTA_CÍM, TELEFON} \rangle)
 \end{aligned} \quad (13)$$

## Feladatok

A most következő feladatok célja, hogy az Olvasó kipróbálhassa, mennyire sikerült elsajátítania a relációalgebrai műveletek használatát. Persze vannak köztük összetettebbek is, a nagyobb ambíciók kielégítése érdekében.

### 1.

Legyen egy adatbázis modellje a következő:

- dolgozó*⟨AZONOSÍTÓ, NÉV, MUNKAKÖR, FŐNÖK\_AZONOSÍTÓ, BELÉPÉSI\_DÁTUM, FIZETÉS, PÓTLÉK, OSZTÁLY\_AZONOSÍTÓ⟩ ,  
*osztály*⟨OSZTÁLY\_AZONOSÍTÓ, OSZTÁLY\_NÉV, VÁROS⟩ .
- Listázza ki a vállalat összes dolgozójának és főnökének nevét.
  - Listázza ki a debreceni dolgozók és főnökeik nevét 2000-ben.
  - Listázza ki „Zelei László” osztályvezető osztályán dolgozó könyvelők nevét és azok összjövedelmét 2000-ben. Megjegyezzük, hogy a FIZETÉS és a PÓTLÉK adatok havi értékeket jelentenek.
  - Listázza ki az egyes osztályok dolgozóinak átlagjövedelmét 2000-ben.

## 2.

Legyen egy adatbázis modellje a következő:

*kézirat*(CÍM, SZERZŐ, TÉMA) ,  
*könyv*(CÍM, SZERZŐ, ÉV, DARAB, KIADÓ, ÁR) ,  
*könyvkiadó*(KIADÓ, ORSZÁG, POSTA\_CÍM, TELEFON) .

- a.) Gyűjtse ki a krimi témájú könyvek kiadóinak nevét és postai címét.
- b.) Listázza ki azon krimi-írókat, akiknek az 1990 és 2000 között kiadott könyveinek átlagára legalább 1000,-Ft. A lista tartalmazza a szerzők nevein kívül a könyvek átlagárát is a fenti időszakban. A lista fejléce legyen (SZERZŐ, ÁTLAGÁR) alakú. Megjegyezzük, hogy egy szerzőt "krimi-írónak" nevezünk, ha a vizsgált időszakban legalább egy krimi írt.

## 3.

Legyen egy adatbázis modellje a következő:

*film1*(CÍM, ÉV, HOSSZ, RENDEZŐ, STÚDIO) ,  
*film2*(CÍM, ÉV, SZÍNÉSZ) ,  
*színészek*(SZÍNÉSZ, ÉV, KORA) .

- a.) Listázza ki a 'Dracula' című filmek szereplői közül azokat, akik a film készítésekor 80 évnél idősebbek voltak.
- b.) Listázza ki a Paramount-színészek nevét és azok éves szerepszámát 1980-tól 1990-ig éves bontásban. Az elkészült lista a (SZÍNÉSZ, ÉV, SZEREPEK SZÁMA) fejléccet tartalmazza. Megjegyezzük, hogy egy színészt "Paramount-színésznek" nevezünk, ha a vizsgált időszakban legalább egy Paramount stúdió által készített filmben szerepelt.
- c.) Listázza ki azokat a stúdiókat, amelyek által 1990 és 2000 között készített filmek átlagos hossza legalább 100 perc. A lista tartalmazza a stúdiók nevein kívül a fenti időszakban általuk készített filmek átlagos hosszát is. Az elkészült lista a (STÚDIO, ÁTLAGOS FILM-HOSSZ) fejléccet tartalmazza.
- d.) Listázza ki a Paramount-színészek és rendezők nevét 1980-tól 1990-ig éves bontásban. Az elkészült lista a (RENDEZŐ, ÉV, SZÍNÉSZ) fejléccet tartalmazza. Megjegyezzük, hogy egy színész, vagy egy rendező Paramount-résztvevő, ha a vizsgált időszakban legalább egy Paramount stúdió által készített film készítésében részt vett.

## 4.

Legyen egy adatbázis modellje a következő:

*telephely*(OSZTÁLY, FÖNÖK, CÍM) ,  
*munkakör*(BEOSZTÁS, FIZETÉS, ÉV) ,  
*dolgozó*(NÉV, BEOSZTÁS, OSZTÁLY, KEZDŐ\_ÉV, ZÁRÓ\_ÉV) .

- a.) Adja meg „Tóth Béla” osztályvezető osztályának bérösszegét 2000-ben. Megjegyezzük, hogy KEZDŐ\_ÉV a munkába lépés évét, a ZÁRÓ\_ÉV munkahely elhagyásának évét jelenti, és az aktív dolgozók ZÁRÓ\_ÉV adata: 9999.
- b.) Listázza ki az osztályok és főnökeik nevét, valamint az évszámokat, továbbá az egyes osztályok bérösszegét 1990-től 2000-ig éves bontásban.

## 6. FEJEZET

# A relációs adatmodell szerkezete

A relációs adatmodell szerint felépülő adatbázisok logikai szerkezetének tárgyalásában a legfontosabb fogalom a függőség, mely a relációk attribútumainak kapcsolatait adja meg, sőt a gyakorlatban – mint látni fogjuk – ezeken keresztül jelöljük ki az adatfeltöltést korlátozó relációsémákat. Felépítjük a relációalgebrához sok vonatkozásban hasonló függőség-algebrát, melyben szintén értelmezünk műveleteket, transzformációkat, csak míg a relációalgebrát elsősorban a relációs modellek lekérdezései során használjuk, addig a függőség-algebrának a relációs adatmodell tervezésében van fontos szerepe.

E fejezet megértéséhez az alapvető halmazelméleti és algebrai ismereteken túl az előző fejezetben bevezetett fogalmakra is szükség van. Ezek közül különösen fontos szerepet tölt be tárgyalásunkban a sémafelosztás (relációsémák osztályozása), mivel ez teremt kapcsolatot a relációalgebra és a függőség-algebra között.

Megjegyezzük, hogy ebben a fejezetben gyakorlatilag csak monohalmazokat használunk, így a használt halmazműveletek is a hagyományos halmazalgebra (mono)műveletei.

## A funkcionális-függőség

Az előző fejezetben egy  $R(\underline{A})$  relációsémát egy  $\underline{A}$  attribútumvektor fölötti

$$D(\underline{A}) \triangleq \times \underline{A}$$

Descartes-szorzat egy részhalmazaként vezettünk be, azaz

$$R(\underline{A}) \subseteq D(\underline{A}).$$

De vajon mi jelöli ki ezt a részhalmazt? A halmazmegadáshoz hasonlóan két lehetőségünk van; a relációséma, mint halmaz elemeinek (a rekordoknak) tételes felsorolása, vagy valamilyen tulajdonsággal való kijelölése.

Az  $R(\underline{A})$  relációséma elemeit (elemi vektorait, a rekordokat) nemigen szoktuk felsorolni (tankönyvi mintapéldák kivételével). Egyrészt általában igen nehézkes (fizikailag kivitelezhetetlen) az összes, elvben lehetséges rekord felsorolása, másrészt értelmetlen dolog is, hiszen egy megvalósított  $R(\underline{A})$  relációba ezeknek csak egy töredéke kerül.

A másik módszer szerint olyan feltételeket, megszorításokat adunk meg, melyek „automatikusan” jelölik ki a  $\mathbf{D}(\underline{A})$  megfelelő részhalmazát. Ezeket neveztük függőségeknek.

Mivel tehát a Descartes-szorzatból a relációsémát a függőségek segítségével választjuk ki, így nyilván a függőség az elsődleges fogalom. Ez jelöli ki a Descartes-szorzatból azokat a rekordokat, melyekből a relációséma felépül.

A relációséma fogalmának bevezetésénél egy  $\mathbf{D}(\underline{A})$  Descartes-szorzatot az  $A$  attribútumhalmaz, mint tulajdonsághalmaz egyes  $A_i \in A$  tulajdonság-típusainak konkrét  $a_i \in A_i$  értékei, a *tulajdonság-előfordulások* által együttesen meghatározott objektumok összességéként (más szóval a tulajdonságok  $A$  halmazának absztrakciós szintjén megkülönböztethető objektumok összességéként) értelmeztük. Ezt röviden úgy mondtuk, hogy a  $\mathbf{D}(\underline{A})$  Descartes-szorzat egy objektumtípust reprezentál. Ez az objektumtípus azonban olyan objektumokat is tartalmaz, melyek a gyakorlatban nem léteznek. A relációséma a Descartes-szorzatnak egy olyan szűkítése tehát, mely az attribútumhalmaz értéktartománya által lehetővé tett, elvben előforduló összes objektumhoz képest, csak a gyakorlati megfontolások alapján megadott függőségek által megengedetteket tartalmazza.

A gyakorlatban tehát megfogalmazzuk a függőségeket, és egy rekordot csak akkor „engedünk be” egy relációba, ha teljesülnek rá ezek a függőségek.

Az alábbiakban e függőségek (speciálisan a funkcionális-függőségek) fogalmát fogjuk algebrailag bevezetni és a relációsémákkal kapcsolatos tulajdonságait bemutatni.

## Alapfogalmak

### Hatványfüggőség, függőség, kompatibilitás

Az  $A$  attribútumhalmaz fölötti *hatványfüggőségnek* nevezzük a

$$\mathbf{P}\{A\} \triangleq 2^A \times 2^A$$

halmazt, ahol  $2^A$  az  $A$  hatványhalmaza, vagyis az  $A$  halmaz összes részhalmazainak halmaza. A hatványfüggőség elemeit *függőség-leképezéseknek*, vagy röviden csak *függőségeknek* nevezzük, és valamely  $X, Y \subseteq A$  esetén egy  $\langle X, Y \rangle \in \mathbf{P}\{A\}$  függőséget  $X \rightarrow Y$ , vagy  $f_{X,Y}$  módon jelölünk, és azt mondjuk, hogy az  $X$  attribútumhalmaztól függ az  $Y$  attribútumhalmaz, vagy egyszerűen csak azt, hogy  *$X$ -től függ az  $Y$* . Az  $X$  és az  $Y$  attribútumhalmazok elemeit együttesen az  $f_{X,Y}$  *függőség attribútumainak* nevezzük.

Egy  $A$  attribútumhalmaz és egy  $f_{X,Y}$  függőség kompatibilisek egymással, ha  $X, Y \subseteq A$ . Ezt  $A \# f_{X,Y}$ , vagy  $f_{X,Y} \# A$  módon jelöljük.

#### Megjegyzés

A hatványfüggőség részhalmazai, vagyis az  $F \subseteq \mathbf{P}\{A\}$  függőség-halmazok tehát olyan *unáris halmazleképezések*, melyeknek az  $A$  alaphalmaza egy hiperhalmaz, maguk a függőségek pedig nyilván elemi halmazleképezések. Megjegyezzük, hogy a függőség-halmazokat a későbbiekben *logikai struktúráknak* fogjuk nevezni.

Láthatóan kis mértékben eltértünk a „Leképezések, függvények” alfejezetben bevezetett jelölésektől; az  $f$  függőséget nem húztuk alá (bár nyilván vektor), az  $X \rightarrow Y$  leképezés jelölésében nem talpas nyilat használtunk (bár ez elemi leképezés), és használni fogjuk az  $f = X \rightarrow Y$  jelölést is. Ilyenmódon áttértünk a szakirodalomban elterjedt jelölésekre. Tárnya-

lásunkban sokszor fogunk hivatkozni a függőség leképezés jellegére, és e tulajdonságát ki is fogjuk használni.

Figyeljünk fel arra, hogy a függőség-leképezés mindkét oldalán megjelenhet az üres halmaz, mivel egy halmaz hatványhalmaza az üres halmazt is tartalmazza (ugyanis minden halmaz részhalmaza).

### Függőség magja, képe, egyszerű és összetett függőség

Legyen  $A$  egy attribútumhalmaz,  $\mathbf{P}\{A\}$  a hatványfüggőség az  $A$  fölött, és  $f_{X,Y} \in \mathbf{P}\{A\}$ . Ekkor az  $f_{X,Y}$  függőség *értelmezési tartománya*, vagy *magja*

$$\text{Dom}(f_{X,Y}) \triangleq X,$$

és az  $X$  attribútumait e függőség *magattribútumainak* nevezzük, továbbá az  $f_{X,Y}$  függőség *képtartománya*, vagy *képe*

$$\text{Im}(f_{X,Y}) \triangleq Y,$$

és  $Y$  attribútumait e függőség *képattribútumainak* nevezzük.

Valamely  $f_{X,Y}$  egy *egyszerű-függőség*, ha csak egyetlen képattribútumot tartalmaz (azaz  $\mu(Y) = 1$ ), egyébként pedig *összetett függőségnek* nevezzük.

#### Megjegyzés

Láthatóan a függőség magja és képe fogalmakat már a leképezéseknél az elemi leképezésekre vonatkozó mag és kép fogalmakkal azonos módon definiáltuk.

Nyilván tetszőleges  $f \in \mathbf{P}\{A\}$  esetén  $\text{Dom}(f), \text{Im}(f) \subseteq A$ .

### Zérusfüggőség, egységfüggőség, triviális függőség

A függőség definíciója szerint (az ott tett megjegyzések alapján) a függőségnek mind a magja, mind a képe lehet üres halmaz. Azokat a függőséget, melyeknek a magja, vagy a képe, vagy mindkettő üres halmaz, *zérusfüggőségnek* nevezzük.

Az olyan  $A$  attribútumhalmaz fölötti függőséget, melyben a mag a teljes attribútumhalmaz (azaz  $\text{Dom}(f) = A$ ), *egységfüggőségnek* nevezzük, és ezek halmazát  $FI\{A\}$ -val jelöljük. Nyilván  $\mu(FI\{A\}) = 2^{\mu(A)}$ , ugyanis annyi egységfüggőség van, ahány képhalmaz képezhető az  $A$ -n, vagyis ahány részhalmaza van az  $A$  halmaznak.

A zérusfüggőségeket és az egységfüggőségeket *triviális függőségekként* nevezzük.

### Funkcionális-függőség és relációsémája

Egy  $A$  attribútumhalmaz valamely  $f_{X,Y} \in \mathbf{P}\{A\}$  függőségét *funkcionális-függőségnek* nevezzük egy  $\mathbf{R} \subseteq \mathbf{D}\langle A \rangle$  relációsémára vonatkozóan, ha minden  $\underline{r}, \underline{s} \in \mathbf{R}$  rekordra fennáll, hogy  $\underline{r} \stackrel{X}{\sim} \underline{s}$  esetén  $\underline{r} \stackrel{Y}{\sim} \underline{s}$  is teljesül. Ekkor azt mondjuk, hogy  $X$ -től *funkcionálisan függ*  $Y$  az  $\mathbf{R}$  relációsémában. Ezt az  $f_{X,Y}$  függőség relációsémájának is nevezzük, és  $\mathbf{R}_{X,Y}\langle A \rangle$  módon jelöljük.

Természetesen egy függőségnek több relációsémája van, például az *üres halmaz minden függőségnek relációsémája*. Egy relációsémára is fennállhat több függőség. *A triviális függőségek például minden relációsémára teljesülnek, ám a Descartes-szorozatra csak ezek.*

Bár a szakirodalomban egyéb függőségeket (duális, erős, gyenge, többértékű) is definiálnak, mi *függőség alatt mindig funkcionális-függőséget* értünk. Ennek megfelelően a  $\mathbf{P}\{A\}$

hatványfüggőséget úgy tekintjük, mint az  $A$  attribútumhalmaz fölötti *funkcionális-függőségek halmazát*. Így tetszőleges  $f \in \mathbf{P}\{A\}$  függőség relációsémáját  $\mathbf{R}_f(\underline{A})$  módon is jelöljük.

*Megjegyzés*

$f_{X,Y} \in \mathbf{P}\{A\}$  azt jelenti tehát, hogy amely  $\mathbf{R}_f(\underline{A})$ -beli rekordok hasonlóak az  $X$  attribútumhalmazra (tulajdonságcsoporthoz) vonatkozóan, azok hasonlóak az  $Y$  attribútumhalmazra is.

## Funkcionális-függőség és relációséma kompatibilitása

Legyen  $A$  egy attribútumhalmaz,  $f$  pedig egy vele kompatibilis függőség.

Ekkor egy  $\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$  relációséma és az  $f$  függőség kompatibilis egymással, ha az  $\mathbf{R}(\underline{A})$  relációsémában fennáll az  $f$  függőség. Ezt  $\mathbf{R}(\underline{A}) \# f$ , vagy  $f \# \mathbf{R}(\underline{A})$  módon jelöljük.

## Irreducibilis relációséma

*Irreducibilisnek* nevezzük az olyan nemtriviális (azaz nem üres) relációsémát, mely csak triviális függőségekkel kompatibilis.

Könnyen belátható, hogy *valamely  $A$  attribútumhalmaz feletti  $\mathbf{D}(\underline{A})$  Descartes-szorzat egy irreducibilis relációséma*, tehát bármely  $f \in \mathbf{P}\{A\}$  függőség esetén csak akkor teljesül  $\mathbf{D}(\underline{A}) \# f$ , ha  $f \in FI\{A\}$ , vagyis  $f$  egy egységfüggőség.

### Példa

Legyen  $\underline{A} = \langle A, B \rangle$  egy attribútumvektor, ahol  $A, B = \{0, 1\}$ . Tekintsük az  $\mathbf{R}^{(1)} = \mathbf{D}(\underline{A})$ , és az  $\mathbf{R}^{(2)} = \mathbf{D}(\underline{A}) \setminus \{\langle 1, 1 \rangle\}$  relációsémák tábláit: (Megjegyezzük, hogy a példatáblák a szemléletesség érdekében általában egész számokat fognak tartalmazni.)

$$\mathbf{R}^{(1)}: \begin{array}{cc} \underline{A}: & \underline{B} \\ \hline 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{array}$$

$$\mathbf{R}^{(2)}: \begin{array}{cc} \underline{A} & \underline{B} \\ \hline 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{array}$$

Látható, hogy csak a triviális (tehát a zérus- és egység-) függőségek kompatibilisek az  $\mathbf{R}^{(1)}$  és  $\mathbf{R}^{(2)}$  relációsémákkal, tehát mindkettő irreducibilis.

## Funkcionális-függőség vektorfüggvénye, „függvényjellege”

Legyen  $A$  egy attribútumhalmaz,  $f_{X,Y} \in \mathbf{P}\{A\}$  egy funkcionális függőség, továbbá  $\mathbf{R}_f(\underline{A})$  az  $f_{X,Y}$  relációsémája (ahol  $\underline{A}$  az  $A$  halmaz egy vektora, és nyilván  $X, Y \subseteq A$ ).

Ekkor az  $f_{X,Y}$  *funkcionális függőség vektorfüggvénye* egy

$$\varphi_f: \mathbf{D}(\underline{X}) \rightarrow \mathbf{D}(\underline{Y})$$

vektorfüggvény, ha

1.  $\text{Dom}(\varphi_f) = \mathbf{R}_f(\underline{A}) \parallel_{\underline{X}}$ ,
2.  $\text{Im}(\varphi_f) = \mathbf{R}_f(\underline{A}) \parallel_{\underline{Y}}$ ,
3. minden  $\underline{x} \in \text{Dom}(\varphi_f)$  magelem és  $\underline{y} = \varphi_f(\underline{x})$  képelem esetén van olyan  $\underline{r} \in \mathbf{R}_f(\underline{A})$  rekord, melyre
  - 3.1.  $\underline{r} \upharpoonright_{\underline{X}} = \underline{x}$ , és
  - 3.2.  $\underline{r} \upharpoonright_{\underline{Y}} = \underline{y}$ .

### Állítás

1. Egy függőség vektorfüggvénye valóban függvény.
2. Egy függőség egyértelműen meghatározza vektorfüggvényét.

### Bizonyítás

A 2. állítás nyilvánvalóan következik a függőség vektorfüggvényének konstrukciójából, az 1. állítás pedig a funkcionális függőség definíciójából, de a vektorhasznosság felidézése érdekében felírjuk az 1. állítás formális bizonyítását:

$$\begin{aligned} f_{X,Y} \in \mathbf{P}\{A\} &\models (\forall \underline{r}, \underline{s} \in \mathbf{R}_f(A) : (\underline{r} \stackrel{X}{\sim} \underline{s} \models \underline{r} \stackrel{Y}{\sim} \underline{s})) \models \\ &\models (\forall \underline{r}, \underline{s} \in \mathbf{R}_f(A) : (\underline{r} \vdash_X = \underline{s} \vdash_X) \models (\underline{r} \vdash_Y = \underline{s} \vdash_Y)) \models \\ &\models (\forall \underline{x}_1, \underline{x}_2 \in \text{Dom}(\varphi_f) : (\underline{x}_1 = \underline{x}_2) \models (\varphi_f(\underline{x}_1) = \varphi_f(\underline{x}_2))) \models \varphi_f \text{ függvény.} \end{aligned}$$

### Megjegyzés

A fentiek alapján beszélünk esetenként a funkcionális függőség *függvényjellegéről*.

## Funkcionális-függőség és vektorfüggvényének szemléltetése

A funkcionális-függőség szemléltetéséhez induljunk ki egy  $A = \langle B, C \rangle$  attribútumvektor fölötti  $f = \{B\} \rightarrow \{C\}$  egyszerű funkcionális-függőségből, ahol nyilván  $B, C \in A$ . Legyen továbbá  $B = \{b_1, b_2\}$  és  $C = \{c_1, c_2\}$ , valamint  $\mathbf{R}_f(A)$  az  $f$  függőség egy relációsémája.

Mit jelent vajon az  $f$  függőség az  $\mathbf{R}_f(A)$  relációséma rekordjaira vonatkozóan? A definíció azt mondja, hogy ha valamely  $\underline{r} \in \mathbf{R}_f(A)$  rekordra  $\underline{r}(B) = b_1$  és  $\underline{r}(C) = c_1$ , akkor bármely  $\underline{s} \in \mathbf{R}_f(A)$  rekord esetén, ha  $\underline{s}(B) = b_1$ , akkor  $\underline{s}(C) = c_1$ .

Gondoljunk most az  $\mathbf{R}_f(A)$  relációséma  $\mathbf{R}_f$  jelű táblájára! Az előbbieket nem mást állítanak, mint azt, hogy ha az  $\mathbf{R}_f$  tábla  $B$  és  $C$  nevű oszlopán egyidejűleg végighúzzuk a bal és a jobb kezünk mutatóujját, akkor minden sorban ahol a  $B$  oszlopban  $b_1$  értéket találunk, ott a  $C$  oszlopban  $c_1$  értéket látunk. Tehát a  $B$  oszlop értékeinek ismeretében meg tudjuk „jósolni” a  $C$  oszlop értékeit!

A fentiek azt jelentik, hogy három adatból  $(\underline{r}(B), \underline{r}(C), \underline{s}(B))$  meg tudjuk határozni a negyediket, az  $\underline{s}(C)$  értéket. Ehhez nem kell mást tennünk, mint megkeresni a relációséma táblájának  $B$  oszlopában azt az  $\underline{r}$  sort, melyben az  $\underline{r}(B)$  érték megegyezik az általunk éppen vizsgált  $\underline{s}$  sorbeli  $\underline{s}(B)$  értékkel, és megkeresni az  $\underline{r}$  sorban az  $\underline{r}(C)$  értéket. Ez éppen a keresett  $\underline{s}(C)$  érték lesz. Megjegyezzük, hogy a függőség csak az egyik irányban áll fenn. Tehát a funkcionális-függőség definíciója megengedi, hogy az iménti példa esetén legyen olyan  $\underline{t} \in \mathbf{R}_f(A)$  rekord, melyre ugyan  $\underline{t}(C) = c_1$ , ám  $\underline{t}(B) \neq b_1$ .

A fentiekben bemutatott példában az  $\underline{r}(B) \mapsto \underline{r}(C)$ ,  $\underline{s}(B) \mapsto \underline{s}(C)$ ,  $\underline{t}(B) \mapsto \underline{t}(C)$  elemi leképezésekkel már át is tértünk az  $f$  függőség  $\varphi_f$  vektorfüggvényére, hiszen azok tulajdonképpen a  $\varphi_f$  elemi leképezései. E példából tehát a  $\varphi_f$  függvényjellege is látható.

A függőség és vektorfüggvényének viszonyát azzal is szemléltethetjük, hogy míg az  $f_{X,Y}$  függőség az  $\mathbf{R}_f(A)$  tábla  $X$  halmazbeli „oszlopneveit” (tehát az  $X$ -beli attribútumokat) képezi le az  $Y$  halmazbeli „oszlopnevekre” (tehát az  $Y$ -beli attribútumokra), addig a hozzátartozó  $\varphi_f$  függvény e táblának az  $X$  halmaz által megjelölt „oszlopait” (tehát az  $\mathbf{R}_f(A)$  tábla  $X$ -beli attribútumok által kijelölt altábláját) képezi le e táblának az  $Y$  halmaz által megjelölt „oszlopaira” (tehát az  $\mathbf{R}_f(A)$  tábla  $Y$ -beli attribútumok által kijelölt altáblájára). Ezzel kapcsolatban lásd még a „Függőségi-felosztás tulajdonságai” bekezdés utáni példát!

## A funkcionális függőség és a sémafelosztás kapcsolata

### Függőségi-felosztás (belső-felosztás)

A függőséghez rendelhető vektorfüggvény jól szemlélteti a függőséget, ám mivel a relációsémák táblákként jelennek meg a gyakorlatban, most megvizsgáljuk a hatását e táblákra.

Valamely  $\mathbf{R} = \mathbf{R}(\underline{A})$  relációséma egy  $f \in \mathbf{P}\{A\}$  függőség szerinti felosztásának, más néven függőségi vagy belső-felosztásának nevezzük

- az  $\mathbf{R}$  relációséma  $T_{\mathbf{R}}^{Dom(f)}$  osztálykritérium szerinti

$$\Phi^D[\mathbf{R}|f] \triangleq \Phi[\mathbf{R}|T_{\mathbf{R}}^{Dom(f)}] \text{ sémafelosztását, vagy másképpen}$$

- az  $\mathbf{R}$  relációséma  $Dom(f)$  attribútumhalmaz szerinti

$$\Phi^D[\mathbf{R}|f] \triangleq \Phi[\mathbf{R}|Dom(f)] \text{ sémafelosztását.}$$

Ekkor az  $f$ -et a *függőségi-felosztás függőségének*, a felosztás elemeit alkotó halmazokat pedig *sémaosztályoknak* nevezzük.

#### Megjegyzés

A függőségi-felosztás definícióját tehát visszavezettük – az előző fejezetben bemutatott – osztálykritérium szerinti, illetve attribútumhalmaz szerinti sémafelosztásra. E két definíció ekvivalens, csupán az attribútum szerinti sémafelosztás definíciójára kell gondolnunk, melyben a jelen esetre alkalmazott osztálykritérium:

$$T_{\mathbf{R}}^{Dom(f)} = ( \text{minden } \mathbf{R}_i \in \Phi^D[\mathbf{R}|f], \underline{r}, \underline{t} \in \mathbf{R}_i, \text{ és } A_k \in Dom(f) \text{ esetén } \underline{r}(A_k) = \underline{t}(A_k) ),$$

ahol a jelölésben szereplő „ $D$ ” betű az angol „dependence” (függőség) szóra utal.

#### Állítás

Ha egy  $\mathbf{R}(\underline{A})$  relációséma függőségi-felosztását valamely  $fI$  egységfüggőség szerint végezzük (tehát  $Dom(fI) = A$ ), akkor elemi-sémafelosztást kapunk, azaz

$$\Phi^D[\mathbf{R}|fI] = \Phi^e[\mathbf{R}], \text{ ahol } \Phi^e[\mathbf{R}] = \{ \{ \underline{r} \} \mid \underline{r} \in \mathbf{R} \}.$$

#### Bizonyítás

Az attribútum szerinti sémafelosztás (előző fejezetben szereplő) definíciója szerint ugyanis egy osztályban azok a rekordok vannak, melyek az osztályképző-attribútumokon páronként azonos értékeket vesznek fel. Mivel azonban az egységfüggőség magja, vagyis az osztályképző-attribútumainak halmaza a teljes attribútumhalmaz, így a páronkénti egyezés esetén a teljes rekordnak egyeznie kellene. Ám a relációséma definíció szerint monohalmaz, tehát a sémaosztályokban csak egy elem lehet, vagyis a sémafelosztás elemi. ■

### A függőségi-felosztás szemléltetése

A fentiek szemléltetéseként tekintsük az  $\mathbf{R}(\underline{A})$  relációsémának a fenti  $f = \{B\} \rightarrow \{C\}$  függőség szerinti felosztását. Jelen esetben ez az  $\mathbf{R}(\underline{A})$  relációséma  $B$  attribútum szerinti felosztását jelenti. Ekkor tehát az osztályképző-attribútum a  $B$ , és nyilván olyan (rekordelemű) sémaosztályokat kapunk, ahol az egy osztályon belüli rekordok  $B$  attribútum szerinti komponensei megegyeznek (hiszen a  $B$  egy osztálytag-attribútum is egyúttal). Ekkor a  $\{B\} \rightarrow$

→ {C} függőség azt jelenti, hogy egy osztályon belül a C attribútum értékei is megegyeznek (vagyis a C is osztálytag-attribútum). Mivel a függőség vektorfüggvénye nem bijektív, így persze több osztályban is lehet a C attribútumnak azonos értéke.

Egy általános  $f_{X,Y} \in \mathbf{P}\{A\}$  függőség az  $\mathbf{R} = \mathbf{R}\langle \underline{A} \rangle$  relációsémát az  $X$  attribútumhalmaz minden attribútuma szerint osztályokra bontja az  $\Phi^D[\mathbf{R}|f]$  függőségi-felosztás szerint. Az egy osztályon belüli rekordoknak ekkor az összes  $X$  attribútumhalmazbeli komponense páronként megegyezik, vagyis az egy osztályon belüli bármely  $\underline{r}$  és  $\underline{s}$  rekordokra minden  $B_i \in X$  esetén  $\underline{r}(B_i) = \underline{s}(B_i)$ . Az  $f_{X,Y}$  függőség ekkor azt jelenti, hogy az egy osztályon belüli rekordoknak az összes  $Y$  attribútumhalmazbeli komponensei is megegyeznek páronként (az előbbi értelemben).

## A függőségi-felosztás tulajdonságai

A fentiek alapján foglaljuk össze a függőségi-felosztás legfontosabb tulajdonságait:

- I. Valamely függőségi-felosztás során egy osztályba kerülnek azok a rekordok, melyek a függőség vektorfüggvényének ugyanazt az elemi leképezését reprezentálják. (Ezt fejezi ki a függőségi-felosztás osztálykritériuma.)
- II. A függőségi-felosztás egyértelmű, és teljesülnek rá a felosztási tulajdonságok (lásd az előző fejezet „Halmazfelosztás osztálykritérium szerint” pontját).
- III. Egy relációsémának egy vele kompatibilis függőség szerinti felosztásában minden sémaosztály minden részhalmaza, mint relációséma szintén kompatibilis lesz a függőséggel, és ez a tulajdonság megmarad akkor is, ha valamelyik sémaosztályt kibővítjük egy másiknak bármelyik rekordjával.
- IV. Ha egy függőség nem kompatibilis azzal a relációsémával, amelyre a függőségi-felosztást végezzük, akkor minden sémaosztály egyelemű lesz (vagyis ekkor a függőségi-felosztás egy elemi-felosztás). Ebből következően minden irreducibilis relációséma bármely függőség szerinti felosztása elemi. (Ez legfontosabb tulajdonságuk.)
- V. Minden relációsémának az egységfüggőség szerinti függőségi-felosztása elemi.

Megállapíthatjuk tehát, hogy a függőségi-felosztáson keresztül a sémafelosztások és a függőségek igen szoros kapcsolatba kerültek. Egyrészt *egy függőség egyértelműen meghatározza egy relációséma függőségi-felosztását*. Másrészt ez a kapcsolat fordítva is fennáll, vagyis, ha létrehozunk egy felosztást egy relációsémán, akkor ahhoz hozzárendelhető olyan függőség, melynek *magattribútumai* (vagyis a baloldali attribútumai) a relációséma osztályképző-attribútumai, a *képattribútumai* (vagyis a függőség jobboldali attribútumai) pedig a relációséma nem osztályképző osztálytag-attribútumai. Ilyen függőség adott relációséma adott felosztása mellett több is lehet, de legalább az üres vektor, mint függőség biztos, hogy hozzárendelhető minden relációséma minden felosztásához.

Végül a függőségi-felosztás fenti I-V. tulajdonságait algebrai módon is összefoglaljuk. Tekintsünk valamely,  $A$  attribútumhalmaz fölötti  $\mathbf{R} = \mathbf{R}\langle \underline{A} \rangle$  relációsémát, egy  $f \in \mathbf{P}\{A\}$  függőséget, valamint egy  $\Phi^D[\mathbf{R}|f]$  függőségi-felosztást az ehhez tartozó  $T = T_{\mathbf{R}}^{\text{Dom}(f)}$  osztálykritériummal. Ekkor teljesülnek az alábbiak:

1. A függőségi-felosztás egy sémaosztálybeli rekordjai a függőség attribútumaiban páronként azonos értéket vesznek fel, azaz

$$\text{minden } \mathbf{R}_i \in \Phi^D[\mathbf{R}|f], \underline{r}, \underline{s} \in \mathbf{R}_i, \text{ és } A_m \in \text{Dom}(f) \cup \text{Im}(f) \text{ esetén } \underline{r}(A_m) = \underline{s}(A_m).$$

Ez a magattribútumokra nézve abból adódik, hogy a felosztást a függőség magja szerint végeztük, a képattribútumokra pedig a függőség függvényjellegéből.

2. A függőségi-felosztás egyértelmű, és teljesülnek rá a felosztási tulajdonságok:
  - 2.1. minden  $R_i \in \Phi^D[R|f]$  esetén  $R_i \subseteq R$ ,
  - 2.2.  $\bigcup_{R_i \in \Phi^D[R|f]} R_i = R$ ,
  - 2.3. minden  $R_i, R_j \in \Phi^D[R|f]$  és  $R_i \neq R_j$  esetén  $R_i \cap R_j = \emptyset$ ,
  - 2.4. minden elemi halmazán (sémaosztályán) teljesül a  $T$  osztálykritérium, azaz minden  $R_i \in \Phi^D[R|f]$  esetén  $T(R_i) = \text{IGAZ}$ , és
  - 2.5. minden elemi halmazát kibővítve egy másik elemi halmazból származó elemmel, az így kapott halmazra már nem teljesül a  $T$  osztálykritérium, azaz minden  $R_i, R_j \in \Phi^D[R|f]$ ,  $R_i \neq R_j$  és  $r \in R_i$  esetén  $T(R_j \cup \{r\}) = \text{HAMIS}$ .

(Ez az osztálykritérium előző fejezetben megadott definícióját követő megjegyzések utolsó bekezdéséből következik, mely szerint az osztálykritériumot a gyakorlatban mindig „pozitív módon”, az „egy osztályban való tartózkodásra” fogalmazzuk meg. Nem mondjuk ki, hanem természetesnek tekintjük, hogy különböző osztályok elemei között nem teljesül.)
3. Ha egy relációséma kompatibilis egy függőséggel, akkor az ezzel végzett függőségi-felosztás osztályai, valamint azok részhalmazai, mint relációsémák szintén kompatibilisek lesznek e függőséggel, továbbá ez a tulajdonság fennmarad akkor is, ha valamelyik osztály egy rekordjával kibővítünk egy másik osztályt, azaz
  - 3.1. ha  $R \# f$ , akkor minden  $R_i \in \Phi^D[R|f]$  és  $R_{i1} \subseteq R_i$  esetén  $R_{i1} \# f$ ,
  - 3.2. ha  $R \# f$ , akkor minden  $R_i, R_j \in \Phi^D[R|f]$ ,  $R_i \neq R_j$ ,  $r \in R_i$  és  $R_{j1} \subseteq R_j \cup \{r\}$  esetén  $R_{j1} \# f$ .
4. Ha egy relációséma nem kompatibilis egy függőséggel, akkor az ezzel végzett függőségi-felosztás elemi lesz, azaz
 

ha  $\neg(R \# f)$ , akkor  $\Phi^D[R|f] = \Phi^e[R]$ .

Ebből már következik, hogy minden irreducibilis relációsémának csak elemi függőségi-felosztása létezik, hiszen az irreducibilis relációsémák egyetlen nemtriviális függőséggel sem kompatibilisek.
5. Minden relációsémának az egységfüggőség szerinti függőségi-felosztása elemi, azaz tetszőleges  $fI \in FI\{A\}$  esetén
 
$$\Phi^D[R|fI] = \Phi^e[R].$$

A függőségi-felosztás tulajdonságainak tárgyalását zárjuk le néhány egyszerű példa bemutatásával, és értelmezésével.

#### Példa

Legyen egy attribútumvektor  $\underline{A} = \langle A, B, C, D, E \rangle$ , ahol  $A = \{6, 7\}$ ,  $B = \{3, 4\}$ ,  $C = \{1, 2\}$ ,  $D = \{3, 4\}$ ,  $E = \{8, 9\}$ , és tekintsük az  $f = \{A, B\} \rightarrow \{C, E\}$  függőséget, valamint az alábbi táblával adott  $R = R\langle A, B, C, D, E \rangle$  relációsémát:

R:	A	B	C	D	E
	6	3	1	4	9
	6	4	2	4	8
	7	3	2	3	8
	6	3	1	3	9
	6	4	2	3	8

*Feladat*

Állítsuk elő az **R** relációséma  $f$  függőség szerinti függőségi-felosztását, vagyis a  $\Phi^D[\mathbf{R}|f]$  felosztást!

*Megoldás*

Először is vegyük észre, hogy az  $f$  függőség és az **R** relációséma kompatibilisek, és így a feladatnak nem csupán az elemi függőségi-felosztás felel meg.

Az **R** relációséma  $f$  függőség szerinti felosztása  $\Phi^D[\mathbf{R}|f] = \{\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \mathbf{R}^{(3)}\}$ , ahol a szereplő osztályok, mint relációsémák táblái:

$\mathbf{R}^{(1)}:$	A	B	C	D	E	$\mathbf{R}^{(2)}:$	A	B	C	D	E	$\mathbf{R}^{(3)}:$	A	B	C	D	E
	6	3	1	4	9		6	4	2	4	8		7	3	2	3	8
	6	3	1	3	9		6	4	2	3	8						

Könnyen belátható, hogy a függőségi-felosztásra a fentiekben bemutatott 1-5. tulajdonság az  $\mathbf{R}^{(1)}$ ,  $\mathbf{R}^{(2)}$  és  $\mathbf{R}^{(3)}$  relációsémákban fennállnak. Ezek közül vizsgáljuk meg az 1. tulajdonság teljesülését, ami ahhoz a megállapításunkhoz kapcsolódik, mely szerint „valamely függőség szerinti felosztás során egy osztályba kerülnek azok a rekordok, melyek a függőség vektorfüggvényének ugyanazt az elemi leképezését reprezentálják” (lásd „A függőségi-felosztás tulajdonságai” bekezdés I. pontját). Hogyan is teljesül ez esetünkben?

Az  $f = \{A, B\} \rightarrow \{C, E\}$  függőség  $\varphi_f: A \times B \rightarrow C \times E$  kétváltozós vektorfüggvényének például a  $\langle 6, 3 \rangle \mapsto \langle 1, 9 \rangle$ , vagy más jelölésben a  $\varphi_f(\langle 6, 3 \rangle) = \langle 1, 9 \rangle$  elemi leképezését az  $\mathbf{R}^{(1)}$  sémaosztály reprezentálja.

Figyeljünk fel végül arra, hogy az  $f$  függőség valóban csak az egyik irányban jelent kényszert, máskülönben a  $\langle 7, 3, 2, 3, 8 \rangle$  rekord miatt nem volna kompatibilis az  $f$  függőség és az **R** relációséma.

**Példa**

E példa megmutatja miként hat ugyanarra a relációsémára két különböző függőség szerinti felosztás. Legyen egy  $\underline{A} = \langle A, B, C, D \rangle$ , ahol  $A = \{6, 8\}$ ,  $B = \{3, 5\}$ ,  $C = \{1\}$ ,  $D = \{3, 4\}$ , és tekintsük az  $f_1 = \{A\} \rightarrow \{D\}$ , valamint az  $f_2 = \{B\} \rightarrow \{C\}$  függőségeket, valamint az alábbi táblával adott  $\mathbf{R} = \mathbf{R}\langle A, B, C, D \rangle$  relációsémát:

R:	A	B	C	D
	6	3	1	4
	6	5	1	4
	8	3	1	3

*Feladat*

Végezzük el az **R** relációséma  $f_1$ , és  $f_2$  függőségek szerinti felosztását. Az eredmény az alábbi táblákban látható.

*Megoldás*

$\Phi^D[\mathbf{R} f_1]:$	A	B	C	D	A	B	C	D
	6	3	1	4	8	3	1	3
	6	5	1	4				

$\Phi^D[\mathbf{R} f_2] :$	<table><tr><th>A</th><th>B</th><th>C</th><th>D</th></tr><tr><td>6</td><td>3</td><td>1</td><td>4</td></tr><tr><td>8</td><td>3</td><td>1</td><td>3</td></tr></table>	A	B	C	D	6	3	1	4	8	3	1	3	<table><tr><th>A</th><th>B</th><th>C</th><th>D</th></tr><tr><td>6</td><td>5</td><td>1</td><td>4</td></tr></table>	A	B	C	D	6	5	1	4	Láthatóan $\Phi^D[\mathbf{R} f_1] \neq \Phi^D[\mathbf{R} f_2].$
A	B	C	D																				
6	3	1	4																				
8	3	1	3																				
A	B	C	D																				
6	5	1	4																				

## A funkcionális-függőség gyakorlati meghatározása

A gyakorlatban (és különösen az oktatásban, például egy zárthelyi dolgozat során) a függőségek megfogalmazását vagy egy attribútumhalmaz megadása alapján várják el (ekkor a FŐKÓD, ALAPFIZ, BEDÁTUM és hasonló „beszédes” attribútumnevek alapján keletkező asszociációk szerint kell a függőségeket definiálni), vagy valamely konkrét reláció, pontosabban egy ezt implementáló valahány soros kitöltött tábla adatai alapján (ekkor az előbbihez hasonló „beszédes” neveket a gondolkodást „segítő” adatok egészítik ki). Tény azonban, hogy ezek csak nagyon hozzávetőleges támogatást adnak a függőségek megfogalmazásához, és ami a legfontosabb, semmi közülük sincs a függőség definíciójához.

Az első módszer (az attribútumhalmaz alapján történő függőség-megfogalmazás) komoly értelmezési problémákat vethet fel (ezeket minden, ilyen típusú zárthelyi dolgozatot írató tanár jól ismeri). A matematikai definíciót ugyanis ekkor bármely függőség, illetve függőség-halmaz megadása kielégíti, hiszen a definíció nem állít mást, mint azt, hogy tetszőleges  $X \rightarrow Y$  páros funkcionális-függőséget alkot, ha  $X, Y \subseteq A$  (ahol  $A$  a megadott attribútumhalmaz), és létezik olyan  $\mathbf{R}\langle A \rangle \subseteq \mathbf{D}\langle A \rangle$  relációséma, melynek minden  $r, s \in \mathbf{R}\langle A \rangle$  rekordjára fennáll, hogy  $r \stackrel{X}{\sim} s$  esetén  $r \stackrel{Y}{\sim} s$  is teljesül. Ilyen pedig mindig van, például az üres halmaz, mint relációséma.

Nyilvánvaló tehát, hogy a függőség algebrai definíciója nem sokat segít a függőség gyakorlati megfogalmazásában. Ez azonban érthető, ha arra gondolunk, hogy éppen a függőségektől várjuk el a relációsémának a gyakorlati szempontok alapján történő megszorítását a Descartes-szorzatból. Amikor egy rendszerelemző nekikezd egy adatbázis megtervezésének, akkor a környezeti tanulmányokból, a riportokból egyrészt ki kell jelölnie az attribútumokat (vagyis azt, hogy milyen adatokat akar tárolni, és az egyes adatoknak mi az engedélyezett értéktartománya, azaz a típusa), másrészt az attribútumok értelmezése alapján megállapítania a függőségeket (vagyis azt, hogy mi a kapcsolat az egyes adatok között). Ezzel meg is mondtuk, hogy a gyakorlatban mit értünk függőség alatt (az algebrai definíció – nem véletlenül – ezzel összhangban van), és hogy általában miként határozzuk meg azokat. Ennek megfelelően a függőségeket a legtöbbször az attribútumokra vonatkozó valamilyen szöveges leírással, egy logikai kényszerkapcsolattal adjuk meg (például egy országon belül az irányítószám meghatározza a település nevét, tehát fennáll az IRÁNYÍTÓSZÁM  $\rightarrow$  TELEPÜLÉS funkcionális-függőség).

A másodikként említett módszer (a valahány soros kitöltött relációtábla alapján történő függőség-megfogalmazás) két okból is elvileg helytelen.

Az egyik ok az, hogy egy konkrét reláció adattáblájából csak az olvasható ki egyértelműen, hogy mely függőségek nem teljesülnek. Pusztán csak azért, mert egy  $R_A$  táblából kiolvasható, hogy valamely  $A_1, A_2 \in A$  attribútumokra minden  $r, s \in R_A$  rekord esetén az  $r(A_1) = s(A_1)$  egyenlőség teljesülésekor az  $r(A_2) = s(A_2)$  is fennáll, még egyáltalán nem biztos, hogy az  $R_A$  tábla egy  $t$  rekorddal való kibővítése esetén is igaz lesz ez (vagyis, hogy az  $r(A_1) = t(A_1)$  egyenlőség esetén is teljesül majd az  $r(A_2) = t(A_2)$  egyenlőség).

Az előbb azt mondtuk, hogy egy konkrét táblából nem olvasható ki egy függőség (illetve ami kiolvasható, az nem biztos, hogy függőség). Ez fordítva is igaz, azaz *egy relációsémára fennálló függőség nem biztos, hogy „látszik” egy konkrét táblában*. Ez tehát a másik oka annak, hogy miért helytelen általában a függőséget egy konkrét relációtábla alapján megfogalmazni. Például az előbb említett  $\{\text{IRÁNYÍTÓSZÁM}\} \rightarrow \{\text{TELEPÜLÉS}\}$  függőség egy olyan táblában, amelyben minden TELEPÜLÉS érték csak egyszer szerepel, természetesen nem látható, ám ettől még nyilvánvalóan fennáll.

Ha a relációtábla alapos rendszerelemzési munka eredménye, a benne szereplő adatok jól reprezentálják a modellezendő világot, akkor természetesen a belőle kiolvasható függőségek is jól használhatók lesznek az adatbázis tervezésében.

A megfogalmazott függőségek a gyakorlatban az alapos tervezési munka ellenére sem mindig teljesítik az „összes rekordra” vonatkozó relatív azonosságot. Az  $\mathbf{R}(\text{NÉV, SZÜLETÉSI\_DÁTUM, ANYJA\_NEVE, BEOSZTÁS, ...})$  relációsémában a  $\{\text{NÉV, SZÜLETÉSI\_DÁTUM, ANYJA\_NEVE}\} \rightarrow \{\text{BEOSZTÁS}\}$  a gyakorlatban funkcionális-függőségként jól használható. Nem lehet azonban megtiltani két Tóthnének, hogy ha már egyszer ugyanazon a napon szülnék fiúgyermeket, akkor azokat például Bélának nevezzék. Ők később pedig mindketten kerülhetnek ugyanarra a munkahelyre... Az ilyen problémákat a bérelszámolók úgy oldják meg, hogy „csinálnak” egy Tóth1 Béla és egy Tóth2 Béla dolgozót, aminek az érintettek nem biztos, hogy örülnek (az adatbázis-adminisztrátorok pedig biztos nem).

Az „összes rekordra” vonatkozó relatív azonosság követelménye esetenként kritikus lehet (például személyi szám generálása esetén) más esetben valamilyen adatmódosítással (lásd az előző példában) teljesíthető, azonban a gyakorlatban a „józan megfontolás” általában elég jó megoldásokat sugall. A fenti definícióban szereplő „...minden  $r, t \in \mathbf{R}(A)$  rekordra...” feltétel tehát a gyakorlatban nem mindig teljesül. (A „minden” szó helyett inkább valami olyat kellene mondani, hogy „elég sok”, vagy „minden olyan, ami a napi gyakorlatban előfordul”, stb.) Mindenesetre az adatmodell-tervezésben a függőségek definiálása egy tervezői döntés, melynek háttérében megfelelő szakmai tapasztalatnak és kompromisszumkészségnek kell állnia.

A fentiek alapján tehát azt mondhatjuk, hogy a gyakorlatban használt „függőségek” nem mindig függőségek az algebrai definíció értelmében.

Végül még néhány szót szólunk függőségek gyakorlati alkalmazásáról. A függőségek figyelembevétele az adatmodell kialakításában többek között azért hasznos, mivel így növelni lehet a tárolási hatékonyságot (másképpen elkerülhetők az úgynevezett adatbázis-anomáliák, lásd később!). Például egy  $X \rightarrow Y$  függőség fennállása esetén megtehetjük, hogy az  $X$ -ben egyező rekordokhoz tartozó  $Y$ -beli alrekordokat csak egyszer tároljuk. A függőségek felismerése tehát lehetőséget ad az adatok közötti egyfajta *redundancia* kiszűrésére. Ám a redundancia elkerülésére való túlzott törekvés merevvé teheti a nyilvántartó rendszert. Hátrányos következményei lehetnek például a  $\{\text{NÉV}\} \rightarrow \{\text{CÍM}\}$  függőség adatbázisba való „beépítésének”, mivel megakadályozhatja olyan személy bevitelét, akinek két lakcíme van.

## A logikai-struktúra

Definíció szerint egy gyakorlati  $R(\underline{A})$  reláció rekordjai az attribútumok  $A$  halmaza fölötti  $R(\underline{A})$  relációséma rekordjaiból kerülnek ki (csak esetleg némelyik rekord többszörözve, hiszen a relációt – a gyakorlati igényekkel összhangban – multihalmazként definiáltuk). Feltettük a kérdést, vajon mi korlátozza a  $D(\underline{A})$  Descartes-szorzat rekordjainak az  $R(\underline{A})$  relációsémába való kerülését, és meg is adtuk a választ; a gyakorlati igények alapján megfogalmazott függőségek. A függőség algebrai definíciója alapján már meghatározható a relációsémák azon halmaza, melyek adott függőségeket (függőség-halmazt) kielégítenek.

Tehát az adatbázis-tervezés első lépése, az attribútumhalmaz kijelölése után (megadva az egyes attribútumok értéktartományait), a második lépés a függőségek kijelölése (gyakorlati megfontolások alapján), és a relációsémát/relációsémákat (vagyis az adatbázisban tárolható rekordok halmazát) már a kijelölt függőségek határozzák meg az engedélyezett tartományon (az attribútumok Descartes-szorzatán) belül. Ilyen módon azt mondhatjuk, hogy az adatbázis-tervezésben a függőség az elsődleges fogalom, és a reláció (pontosabban a relációséma) a másodlagos, a származtatott fogalom.

Megjegyezzük, hogy ez az elsődlegesség természetesen az adatbázis tervezésére és az adatokkal való feltöltésére vonatkozik. (Ez utóbbi esetben a függőségek – mint „megszorítások” – korlátozzák az adatbázisba bevihető adatok körét.) Az adatbázis-lekérdezések során a függőségeknek már csak közvetve – az egyes relációtáblák kapcsolódásain keresztül – van jelentőségük. Ekkor a felhasználó csak a relációkat „látja”, így akkor ez az elsődleges.

## Alapfogalmak

### Logikai-struktúra, kompatibilitás, részstruktúra, triviális struktúra

Az adatbázis tervezésében, feltöltésében kiemelkedő – fentiekben vázolt – jelentősége miatt a tervezés során kijelölt függőségek halmazát az adatbázis *logikai-struktúrájának* nevezzük. Ez azonban (ellentétben az attribútumok halmazával és értéktartományaival) a tervezés során változik. Tulajdonképpen az adatbázis-tervezés módszertanilag nem más, mint a logikai-struktúra különböző szempontok szerint legmegfelelőbb alakra való transzformálása. Ennek megfelelően *logikai-struktúrának* nevezzük az *adatbázis-tervezés különböző lépéseiben az aktuális függőség-halmazt*.

A logikai-struktúra szoros kapcsolatban van az attribútumhalmazzal, hiszen annak elemei (az egyes attribútumok) közötti „kényszereket” fejeznek ki a logikai-struktúra elemeit alkotó függőségek. Nyilván nem szükséges, hogy egy adatbázis attribútumhalmazának minden eleme szerepeljen valamelyik függőségben, de az természetesen kötelező, hogy a függőségek attribútumai csak azok közül kerüljenek ki. Ez a *kompatibilitási követelmény*.

A fentieket alapján tehát valamely  $A$  attribútumhalmaz fölötti *logikai-struktúrának* nevezzük függőségek tetszőleges  $F \subseteq \mathbf{P}\{A\}$  halmazát, vagyis egy  $A$  *alaphalmazú unáris halmaz-függvényt*. A benne szereplő függőségek attribútumait összefoglalóan a *logikai-struktúra attribútumainak* is nevezzük. Az algebrai definícióból láthatóan egy logikai-struktúra lehet üres halmaz is. Egy  $F$  logikai-struktúra tetszőleges részhalmazát az  $F$  *részstruktúrájának* nevezzük.

*Triviális logikai-struktúrának* nevezzük az üres halmazt, mint logikai-struktúrát, valamint az olyan logikai-struktúrát is, mely csak egyetlen függőséget tartalmaz, és az triviális.

Kiterjesztjük az attribútumhalmaz függőséggel való kompatibilitásának fogalmát a logikai-struktúrára, és azt mondjuk, hogy egy  $F$  logikai-struktúra és egy  $A$  attribútumhalmaz kompatibilisek egymással, ha az  $F$  minden függősége kompatibilis az  $A$ -val, azaz ha minden  $f \in F$  függőség esetén  $\text{Dom}(f), \text{Im}(f) \subseteq A$ , vagyis az  $F$  minden attribútuma szerepel az  $A$ -ban. Tehát egy  $F$  logikai-struktúra kompatibilis egy  $A$  attribútumhalmazzal, azaz

$$F \# A, \text{ ha } F \subseteq \mathbf{P}\{A\},$$

és ekkor használjuk az  $A \# F$ , valamint az  $F$  logikai-struktúrára az  $F\{A\}$  jelölést is.

## A logikai-struktúra és a relációséma kapcsolata

### Logikai-struktúra relációsémája

Legyen  $A$  egy attribútumhalmaz,  $F = F\{A\}$  egy vele kompatibilis logikai-struktúra (azaz  $F \subseteq \mathbf{P}\{A\}$ ), és  $\mathbf{R}_F \subseteq \mathbf{D}\langle A \rangle$  egy relációséma.

Ekkor az  $F$  logikai-struktúra relációsémája  $\mathbf{R}_F$ , ha minden  $f \in F$  függőségnek van olyan  $\mathbf{R}_f$  relációsémája, melyre  $\mathbf{R}_f = \mathbf{R}_F$ .

Másképpen ezt úgy mondhatjuk, hogy az  $F$  logikai-struktúra relációsémája  $\mathbf{R}_F$ , ha az összes  $r, s \in \mathbf{R}_F$  rekordon minden  $X \rightarrow Y \in F$  függőségre fennáll, hogy  $r \stackrel{X}{\sim} s$  esetén  $r \stackrel{Y}{\sim} s$  is teljesül. Természetesen egy logikai-struktúrához is több relációséma tartozik. Az alábbi állítások e definíció nyilvánvaló következményeit mondják ki.

#### Állítás

1. Az üres halmazra, mint relációsémára és az egyetlen rekordból álló relációsémára minden függőség teljesül.
2. Az üres halmaznak, mint logikai-struktúrának a függőségsémája a Descartes-szorzat.

### Relációséma struktúrája, kompatibilitás

A fenti definíciókban függőség és függőség-halmaz (logikai-struktúra) relációsémájáról beszéltünk. Ez a hozzárendelés azonban fordított irányban is értelmezhető.

Legyen  $A$  egy attribútumhalmaz,  $F = F\{A\}$  egy vele kompatibilis logikai-struktúra (azaz  $F \subseteq \mathbf{P}\{A\}$ ),  $\mathbf{D}\langle A \rangle$  az  $A$  fölötti Descartes-szorzat és  $\mathbf{R} \subseteq \mathbf{D}\langle A \rangle$  egy relációséma.

Ekkor az  $\mathbf{R}$  relációséma  $F$  struktúrájú, ha  $\mathbf{R}$  az  $F$ -nek egy relációsémája, tehát az  $\mathbf{R}$  relációsémában fennáll az  $F$  logikai-struktúra összes függősége (vagyis az  $\mathbf{R}\langle A \rangle$  mindegyikkel kompatibilis).

Ekkor az  $\mathbf{R}$  relációséma és az  $F$  logikai-struktúra kompatibilisek egymással, amit  $\mathbf{R} \# F$ , vagy  $F \# \mathbf{R}$  módon jelölünk.

Értelmezzük az  $A$  attribútumhalmaz és az  $\mathbf{R}$  relációséma kompatibilitását, ha  $\mathbf{R} \subseteq \mathbf{D}\langle A \rangle$ , és ezt  $\mathbf{R} \# A$ , vagy  $A \# \mathbf{R}$  módon, illetve  $\mathbf{R}\langle A \rangle$  jelöljük.

A következő állítások a fenti definíció nyilvánvaló következményeit mondják ki.

**Állítás**

1. Minden relációsémához létezik vele kompatibilis logikai-struktúra, például az üres halmaz, mint logikai-struktúra, minden relációsémával kompatibilis.
2. A Descartes-szorozathoz, mint relációsémához az egyetlen kompatibilis logikai-struktúra az üres halmaz.

**Példa**

Legyen egy attribútumvektor  $\underline{A} = \langle A, B, C, D, E \rangle$ , ahol  $A = \{6, 7\}$ ,  $B = \{3, 4\}$ ,  $C = \{1, 2\}$ ,  $D = \{3, 4\}$ ,  $E = \{8, 9\}$ .

Az  $f = \{A, B\} \rightarrow \{C, E\}$  függőséggel való kompatibilitásra tekintsük példaként az alábbi  $\mathbf{R}^{(1)}$ ,  $\mathbf{R}^{(2)}$  és  $\mathbf{R}^{(3)}$  táblájukkal megadott  $\mathbf{D}(\underline{A})$ -beli relációsémákat.

$\mathbf{R}^{(1)}$ :	A	B	C	D	E	$\mathbf{R}^{(2)}$ :	A	B	C	D	E	$\mathbf{R}^{(3)}$ :	A	B	C	D	E
	6	3	1	4	9		6	3	2	4	8		6	4	1	4	9
	6	4	2	4	8		7	3	1	3	9		6	3	2	3	8
	7	3	2	3	8		7	3	1	4	9		6	4	1	3	9
	6	3	1	3	9		6	4	1	3	9		7	4	1	4	8
	6	4	2	3	8												

Figyeljünk fel arra, hogy  $\mathbf{R}^{(1)}$ ,  $\mathbf{R}^{(2)}$  és  $\mathbf{R}^{(3)}$  egyaránt kompatibilisek  $f$ -fel, továbbá

- ha  $\mathbf{R}^{(1)}$  és  $\mathbf{R}^{(2)}$  közül valamelyiket kibővítjük (monounió által) a másik egy rekordjával, akkor a kibővített relációséma már nem lesz kompatibilis  $f$ -fel, míg
- ha az  $\mathbf{R}^{(2)}$  és az  $\mathbf{R}^{(3)}$  között végezzük el ezt a rekordbővítést, akkor a kibővített relációséma is kompatibilis marad  $f$ -fel,
- ha viszont az  $\mathbf{R}^{(1)}$  és az  $\mathbf{R}^{(3)}$  között végezzük el ezt a rekordbővítést, akkor a kibővített relációséma attól függően lesz kompatibilis az  $f$ -fel, hogy mely rekordot vittük át a másik relációsémába!

Ennek a jelenségnek az értelmezésével a későbbiekben fogunk majd foglalkozni (az ortogonális-felosztás tárgyalásánál).

Tekintsük ezután az  $\mathbf{R}^{(3)}$  relációsémát. Ez nyilván kompatibilis az  $f$  függőséggel, és a függőségek egyszerű-függőségekre való felbonthatóságából következően természetesen a  $g = \{A, B\} \rightarrow \{E\}$  függőséggel is, de láthatóan például a  $h = \{C\} \rightarrow \{B\}$  függőséggel is.

**Struktúra-kompatibilitás, séma-kompatibilitás, ekvivalencia**

A fentiekben láttuk, hogy a relációsémák és a logikai-struktúrák szimmetrikus kapcsolatban állnak egymással. Ez nyilván kihat egyrészt a relációsémák egymásközi kapcsolatára, valamint hasonlóképpen a logikai-struktúrák egymásközi kapcsolatára is. Ennek alapján bevezetjük az alábbi fogalmakat:

Valamely  $\mathbf{R}_1, \mathbf{R}_2 \subseteq \mathbf{D}(\underline{A})$  relációsémákat *struktúra-kompatibilisnek*, vagy *ekvivalensnek* nevezzük, ha van olyan  $F \subseteq \mathbf{P}\{A\}$  nemtriviális logikai-struktúra, mely mindkettővel kompatibilis. Ezt  $\mathbf{R}_1 \# \mathbf{R}_2$  módon jelöljük.

Valamely  $F_1, F_2 \subseteq \mathbf{P}\{A\}$  logikai-struktúrákat *séma-kompatibilisnek*, vagy *ekvivalensnek* nevezzük, ha van olyan  $\mathbf{R} \subseteq \mathbf{D}(\underline{A})$  nemtriviális relációséma, mely mindkettővel kompatibilis. Ezt  $F_1 \# F_2$  módon jelöljük.

## Kompatibilitások és halmazműveletek kapcsolata

Legyen  $A$  egy attribútumhalmaz,  $F, F_1$  és  $F_2$   $A$ -val kompatibilis logikai-struktúra,  $R, R_1$  és  $R_2$  pedig  $A$ -val kompatibilis relációséma (azaz  $F, F_1, F_2 \subseteq P\{A\}$ , és  $R, R_1, R_2 \subseteq D(A)$ ).

### Állítás

1. *Részséma és részstruktúra kompatibilitása*  
Ha  $F \# R$ , akkor  $F' \# R'$  tetszőleges  $F' \subseteq F$  és tetszőleges  $R' \subseteq R$  esetén.
2. *Kompatibilis relációsémák metszete*  
Ha  $F \# R_1$  és  $F \# R_2$ , akkor  $F \# (R_1 \cap R_2)$ .
3. *Kompatibilis függőségek egyesítése*  
Ha  $F_1 \# R$  és  $F_2 \# R$ , akkor  $(F_1 \cup F_2) \# R$ .

### Bizonyítás

1. Egy  $R$  relációsémának egy  $F$  logikai-struktúrával való kompatibilitása azt jelenti, hogy az összes  $r, s \in R$  rekordon minden  $X \rightarrow Y \in F$  függőségre fennáll, hogy  $r \stackrel{X}{\sim} s$  esetén  $r \stackrel{Y}{\sim} s$  is teljesül. Ha pedig az összes rekordra vonatkozó minden függőségi feltétel teljesül, akkor e feltételek, vagy ezek tetszőleges részhalmaza is nyilván teljesül a rekordok tetszőleges  $R' \subseteq R$  részében is. ■
2. Ha az  $R_1$  és  $R_2$  minden rekordpárosára fennáll minden  $F$ -beli függőség, akkor nyilván az  $R_1$  és  $R_2$  közös részében levőkre is fennáll. Ha  $R_1 \cap R_2$  az üres halmaz, vagy csak egyetlen rekordból áll, akkor a korábbiak értelmében minden függőséggel kompatibilis. ■
3. Az, hogy  $F_1$  és  $F_2$  minden függősége az  $R$  minden rekordpárosára fennáll, pontosan azt jelenti, hogy az  $F_1$  és  $F_2$  egyesítése kompatibilis az  $R$ -rel. ■

### Megjegyzés

Bár a fenti állításokat nagyon egyszerű belátni, jelentőségük mégis igen nagy. A relációsémák és a logikai-struktúrák közötti kapcsolatban többször fogunk rájuk hivatkozni, igazi jelentőségük azonban majd a logikai-struktúrák algebrai vizsgálatánál fog látszani.

### Példa

Legyen  $A = \langle A, B, C, D \rangle$  egy attribútumvektor, ahol  $A = \{6, 7, 8\}$ ,  $B = \{3, 4, 5\}$ ,  $C = \{1, 2\}$ ,  $D = \{3, 4\}$ , továbbá  $F = \{f_1, f_2\}$  egy logikai-struktúra, melyben  $f_1 = \{A\} \rightarrow \{D\}$ ,  $f_2 = \{B\} \rightarrow \{C\}$ . valamint az alábbi, táblájával adott  $R$  relációséma.

$R$ : <u>ABCD</u>		$R'$ : <u>ABCD</u>	Láthatóan $R \# F$ mellett fennállnak a
6 3 1 4		6 3 1 4	következők is:
7 4 2 4	$R' \subseteq R$	8 3 1 3	$R \# f_1$ és $R \# f_2$ továbbá
8 3 1 3	$\Rightarrow$	6 5 1 4	$R' \# F, R' \# f_1$ és $R' \# f_2$ .
6 5 1 4			
8 4 2 3			

## Logikai-struktúra hatása relációsémára

### A logikai-struktúrák hatásának vizsgálata, ekvivalencia, ortogonalitás

A fentiekben szó volt arról, hogy a logikai-struktúrákat (függőség-halmazokat) a tervezés különböző fázisaiban (később tárgyalandó szempontoknak való megfelelés érdekében) különböző módon transzformáljuk, és ily módon – általában többször is – megváltoztatjuk az eredetihez (a gyakorlati szempontok alapján megfogalmazotthoz) képest.

Ezeknek a logikai-struktúráknak persze valamilyen szempontból ekvivalensnek kell lenniük. Pontos definíciót adunk majd erre az ekvivalenciára, ám az eddigiek alapján azt várjuk, hogy két logikai-struktúra (az attribútumok rögzített halmazán) akkor ekvivalens, ha segítségével ugyanazok az adatbázisok hozhatók létre. Persze egy gyakorlati adatbázisban soha sem fordulhat elő az elvben lehetséges összes rekord, így inkább azt lenne célszerű hangsúlyozni, hogy az ekvivalens logikai-struktúrák ugyanazokat a rekordokat „tiltják ki” a létrehozandó adatbázisokból, definiálva ezzel az ekvivalens logikai-struktúrák által meghatározott relációsémák halmazát. Nyilván e relációsémák közül a „legbővebb” (amely tehát az összes többi tartalmazza részhalmazként) jelöli ki a logikai-struktúra által engedélyezett, „maximális” adatbázis különböző rekordokkal való feltöltésének elvi határait. Csakhogy egyértelműen meghatározott-e ez a „legbővebb” relációséma? Sajnos nem! Nem csupán a (későbbi definíció szerint) ekvivalens logikai-struktúrák „legbővebb” relációsémái nem azonosak, de még egyetlen logikai-struktúrához tartozók is különböznek (amint az látható lesz az alábbi példában is). Ez azt jelenti, hogy az attribútumhalmaz, és a logikai-struktúra együttesen nem határozzák meg egyértelműen a „legbővebb” relációsémát! Az eddigiek alapján nem ezt vártuk...

Mi lehet e jelenség oka? Ennek megértéséhez a függőség definíciójából kell kiindulni. Figyeljünk fel arra, hogy a függőség „kényszerének” érvényesítéséhez

- legalább két rekordra van szükség, és
- a Descartes-szorzat bármely rekordjából kiindulhatunk.

Eszerint tehát a logikai-struktúra egyrészt

- egyetlen rekordját sem zárja ki a Descartes-szorzatnak, másrészt viszont
- különböző rekordokból kiindulva különböző relációsémákhoz juthatunk.

A keletkező relációsémák kapcsolatára az jellemző, hogy

- a relációsémák egy részének van közös része más relációsémákkal, sőt néhányat részhalmazként tartalmaz is, míg
- a relációsémák más része diszjunkt, azaz kölcsönösen nem tartalmazzák egymás rekordjait.

Összefoglalóan tehát azt mondhatjuk, hogy *egy logikai-struktúra a vele kompatibilis attribútumhalmaz fölötti Descartes-szorzatot felosztja (particionálja, azaz diszjunkt részhalmazokra, úgynevezett sémaosztályokra bontja).*

A fentiekből az következik, hogy a logikai-struktúrát alkotó függőségek mindegyik sémaosztályban, mint relációsémában teljesülnek. Az egyes sémaosztályok ebben a fontos dologban azonosak. Akkor hát miben különböznek?

A magyarázat nyilvánvalóan a függőség függvényjellegéből következik. Ha egy relációsémában egy  $\{B\} \rightarrow \{C\}$  függőség a  $B = 3$  értékhez a  $C = 1$  értéket rendeli, akkor nyilván e relációsémában nem lehet egy olyan összerendelés, mely a  $B = 3$  értékhez a  $C = 2$

értéket rendeli, noha ez épp úgy nincs ellentmondásban  $\{B\} \rightarrow \{C\}$  függőséggel, mint a másik. Ez a magyarázata tehát annak, hogy az egyes sémaosztályok, bár ugyanannak a logikai-struktúrának (ugyanazoknak a függőségi kényszereknek) felelnek meg, mégis diszjunkt halmazként elkülönülnek; más elemi leképezés kombinációkat tartalmaznak. Ez az erős elkülönülés indokolja, hogy a felosztásnak ezt a típusát *ortogonálisnak* nevezzük (amelynek pontos algebrai definícióját majd később adjuk meg).

### Néhány szó a logikai-struktúrákról

A logikai-struktúrák által végzett ortogonális-felosztás jelentőségét a gyakorlatban az adja, hogy fel kell ismernünk, a függőségi kényszerek ugyan biztosítják az adatbázisok konzisztenciáját (ellentmondás-mentességét), de semmilyen védelmet nem nyújtanak ahhoz, hogy az adatok megfeleljenek a valóságnak. Sőt, éppen a deklarált függőségek akadályozhatják meg, hogy egy adatbázisba a helyes adatok kerüljenek! Gondoljunk csak arra, hogy ha hibás adatokkal (például hibás személyi számmal) felvettünk valakit egy adatbázisba, akkor azt a személyt, aki valójában azokkal az adatokkal rendelkezik, már éppen a függőségek miatt nem tudjuk felvenni. Minden sémaosztály egy külön világ, de a valódi csak egy ezek közül!

Feltehető ezek után a kérdés, hogy helyes-e az adatbázisok működtetésébe beépíteni a függőségi megszorításokat?

Ha arra gondolunk, hogy mennyi adat kerül be tévesen (például adatbeviteli hibából adódóan) az adatbázisokba, akkor a válasz az, hogy nem! Ha arra gondolunk, hogy van olyan nyilvántartó és információ-feldolgozó rendszer, amely tele van hibás adatokkal, mégis képes befogadni a helyes adatokat is, sőt még a hibás adatok birtokában is képes általában megfelelő válaszokat adni a különböző „lekérdezésekre”, képes helyes döntéseket hozni, akkor a válasz megint csak az, hogy nem! (Hogy melyik ez a rendszer? Ott van mindjárt a kedves olvasó e sorokat olvasó szeme mögött...!)

Mi hát a magyarázata, hogy mégis annyira ragaszkodunk a függőségekhez? Először is ne feledjük, a függőségekben testesül meg tudásunk a világról! A függőségek fontos részét képezik annak a modellnek, amelyet alapos elemzés után építettünk fel egy adatbázis létrehozása érdekében. A függőségekre tehát biztos, hogy szükségünk van. Valószínűleg az alkalmazásával vannak inkább problémák. Talán nem annyira az adatok befogadásánál kellene azokat érvényesíteni, hanem inkább a lekérdezésekre adott válaszokban, a döntésekben. Éppen ez az a szemlélet, mely napjainkban új irányokba viszi az információtárolást. Ez az irány olyan kifejezésekkel jellemezhető, mint „szakértői rendszerek”, „tudás hálók”, „neurális architektúrák”, „asszociatív adatvizsgáló”, stb. Amíg viszont a technológia (és ez a hardver és szoftver technológiára egyaránt igaz) nem lesz elég hatékony, nem várható, hogy ezek kiszorítsák a jelenleg elterjedt rendszereket.

Addig pedig nem tehetünk mást, mint hogy bízunk azokban a kedves és szorgalmas hölgyekben, akik fáradhatatlanul töltik fel adatokkal korunk minden korábnál hatalmasabb nyilvántartó rendszereit, hogy azok segítségével aztán majd meghozhassák komoly és felelősségteljes urak a minket érintő döntéseiket...

## Néhány algebrai megjegyzés, ekvivalencia

Az ortogonális-felosztást a fent bemutatott tulajdonságai alapján az algebraiban *metszet-félhálónak* nevezik. Az elnevezést az indokolja, hogy létezik egy háló nevű algebrai struktúra is, melyben két olyan művelet van (a metszet és a monounióval való egyesítés), melyre teljesülnek az algebrai struktúrákban alapvetően elvárt tulajdonságok, nevezetesen a teljesesség, vagyis az, hogy bármely két hálóbeli elem között elvégezhető legyen a művelet, valamint a zártság, vagyis az, hogy minden előbbi művelet eredménye is a háló eleme legyen. (A hálónak még további érdekes tulajdonságai vannak, de ezekre itt most nem térünk ki.) A logikai-struktúrával felosztott Descartes-szorzat metszetfélháló, mivel csak egy „hálószerű” művelete van, ez a metszet. A metszet esetén a teljesség nyilván teljesül (hiszen egy attribútumhalmaz feletti bármely két relációséma, mint halmaz között nyilván elvégezhető a metszet művelet), de mi a helyzet a zártsággal? Nos ez is teljesül, hiszen a különböző sémaosztálybeli részhalmazokkal, mint relációsémákkal végezve a metszetet olyan részhalmazokat kapunk, melyekre, mint relációsémákra ismét teljesül az ortogonális-felosztást végző logikai-struktúrával való kompatibilitás. Ennek belátásánál a „Kompatibilitások és halmazműveletek” pontban bizonyított állításokra kell gondolnunk. (Vigyázat, az egyesítésre ez már nem igaz, azaz két, ugyanazzal a logikai-struktúrával kompatibilis relációséma egyesítéseként kapott relációséma egyáltalán nem biztos, hogy megtartja ezt a kompatibilitást. Lásd az alábbi példát!)

További „hálószerű” tulajdonsága a logikai-struktúrával felosztott Descartes-szorzatnak, hogy minden sémaosztálya háló, vagyis az egyes sémaosztályok részhalmazainak összessége, mint relációsémák halmaza a metszeten kívül már az egyesítésre is zárt, sőt van „minimális” eleme (az üres halmaz), és „maximális” eleme is. Ez utóbbi az a relációséma, mely a sémaosztály összes rekordját tartalmazza, vagyis maga a sémaosztály. E „hálószerű” tulajdonságok alapján definiált függőségműveletek alkotják majd azokat a logikai-struktúra transzformációkat, melyek a gyakorlati igényeket (elsősorban a minél kisebb redundanciájú adattárolást) kielégítő relációsémák megtervezését lehetővé teszik.

Most már meg tudjuk fogalmazni a logikai-struktúrák hatásának vizsgálatánál feltett kérdést, nevezetesen azt, hogy mit is jelent az ekvivalencia a logikai-struktúrák között. *Két logikai-struktúra ekvivalens valamely adott (és velük kompatibilis) attribútumhalmazon, ha ezen attribútumhalmaz fölötti Descartes-szorzatot azonos módon osztják fel.* A következőkben a fent elmondottakat fogjuk pontosabb algebrai formába önteni.

A fentiek szerint tehát egy logikai-struktúrához egyértelműen hozzárendelhető a vele kompatibilis attribútumhalmaz fölötti Descartes-szorzat egy felosztása, amit e logikai-struktúrával történő *ortogonális-felosztásnak* nevezünk.

E hozzárendelés a másik irányban is egyértelmű oly módon, hogy egy, a fentiekben említett (a későbbiekben definiálandó) függőségműveletekre zárt logikai-struktúrák, az úgynevezett *függőségcsaládok és a Descartes-szorzat ortogonális-felosztásai kölcsönösen egyértelműen meghatározzák egymást.* Az alábbiakban megadjuk a pontos algebrai definíciókat.

## A logikai-struktúra és a sémafelosztás kapcsolata

### Ortogonalis relációsémák

Legyen  $A$  egy attribútumhalmaz,  $F\{A\}$  egy vele kompatibilis logikai-struktúra, és  $R_1, R_2 \subseteq D\langle A \rangle$  két relációséma. Ekkor az  $R_1$  és  $R_2$  relációsémák *ortogonalisak*, azaz

$$R_1 \perp R_2,$$

ha bármelyikből kivéve egy tetszőleges rekordot és azzal a másikat kibővítve, a kapott relációséma nem kompatibilis az  $F\{A\}$  logikai-struktúrával.

A fenti definíciót formálisan az alábbi módon írhatjuk fel:

$$R_1 \perp R_2 \Leftrightarrow (\forall R_i, R_j \in \{R_1, R_2\}, R_i \neq R_j, \forall r \in R_i) : \neg (R_j \cup \{r\} \# F\{A\}).$$

#### Megjegyzés

Figyeljünk fel arra, hogy az ortogonalis relációsémák definíciója nem írja elő a relációsémák kompatibilitását, vagy inkompatibilitását a hivatkozott logikai-struktúrával! A megkötés kizárólag a kibővített relációsémára vonatkozik.

### Ortogonalis-felosztás (külső-felosztás)

Valamely  $R = R\langle A \rangle$  relációséma egy  $F \subseteq P\{A\}$  logikai-struktúra szerinti *ortogonalis-felosztásának*, vagy *külső-felosztásának* nevezzük az  $R$ -nek egy  $T_R^F$  osztálykritérium szerinti

$$\Phi^\perp[R|F] \triangleq \Phi[R|T_R^F]$$

sémafelosztását, ahol az osztálykritérium

$$T_R^F = (\text{minden } R_i \in \Phi^\perp[R|F], \text{ és } f \in F \text{ esetén } R_i \# f).$$

Az  $F$ -et az *ortogonalis-felosztás logikai-struktúrájának*, a felosztás elemeit alkotó halmazokat pedig ezúttal is *sémaosztályoknak* nevezzük.

#### Állítás

Ha az  $R\langle A \rangle$  relációséma ortogonalis-felosztását egy egységfüggőségből álló  $F$  logikai-struktúra szerint végezzük (tehát  $F = \{fI\}$ ,  $\text{Dom}(fI) = A$ ), akkor egységfelosztást kapunk, azaz

$$\Phi^\perp[R|F] = \Phi^E[R], \text{ ahol } \Phi^E[R] = R.$$

(Megjegyezzük, hogy az egységfelosztás fogalmát az előző fejezetben vezettük be.)

#### Bizonyítás

A felosztás tulajdonságai szerint ugyanis minden sémaosztályban, mint relációsémában teljesülnie kell az  $fI \in F$  egységfüggőségnek, ám, ha bármelyik sémaosztályból egy rekordot áttennénk valamelyik másikba, akkor az így kibővített relációsémában az már nem teljesülhet. Azonban az egységfüggőség minden relációsémában teljesül, tehát csak egy sémaosztály lehet, és az maga az egységfüggőséggel ortogonalisan felosztott relációséma. ■

## Az ortogonális-felosztás tulajdonságai

A fentiek alapján az alábbiakban foglalhatjuk össze az ortogonális-felosztás legfontosabb tulajdonságait:

- I. Az ortogonális-felosztás sémaosztályai - mint relációsémák - a felosztás logikai-struktúrájával kompatibilisek, és páronként ortogonálisak. (Ezt fejezi ki az ortogonális-felosztás osztálykritériuma.)  
Ebből már következik, hogy valamely ortogonális-felosztás során különböző osztályba kerülnek azok a rekordok, melyek a logikai-struktúra elemi függőségeihez tartozó vektorfüggvények különböző (ugyanarról a magról különböző képbe való) elemi leképezéseit reprezentálják.
- II. Az ortogonális-felosztás egyértelmű, és teljesülnek rá a felosztási tulajdonságok (lásd az előző fejezet „Halmazfelosztás osztálykritérium szerint” pontját).
- III. Ortogonális-felosztás esetén minden sémaosztály minden részhalmaza, mint relációséma kompatibilis a felosztás logikai-struktúrájával.
- IV. Ha egy relációséma kompatibilis egy logikai-struktúrával, akkor az eszerint végzett ortogonális felosztásnak egyetlen sémaosztálya lesz, amely maga a relációséma (vagyis az ortogonális-felosztás ekkor egy egységfelosztás).

Tehát az ortogonális-felosztáson keresztül a sémafelosztások a logikai-struktúrákkal kerültek igen szoros kapcsolatba. E kapcsolatra vonatkozó legfontosabb megállapítás, hogy *egy logikai-struktúra szerinti ortogonális-felosztás egyértelműen felosztja a relációsémákat e logikai-struktúrával kompatibilis, ortogonális részsémákra*. (Éppen ezért nevezzük ezt a felosztást ortogonálisnak.)

Végül algebrai módon összefoglaljuk az ortogonális-felosztás fenti I-IV. tulajdonságait. Tekintsünk valamely,  $A$  attribútumhalmaz fölötti  $\mathbf{R} = \mathbf{R}(\underline{A})$  relációsémát, egy  $F \subseteq \mathbf{P}\{A\}$  logikai-struktúrát, valamint egy  $\Phi^+[\mathbf{R}|F]$  ortogonális-felosztást az ehhez tartozó  $T = T_{\mathbf{R}}^F$  osztálykritériummal. Ekkor teljesülnek az alábbiak:

1. Az ortogonális-felosztás sémaosztályai - mint relációsémák - a felosztás logikai-struktúrájával kompatibilisek, és páronként ortogonálisak, azaz
  - 1.1. minden  $\mathbf{R}_i \in \Phi^+[\mathbf{R}|F]$  esetén  $\mathbf{R}_i \# F$ ,
  - 1.2. minden  $\mathbf{R}_i, \mathbf{R}_j \in \Phi^+[\mathbf{R}|F]$ ,  $\mathbf{R}_i \neq \mathbf{R}_j$  esetén  $\mathbf{R}_i \perp \mathbf{R}_j$ .
2. Az ortogonális-felosztás egyértelmű, és teljesülnek rá a felosztási tulajdonságok:
  - 2.1. minden  $\mathbf{R}_i \in \Phi^+[\mathbf{R}|F]$  esetén  $\mathbf{R}_i \subseteq \mathbf{R}$ ,
  - 2.2.  $\bigcup_{\mathbf{R}_i \in \Phi^+[\mathbf{R}|F]} \mathbf{R}_i = \mathbf{R}$ ,
  - 2.3. minden  $\mathbf{R}_i, \mathbf{R}_j \in \Phi^+[\mathbf{R}|F]$  esetén  $\mathbf{R}_i \cap \mathbf{R}_j = \emptyset$ .
  - 2.4. minden elemi halmazán (sémaosztályán) teljesül a  $T$  osztálykritérium, azaz minden  $\mathbf{R}_i \in \Phi^+[\mathbf{R}|F]$  esetén  $T(\mathbf{R}_i) = \text{IGAZ}$ , és
  - 2.5. bármely elemi halmazát kibővítve egy másik elemi halmazból származó elemmel, az így kapott halmazra már nem teljesül a  $T$  osztálykritérium, azaz minden  $\mathbf{R}_i, \mathbf{R}_j \in \Phi^+[\mathbf{R}|F]$ ,  $\mathbf{R}_i \neq \mathbf{R}_j$  és  $\underline{r} \in \mathbf{R}_i$  esetén  $T(\mathbf{R}_j \cup \{\underline{r}\}) = \text{HAMIS}$ .  
(Ez éppen az  $\mathbf{R}_i$  és az  $\mathbf{R}_j$  relációsémák ortogonalitását jelenti.)

3. Az ortogonális-felosztás minden sémaosztályának minden részhalmaza, mint relációs-séma kompatibilis a felosztás logikai-struktúrájával, azaz minden  $\mathbf{R}_i \in \Phi^\perp[\mathbf{R}|F]$ ,  $\mathbf{R}_{i1} \subseteq \mathbf{R}_i$  esetén  $\mathbf{R}_{i1} \# F$ .
4. Ha egy relációs-séma kompatibilis egy logikai-struktúrával, akkor az eszerint végzett ortogonális felosztás egy egységfelosztás, azaz ha  $\mathbf{R} \# F$ , akkor  $\Phi^\perp[\mathbf{R}|F] = \Phi^E[\mathbf{R}]$ , ahol  $\Phi^E[\mathbf{R}] = \mathbf{R}$ .

**Példa** (Az ortogonális-felosztás tulajdonságainak bemutatása)

Legyen  $\underline{A} = \langle A, B, C, D \rangle$  egy attribútumvektor, ahol  $A = \{6, 7, 8\}$ ,  $B = \{3, 4, 5\}$ ,  $C = \{1, 2\}$ ,  $D = \{3, 4\}$ , továbbá  $F = \{f_1, f_2\}$  egy logikai-struktúra, melyben  $f_1 = \{A\} \rightarrow \{D\}$ ,  $f_2 = \{B\} \rightarrow \{C\}$ . valamint az alábbi, táblájával adott  $\mathbf{R}$  relációs-séma.

*Feladat*

Vizsgáljuk meg a  $\Phi^\perp[\mathbf{R}|F]$  ortogonális-felosztást!

*Megoldás*

$\mathbf{R}$ : <u>ABCD</u>		$\mathbf{R}^{(1)}$ : <u>ABCD</u>	$\mathbf{R}^{(2)}$ : <u>ABCD</u>	$\mathbf{R}^{(3)}$ : <u>ABCD</u>	$\mathbf{R}^{(4)}$ : <u>ABCD</u>
6 3 1 4	$\Phi^\perp[\mathbf{R} F]$ $\Rightarrow$	6 3 1 4	6 4 1 3	6 4 1 4	7 5 1 3
7 4 2 4		7 4 2 4	7 3 2 3	7 3 2 4	6 5 1 3
8 3 1 3		8 3 1 3	8 4 1 4	8 4 1 3	
6 5 1 4		6 5 1 4	6 5 2 3	6 5 2 4	
8 4 2 3		8 4 2 3	8 3 2 4	8 3 2 3	
<u>6 4 1 3</u>		7 5 1 4	7 4 1 3	7 4 1 4	
7 3 2 3					
8 4 1 4		$\Downarrow$	$\Downarrow$	$\Downarrow$	
6 5 2 3					
<u>8 3 2 4</u>		$\mathbf{R}_1$ : <u>ABCD</u>	$\mathbf{R}_2$ : <u>ABCD</u>	$\mathbf{R}_3$ : <u>ABCD</u>	
6 4 1 4		6 3 1 4	6 4 1 3	6 4 1 4	
7 3 2 4		6 5 1 4	7 3 2 3	7 4 1 4	
8 4 1 3		8 3 1 3	8 4 1 4	8 4 1 3	
6 5 2 4					
<u>8 3 2 3</u>					
7 4 1 3					
7 4 1 4					
7 5 1 3					
7 5 1 4					
6 5 1 3					

- Vegyük észre, hogy az  $\mathbf{R}$  relációs-séma irreducibilis, és figyelembe véve az egyes attribútum értéktartományokat legfeljebb  $3 \times 3 \times 2 \times 2 = 36$  különböző rekordja lehetne, azaz  $\mu(\mathbf{D}(\underline{A})) = 36$ .
- Bemutatjuk, hogy  $\Phi^\perp[\mathbf{R}|F] = \{\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \mathbf{R}^{(3)}, \mathbf{R}^{(4)}\}$ , ahol az  $\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \mathbf{R}^{(3)}$  és  $\mathbf{R}^{(4)}$  sémaosztályok, mint relációs-sémák táblái a fentiek. (Az  $\mathbf{R}$  relációs-séma tábláját szaggatott vonalakkal tagoltuk a könnyebb ellenőrzés érdekében.)
  - Elsőként azt mutatjuk meg, hogy az  $\{\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \mathbf{R}^{(3)}, \mathbf{R}^{(4)}\}$  relációs-séma-halmaz valóban az  $\mathbf{R}$  relációs-séma felosztása. Ehhez az alábbi három feltételnek kell teljesülnie:

- 2.1.1. minden  $\mathbf{R}^{(i)} \in \Phi^\perp[\mathbf{R}|F]$  esetén  $\mathbf{R}^{(i)} \subseteq \mathbf{R}$ ,
  - 2.1.2.  $\mathbf{R}^{(1)} \cup \mathbf{R}^{(2)} \cup \mathbf{R}^{(3)} \cup \mathbf{R}^{(4)} = \mathbf{R}$ ,
  - 2.1.3. minden  $\mathbf{R}^{(i)}, \mathbf{R}^{(j)} \in \Phi^\perp[\mathbf{R}|F]$  esetén  $\mathbf{R}^{(i)} \cap \mathbf{R}^{(j)} = \emptyset$ .
- Láthatóan mindhárom feltétel teljesül.
- 2.2. Második lépésként azt kell ellenőrizni, hogy a  $\Phi^\perp[\mathbf{R}|F]$  elemi relációsémái mind kompatibilisek-e az  $F$  logikai-struktúrával, vagyis azt, hogy az  $\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \mathbf{R}^{(3)}$  és  $\mathbf{R}^{(4)}$  relációsémákban az  $f_1$  és  $f_2$  függőségek mindegyike fennáll-e. Láthatóan fennáll.
  - 2.3. Végül meg kell vizsgálni, hogy  $\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \mathbf{R}^{(3)}$  és  $\mathbf{R}^{(4)}$  relációsémák páronként ortogonálisak-e, vagyis azt, hogy ha bármelyikből egy rekordot hozzáveszünk bármelyik másikhoz (monounióval), akkor az így kapott relációséma elveszíti-e az  $F$  logikai-struktúrával való kompatibilitást. Ezt kissé hosszadalmas ellenőrizni, de próbálgatással és egy kis odafigyeléssel beláthatjuk, hogy valóban elveszik az  $F$ -el való kompatibilitás, tehát teljesül a páronkénti ortogonalitás.
3. Tanulságos megvizsgálni a  $\Phi^\perp[\mathbf{R}|F]$  ortogonális-felosztás sémaosztályainak néhány részhalmazát, mint relációsémát. Tekintsük tehát az  $\mathbf{R}_1, \mathbf{R}_2$  és  $\mathbf{R}_3$  relációsémákat, ahol minden  $\mathbf{R}_i \subseteq \mathbf{R}^{(i)}$  és  $\mathbf{R}^{(i)} \in \Phi^\perp[\mathbf{R}|F]$ .
- Tapasztalatainkat az alábbiakban foglalhatjuk össze:
- 3.1. Tudjuk korábbról, hogy ha egy relációséma kompatibilis egy függőséggel, akkor ezt a tulajdonságot minden részhalmazra is örökli. Ezek után tehát nem meglepő, hogy minden  $\mathbf{R}_i$  esetén  $\mathbf{R}_i \# F$ .
  - 3.2. Ha elvégezzük a megfelelő ellenőrzéseket, akkor azt találjuk, hogy az  $F$  logikai-struktúrára vonatkozóan az  $\mathbf{R}_1$  és az  $\mathbf{R}_2$ , továbbá az  $\mathbf{R}_2$  és az  $\mathbf{R}_3$  relációsémák ortogonálisak, azaz  $\mathbf{R}_1 \perp \mathbf{R}_2$ , továbbá  $\mathbf{R}_2 \perp \mathbf{R}_3$ . Helytelen volna azonban ebből arra következtetni, hogy az  $\mathbf{R}_1 \perp \mathbf{R}_3$  is teljesül.
  - 3.3. Szintén elvégezve az ellenőrzéseket azt találjuk, hogy az  $\mathbf{R}_1$  és az  $\mathbf{R}_3$  relációsémák nem ortogonálisak, hiszen az egyikük bármelyik rekordjával bővítve a másikat, szintén az  $F$  logikai-struktúrával kompatibilis relációsémát kapunk.

# Függőségialgebra

## Alapfogalmak

A függőségek közül a gyakorlati alkalmazás szempontjából kiemelkedő fontosságúak azok, amelyek a teljes attribútumhalmazra képeznek le, mivel ezek révén lehetőség van arra, hogy a relációk egyes rekordjait (a relációtábla egyes sorait) az esetleges egyezéstől eltekintve egyértelműen kijelöljük.

### Kulcs (kulcshalmaz, elsődleges kulcs, idegen kulcs)

Egy  $R(A)$  relációsémában valamely  $K \subseteq A$  attribútumhalmazt *kulcsnak* nevezünk, ha

1. Létezik  $K \rightarrow A$  függőség, és
2. minden  $X \subseteq K$  attribútumhalmaz és  $X \rightarrow A$  függőség esetén  $X = K$ .

Ezekután egy  $R(A)$  relációséma kulcsainak halmazát, vagyis a *kulcshalmazt*

$$K\{R(A)\}$$

módon jelöljük. Ezek közül egyet önkényesen kiválasztva, azt *elsődleges kulcsnak* nevezük, a többi *kulcsjelöltnek*. Ha egy kulcs egyetlen attribútumból áll (tehát egyelemű attribútumhalmaz), akkor *egyszerű kulcsnak*, egyébként pedig *összetett kulcsnak* nevezzük.

Legyen  $R(A)$  és  $S(B)$  két relációséma, továbbá  $K \subseteq A \cap B$  és  $K \in K\{R(A)\}$  az elsődleges kulcs az  $R(A)$  relációsémában. Ekkor a  $K$  attribútumhalmazt *idegen kulcsnak* nevezzük az  $S(B)$  relációsémában.

#### Megjegyzés

Tehát egy relációséma kulcsa az attribútumoknak egy olyan halmaza, mely egyértelműen azonosítja a relációséma minden rekordját, de bármely attribútum elhagyásával ez már nem teljesül. (Az „egyértelmű azonosítás” természetesen relációsémára vonatkozik és nem relációra, hiszen abban azonos rekordok is lehetnek!)

Ha egy konkrét relációsémában az attribútumnevek vektorát adjuk meg, akkor a kulcs-attribútumokat aláhúzással fogjuk jelölni.

Az adatmodell tervezője a relációséma kulcsai közül valamilyen gyakorlati szempont alapján választ ki egyet és nevezi elsődleges kulcsnak. Ezután az adott tábla kezeléséhez az elsődleges kulcsot fogja használni.

Nyilván minden relációsémának van kulcsa, ha más nem, hát a teljes attribútumhalmaz!

#### Példa

Tekintsük az  $R(\text{TELEPÜLÉS}, \text{UTCA}, \text{HÁZSZÁM}, \text{IRÁNYÍTÓSZÁM})$  relációsémát. Ebben nyilvánvalóan fennállnak az alábbi funkcionális-függőségek:

$$\{\text{TELEPÜLÉS}, \text{UTCA}, \text{HÁZSZÁM}\} \rightarrow \{\text{IRÁNYÍTÓSZÁM}\}, \text{ és } \\ \{\text{IRÁNYÍTÓSZÁM}\} \rightarrow \{\text{TELEPÜLÉS}\}.$$

Az is könnyen belátható, hogy a

$$\{\text{TELEPÜLÉS}, \text{UTCA}, \text{HÁZSZÁM}\}, \text{ valamint az } \\ \{\text{UTCA}, \text{HÁZSZÁM}, \text{IRÁNYÍTÓSZÁM}\}$$

attribútumhalmazok kulcsai a fenti  $\mathbf{R}$  relációsémának. Így az előző megjegyzés értelmében a kulcsattribútumok jelzésére alkalmazhatjuk az

$\mathbf{R}(\underline{\text{TELEPÜLÉS}}, \underline{\text{UTCA}}, \underline{\text{HÁZSZÁM}}, \underline{\text{IRÁNYÍTÓSZÁM}})$ , vagy az  
 $\mathbf{R}(\text{TELEPÜLÉS}, \text{UTCA}, \text{HÁZSZÁM}, \text{IRÁNYÍTÓSZÁM})$  jelöléseket.

## Szuperkulcs

Legyen  $\mathbf{R}(\underline{A})$  egy relációséma,  $Y \subseteq X \subseteq A$ . Ha  $Y \in \mathbf{K}\{\mathbf{R}(\underline{A})\}$ , azaz kulcs, akkor az  $X$  attribútumhalmazt *szuperkulcsnak* nevezzük.

### Megjegyzés

A definíció szerint tehát egy kulcs bármely bővítése szuperkulcs (a bővítést természetesen az adott relációséma attribútumhalmazából végezzük), és nyilván minden kulcs egyben szuperkulcs is.

A szakirodalomban helyenként (főleg a korai cikkekben) az általunk definiált kulcsfogalmat minimális kulcsnak nevezik, és ennek megfelelően a kulcs definíciójából elhagyják a 2. pontot.

## A logikai-struktúrák bővítése

Az ortogonális-felosztás definíciójánál említettük, hogy bizonyos (a későbbiekben definiálandó) függőségműveletekre zárt logikai-struktúrák, az úgynevezett függőségcsaládok és az ezekkel kompatibilis attribútumhalmaz fölötti Descartes-szorzat ortogonális-felosztásai kölcsönösen egyértelműen meghatározzák egymást. Azt is elmondtuk, hogy az egyes osztályokban lévő relációsémák metszETFélhálót alkotnak, ami (többek között) azt jelenti, hogy a relációsémák a metszETFélpézésre zártak. Láthatjuk tehát, hogy a logikai-struktúrák és a relációsémák között igen szoros a kapcsolat.

Az alábbiakban, adott relációsémához tartozó logikai-struktúrák bizonyos bővítési tulajdonságairól lesz szó. E tulajdonságok fognak bennünket elvezetni a korábban említett függőségműveletekhez, és az ezekre zárt logikai-struktúrákhoz, a függőségcsaládokhoz.

A logikai-struktúrák és a relációsémák szoros kapcsolata alapján nem meglepő, hogy a logikai-struktúrák számos tulajdonsága a relációsémák tulajdonságaiból levezethető. Az alábbiakban először bevezetjük e szoros kapcsolat legfontosabb fogalmait, majd pedig a funkcionális-függőségekből álló logikai-struktúrák Armstrongtól származó bővítési tulajdonságait (az úgynevezett Armstrong-szabályokat), melyeket a szakirodalomban esetenként axiómáknak is neveznek (bár ezek az említett levezetésekkel keresztül bizonyíthatóak, így helyesebb lenne a tétel elnevezés).

## Relációséma burokstruktúrája

Legyen  $A$  egy attribútumhalmaz,  $\mathbf{R}(\underline{A})$  egy relációséma és  $\mathbf{P}\{A\}$  a hatványfüggőség az  $A$  fölött. Ekkor az  $\mathbf{R}(\underline{A})$  relációséma burokstruktúrája

$$\mathbf{F}\{\mathbf{R}(\underline{A})\} \triangleq \{f \mid f \in \mathbf{P}\{A\}, f \# \mathbf{R}(\underline{A})\},$$

mely nem más, mint az  $\mathbf{R}(\underline{A})$  relációséma legbővebb logikai-struktúrája.

*Megjegyzés*

Felmerülhet a kérdés, hogy a fenti definícióban mi biztosítja azt, hogy  $\mathbf{F}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  a legbővebb legyen az  $\mathbf{R}\langle\mathbf{A}\rangle$  relációséma logikai-struktúrái közül. Gondoljunk egy halmaznak az elemei tulajdonságával való megadására. Az előző fejezet elején, a jelölések bevezetésénél a  $\{x \mid T(x)\}$  alakot úgy értelmeztük, mint mindazon  $x$ -ek halmazát, melyekre  $T(x)$  teljesül. Jelen esetben tehát  $T(f) = (f \in \mathbf{P}\{A\}, f \# \mathbf{R}\langle\mathbf{A}\rangle)$  jelenti mindazon  $\mathbf{P}\{A\}$ -beli függőségeket, melyek kompatibilisek  $\mathbf{R}\langle\mathbf{A}\rangle$ -val, vagyis valóban az  $\mathbf{R}\langle\mathbf{A}\rangle$  legbővebb logikai-struktúráját.

**Állítás**

Egy  $\mathbf{R}\langle\mathbf{A}\rangle$  relációséma  $\mathbf{F}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  burokszerkeztúrája egyértelműen meghatározott.

*Bizonyítás*

A függőségek valamely relációsémával való kompatibilitása egymástól független, vagyis egy  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémával kompatibilis függőségek halmazát bármelyik megfelelő függőséggel kezdjük is felépíteni, az összes ilyen függőséggel való bővítés elvégezhető. Ez pedig azt jelenti, hogy csak egyetlen  $\mathbf{F}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  logikai-struktúra állítható elő. ■

**Logikai-struktúra buroksémája, burokséma-halmaza**

Legyen  $A$  egy attribútumhalmaz,  $F\{A\}$  egy logikai-struktúra és  $\mathbf{D}\langle\mathbf{A}\rangle$  a Descartes-szorzat az  $A$  fölött. Ekkor az  $F\{A\}$  *logikai-struktúra burokséma-halmaza*

$$\mathbf{R}\{F\{A\}\} \triangleq \{ \mathbf{R} \mid \mathbf{R} \subseteq \mathbf{D}\langle\mathbf{A}\rangle, \mathbf{R} \# F\{A\}, \forall \mathbf{r} \in (\mathbf{D}\langle\mathbf{A}\rangle \setminus \mathbf{R}) : \neg (\mathbf{R} \cup \{\mathbf{r}\}) \# F\{A\} \},$$

mely nem más, mint az  $F\{A\}$  logikai-struktúra legbővebb relációsémáinak halmaza.

*Megjegyzés*

Ezúttal külön feltétel megadásával kellett gondoskodnunk arról, hogy a definíció valóban az  $F\{A\}$  logikai-struktúra legbővebb relációsémáit állítsa elő. Nyilvánvaló ugyanis, hogy ha a  $\forall \mathbf{r} \in (\mathbf{D}\langle\mathbf{A}\rangle \setminus \mathbf{R}) : \neg (\mathbf{R} \cup \{\mathbf{r}\}) \# F\{A\}$  feltételt elhagyjuk, akkor  $\mathbf{R}\{F\{A\}\}$  az összes,  $F\{A\}$ -val kompatibilis relációsémát tartalmazná, és nem csupán a legbővebbeket.

**Állítás**

Adott  $\mathbf{D}\langle\mathbf{A}\rangle$  Descartes-szorzatban egy  $F\{A\}$  logikai-struktúra buroksémája általában nem egyértelműen meghatározott, azonban az  $F\{A\}$  burokséma-halmaza már egyértelmű.

*Bizonyítás*

A bizonyítás során induljunk ki valamely tetszőleges  $\mathbf{r} \in \mathbf{D}\langle\mathbf{A}\rangle$  rekordból és az  $\mathbf{R} = \{\mathbf{r}\}$  relációsémából (mely nyilván minden  $A$  fölötti függőséggel kompatibilis). Ezután bővítsük az  $\mathbf{R}$ -et (monounió módon) olyan  $\mathbf{D}\langle\mathbf{A}\rangle$ -beli rekordokkal, hogy teljesüljön az  $F\{A\}$ -val való kompatibilitás. Tegyük ezt mindaddig, amíg az alkalmas rekordok el nem fogytak.

A bizonyítás első alapkérdése az, hogy vajon különböző rekordokból elindulva ugyanahhoz a relációsémához jutunk-e.

Tekintsünk egy  $F\{A\} = \{X \rightarrow Y\}$  logikai-struktúrát (ahol tehát  $X, Y \subseteq A$ ), továbbá olyan  $\mathbf{r}, \mathbf{s} \in \mathbf{D}\langle\mathbf{A}\rangle$  rekordokat, melyekre  $\mathbf{r} \vdash_X = \mathbf{s} \vdash_X$ , de  $\mathbf{r} \vdash_Y \neq \mathbf{s} \vdash_Y$  (ahol  $\underline{X}$  és  $\underline{Y}$  az  $X$  és  $Y$  valamely vektorai). Nyilvánvaló, hogy a fenti módon az  $\mathbf{r}$ -ből és az  $\mathbf{s}$ -ből kiindulva egyaránt felépíthetünk olyan  $\mathbf{R}_r$  és  $\mathbf{R}_s$  relációsémát, melyek mindegyike kompatibilis az  $F\{A\}$ -val, de az  $\mathbf{R}_r$  nem tartalmazhatja az  $\mathbf{s}$  rekordot,  $\mathbf{R}_s$  pedig az  $\mathbf{r}$  rekordot. A fenti kérdésre adott válaszuk tehát az, hogy nem. Ezzel az állítás első felét bebizonyítottuk.

A bizonyítás másik alapkérdése az, hogy vajon adott logikai-struktúrához adott Descartes-szorzatban tartozhat-e két különböző burokséma-halmaz.

Különböző burokséma-halmazokhoz két úton lehetne eljutni, rekordelhagyással és rekordbővítéssel. A burokséma-halmaz definíciója szerint minden burokséma maximális, azaz nem bővíthető a logikai-struktúrával való kompatibilitás megtartása mellett. Ebből következik, hogy egyetlen buroksémából sem hagyható el rekord, hiszen akkor nem volna maximális, de éppen ezért bővíteni sem lehet. Tehát a második kérdésre adott válaszuk is az, hogy nem. Így az állítás második felét is bebizonyítottuk. ■

## A burokséma és az ortogonális felosztás kapcsolata

Legyen  $A$  egy attribútumhalmaz,  $F\{A\}$  egy logikai-struktúra és  $\mathbf{D}\langle A \rangle$  a Descartes-szorzat az  $A$  fölött. Ekkor igaz a következő állítás

$$\mathbf{R}\{F\{A\}\} = \Phi^\perp[\mathbf{D}\langle A \rangle | F\{A\}],$$

vagyis egy  $F\{A\}$  logikai-struktúra buroksémáinak halmaza nem más, mint a  $\mathbf{D}\langle A \rangle$  Descartes-szorzat ortogonális felosztása az  $F\{A\}$  logikai-struktúra szerint.

### Bizonyítás

Az ortogonális felosztás definícióját a  $\mathbf{D} = \mathbf{D}\langle A \rangle$ , valamint  $F = F\{A\}$  behelyettesítéssel alkalmazva kapjuk, hogy

$$\Phi^\perp[\mathbf{D} | F] = \Phi[\mathbf{D} | T_{\mathbf{D}}^F],$$

ahol az osztálykritérium

$$T_{\mathbf{D}}^F = (\text{minden } \mathbf{R}_i \in \Phi^\perp[\mathbf{D} | F], \text{ és } f \in F \text{ esetén } \mathbf{R}_i \# f).$$

A  $\Phi^\perp[\mathbf{D} | F]$  ortogonális felosztás sémaosztályai, mint relációsémák tehát kompatibilisek az  $F$  logikai-struktúrával, és a felosztás tulajdonságából következően a kompatibilis relációsémák közül a legbővebbek is. ■

## Logikai-struktúra burokstruktúrája, ekvivalens bővítése

Legyen  $A$  egy attribútumhalmaz és  $F\{A\}$  egy ezzel kompatibilis logikai-struktúra. Ekkor az  $F\{A\}$  logikai-struktúra burokstruktúrája

$$\mathbf{F}\{F\{A\}\} \triangleq \mathbf{F}\{\mathbf{R}_i\}, \text{ ahol } \mathbf{R}_i \in \mathbf{R}\{F\{A\}\}, \text{ azaz}$$

$$\mathbf{F}\{F\{A\}\} \triangleq \{f | f \in \mathbf{P}\{A\}, f \# \mathbf{R}\langle A \rangle, \exists \mathbf{R}_i \in \mathbf{R}\{F\{A\}\} : f \# \mathbf{R}_i\},$$

mely tehát nem más, mint egy, az  $F\{A\}$  logikai-struktúrával kompatibilis  $\mathbf{R}_i$  relációséma legbővebb logikai-struktúrája.

Az  $\mathbf{F}\{F\{A\}\}$  logikai-struktúrát az  $F\{A\}$  logikai-struktúra *felosztástartó bővítésének*, *séma-kompatibilis bővítésének*, vagy *ekvivalens bővítésének* is nevezzük.

### Állítás

1. Egy logikai-struktúra és burokstruktúrája ekvivalensek, azaz tetszőleges  $F\{A\}$  logikai-struktúra esetén

$$F\{A\} \# \mathbf{F}\{F\{A\}\}.$$

2. Egy logikai-struktúrának és a burokstruktúrájának burokséma-halmaza megegyezik, azaz tetszőleges  $F\{A\}$  logikai-struktúra esetén

$$\mathbf{R}\{F\{A\}\} = \mathbf{R}\{\mathbf{F}\{F\{A\}\}\}.$$

3. Egy  $F\{A\}$  logikai-struktúra legtagabb ekvivalense  $\mathbf{F}\{F\{A\}\}$ .

#### Bizonyítás

1. A burokséma és az ortogonális felosztás fent bemutatott kapcsolata szerint az  $F\{A\}$  logikai-struktúra  $\mathbf{R}\{F\{A\}\}$  buroksémája nem más, mint a  $\mathbf{D}\langle A \rangle$  Descartes-szorzat  $F\{A\}$  szerinti ortogonális felosztása. Mivel az ennek során kapott  $\mathbf{R}_i$  sémaosztályok, mint relációsémák mind kompatibilisek  $F\{A\}$ -val (az ortogonális-felosztás definíciója szerint), így ezen  $\mathbf{R}_i$  relációsémák bármely struktúrája nyilván séma-kompatibilis, azaz ekvivalens  $F\{A\}$ -val. ■
2. A burokséma fogalmának bevezetésekor megmutattuk, hogy egy adott  $\mathbf{D}\langle A \rangle$  Descartes-szorzatban egy  $F\{A\}$  logikai-struktúra burokséma-halmaza egyértelmű. Mihez kötődik azonban ez az egyértelműség? Ez azt jelentené, hogy két különböző logikai-struktúrának a burokséma-halmaza különböző? Ha megvizsgáljuk a burokséma-halmaz definícióját, akkor azt láthatjuk, hogy e definíció az egyes buroksémákat az adott logikai-struktúrával való kompatibilitáshoz köti. Ezek szerint ha két logikai-struktúra ekvivalens, akkor ugyanazokat a buroksémákat kapjuk, tehát a burokséma-halmaz megegyezik. Az előző állítás szerint  $F\{A\}$  és  $\mathbf{F}\{F\{A\}\}$  ekvivalensek, tehát a 2. állítás is igaz. ■
3. A 3. állítás már nyilvánvalóan következik az előző kettőből. ■

#### Megjegyzés

Tehát éppen a fenti állítások indokolják azt, hogy egy logikai-struktúra burokstruktúráját felosztástartó, séma-kompatibilis, vagy ekvivalens bővítésnek nevezzük.

### Az Armstrong-szabályok

Az Armstrong-szabályok a burokstruktúrák bizonyos struktúrabővítésekre való zártságát mondják ki. Ennek jelentőségét az adja, hogy segítségükkel lehetőségünk van valamely gyakorlati megfontolásokból előállított logikai-struktúra helyettesítésére egy vele ekvivalens (azaz séma-kompatibilis) logikai-struktúrával, amely más szempontokból (a későbbiekben bemutatandó normalizálások szempontjából) megfelelőbb.

A logikai-struktúrák ekvivalenciája – mint a korábbiakból tudjuk – azt jelenti, hogy e logikai struktúrák valamely attribútumhalmaz feletti Descartes-szorzatot azonos módon osztályoznak (particionálnak) kompatibilis relációsémákra. Mivel ezek a relációsémák – bár egymást kizáró módon – de végülis ugyanazon feladat megengedett adatbázisait reprezentálják, így azt mondhatjuk, hogy az ekvivalens logikai-struktúrák éppen attól ekvivalensek, hogy logikailag ugyanazt a feladatmegoldást eredményezik.

Amikor tehát az Armstrong-szabályok segítségével létrehozunk egy logikai-struktúra legtagabb ekvivalensét, akkor nem másat teszünk, mint kijelöljük azt a logikai-struktúrát, melynek bizonyos részhalmazai adják az egyéb szempontból megfelelőbb, ekvivalens logikai-struktúrát.

Legyen ezek után  $A$  egy attribútumhalmaz,  $X, Y, V, Z \subseteq A$ , és  $\mathbf{R}\langle A \rangle$  egy relációséma az  $A$  fölött. Ekkor az  $\mathbf{R}\langle A \rangle$  relációséma  $\mathbf{F}\{\mathbf{R}\langle A \rangle\}$  burokstruktúrájára teljesülnek az alábbiak:

**A1. Reflexivitási szabály**

Ha  $Y \subseteq X \subseteq A$ ,  
akkor  $X \rightarrow Y \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ .

**A2. Bővítési szabály**

Ha  $X \rightarrow Y \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$  és  $Z \subseteq A$ ,  
akkor  $X \cup Z \rightarrow Y \cup Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ .

**A3. Transzitivitási szabály**

Ha  $X \rightarrow Y \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$  és  $Y \rightarrow Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ ,  
akkor  $X \rightarrow Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ .

**A4. Egyesítési szabály**

Ha  $X \rightarrow Y \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$  és  $X \rightarrow Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ ,  
akkor  $X \rightarrow Y \cup Z$ .

**A5. Pszeudotranzitivitási szabály**

Ha  $X \rightarrow Y \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$  és  $V \cup Y \rightarrow Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ ,  
akkor  $X \cup V \rightarrow Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ .

**A6. Dekompozíciós szabály**

Ha  $X \rightarrow Y \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$  és  $Z \subseteq Y$ ,  
akkor  $X \rightarrow Z \in \mathbf{F}\{\mathbf{R}\langle A \rangle\}$ .

**Megjegyzés**

A második három szabály levezethető az első háromból (ezért az **A1-A3** szabályokat *alapvető Armstrong szabályoknak* is nevezzük), de az első három is levezethető a függőség definíciójából és egyéb alapfogalmakból. E terület tipikus gondolkodásmódjának érzékeltetése érdekében az alábbiakban bebizonyítjuk az **A1** és az **A4** szabályt.

**Az A1 szabály bizonyítása**

(az alrekord, a relatív azonos rekordok, és a függőség definíciója alapján):

- a.) A függőség definíciója szerint a  $X \rightarrow Y$  állítás ekvivalens azzal az állítással, hogy minden  $\underline{r}, \underline{s} \in \mathbf{R}\langle A \rangle$  esetén ha  $\underline{r} \stackrel{X}{\sim} \underline{s}$  teljesül, akkor  $\underline{r} \stackrel{Y}{\sim} \underline{s}$  is fennáll.
- b.) Az  $\underline{r} \stackrel{X}{\sim} \underline{s}$  teljesülésének feltétele a relatív azonos rekordok 2. definíciója szerint az, hogy minden  $A_i \in X$  esetén az  $\underline{r}(A_i) = \underline{s}(A_i)$  komponens-egyezőség teljesüljön.
- c.) Ha azonban a komponens-egyezőség a  $X$  attribútum halmaz egészén teljesül, akkor nyilván teljesül egy  $Y \subseteq X$  részhalmazon is, azaz minden  $A_i \in Y$  esetén is  $\underline{r}(A_i) = \underline{s}(A_i)$ , amiből pedig következik, hogy  $\underline{r} \stackrel{Y}{\sim} \underline{s}$  is fennáll.
- d.) A b.) és c.) állítások szerint, ha  $Y \subseteq X$ , akkor minden  $\underline{r}, \underline{s} \in \mathbf{R}\langle A \rangle$  esetén az  $\underline{r} \stackrel{X}{\sim} \underline{s}$  teljesüléséből következik  $\underline{r} \stackrel{Y}{\sim} \underline{s}$ . Ezt összevetve az a.) állítással kapjuk, hogy ha  $Y \subseteq X$ , akkor  $X \rightarrow Y$ , amit bizonyítanunk kellett. ■

**Az A4 szabály bizonyítása**

(a bővítési és a transzitivitási szabályok alapján):

- a.) Induljunk ki az egyik feltételként adott  $X \rightarrow Y$  kifejezésből. Az **A2** (bővítési) szabály alapján bővítsük e kifejezést jobbról  $X$ -szel. Ekkor a  $X \cup X \rightarrow Y \cup X$  kifejezést kapjuk. Mivel  $X \cup X = X$ , így ez leegyszerűsödik  $X \rightarrow Y \cup X$  alakúra.
- b.) Induljunk ki most a másik feltételként adott  $X \rightarrow Z$  kifejezésből. Ezt bővítsük balról az  $Y$ -nal. Ekkor az  $Y \cup X \rightarrow Y \cup Z$  kifejezést kapjuk.

- c.) Az a.) és b.) állítások szerint, ha  $X \rightarrow Y$  és  $X \rightarrow Z$ , akkor  $X \rightarrow Y \cup Z$  és  $Y \cup X \rightarrow Y \cup Z$ .  
E két utóbbi kifejezésre alkalmazva azonban az **A3** (transzitivitási) szabályt, kapjuk az  $X \rightarrow Y \cup Z$  kifejezést. Ezzel bebizonyítottuk az **A4** szabályt. ■

### Az Armstrong-bővítés

Valamely  $F\{A\}$  logikai-struktúrának az Armstrong-szabályok alapján történő bővítését *Armstrong-bővítésnek* nevezzük.

### Az Armstrong-szabályok következményei

Legyen  $R\langle A \rangle$  egy relációséma és  $X, Y, V, Z \subseteq A$ . Ekkor az alábbi állítások az Armstrong-szabályokból levezethetők. (Szigorúan véve már az **A4-A6** Armstrong-szabályok is következményeknek tekinthetők, nevezetesen az **A1-A3** Armstrong-szabályok következményei.)

#### K1. következmény

Ha  $X \rightarrow Y$  és  $V \subseteq X$ , akkor  $X \rightarrow Y \cup V$ .  
(Következik a reflexivitási és az egyesítési szabályból.)

#### K2. következmény

Ha  $X \rightarrow Y$ ,  $V \supseteq X$  és  $Z \subseteq Y$ , akkor  $V \rightarrow Z$ .  
(Következik a reflexivitási, az egyesítési és a transzitivitási szabályból.)

#### K3. következmény

Ha  $X \rightarrow Y$ ,  $X \rightarrow V$  és  $Y \cup V = A$ , akkor  $X \rightarrow A$ , tehát  $X$  egy superkulcs.  
(Következik az egyesítési szabályból.)

#### K4. következmény

Ha  $X \rightarrow Y$  és  $Y = \{A_1, A_2, \dots, A_k\}$ , akkor  $X \rightarrow \{A_1\}$ ,  $X \rightarrow \{A_2\}$ , ...,  $X \rightarrow \{A_k\}$ .  
(Következik a dekompozíciós szabályból.)

### Kulcs meghatározása az Armstrong-szabályok segítségével

Az alábbi példán bemutatjuk, hogy miként lehet attribútumhalmazával és logikai-struktúrájával adott relációséma esetén igazolni az attribútumok egy részhalmazáról, hogy azok kulcsot alkotnak.

#### Példa

Legyen  $R\langle A \rangle$  egy relációséma, melyben

$A = \{ \text{NÉV, SZÜL\_DÁTUM, ADÓSZÁM, LAKHELY, BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \}$ .

Egy tipikus munkahelyi környezetben feltételezhető (lásd „A funkcionális-függőségek gyakorlati meghatározása” pontban tett megjegyzéseket!) az  $F\{A\} = \{f_1, f_2, f_3, f_4, f_5\}$  logikai-struktúra, ahol

$f_1 = \{ \text{NÉV, SZÜL\_DÁTUM} \} \rightarrow \{ \text{ADÓSZÁM} \}$ ,  
 $f_2 = \{ \text{ADÓSZÁM} \} \rightarrow \{ \text{NÉV, SZÜL\_DÁTUM} \}$ ,  
 $f_3 = \{ \text{NÉV, SZÜL\_DÁTUM} \} \rightarrow \{ \text{LAKHELY, BEOSZTÁS} \}$ ,  
 $f_4 = \{ \text{BEOSZTÁS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}$ ,  
 $f_5 = \{ \text{FIZ\_OSZTÁLY} \} \rightarrow \{ \text{FIZETÉS} \}$ .

*Feladat*

Bizonyítsuk be, hogy  $\mathbf{K}\{\mathbf{R}\langle\mathbf{A}\rangle\} = \{K_1, K_2\}$ , ahol

$K_1 = \{\text{NÉV}, \text{SZÜL\_DÁTUM}\}$ , és

$K_2 = \{\text{ADÓSZÁM}\}$ .

*Megoldás*

A bizonyítás három lépésből fog állni. Először belátjuk, hogy  $K_1$  és  $K_2$  szuperkulcsok (a.), második lépésként, hogy kulcsok (b.), végül pedig, hogy az  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémában nincs több kulcs (c.). A bizonyítás során az **A1-A6** Armstrong-szabályokra fogunk hivatkozni.

a1.) Az **A1** (reflexivitási) szabály szerint

$$f_6 = K_1 \rightarrow K_1.$$

a2.) Az **A4** (egyesítési) szabályból,  $f_1$ -ből és  $f_3$ -ból

$$f_7 = K_1 \rightarrow \{\text{ADÓSZÁM}, \text{LAKHELY}, \text{BEOSZTÁS}\}.$$

a3.) Az **A3** (transzitivitási) szabályból,  $f_4$ -ből és  $f_5$ -ből

$$f_8 = \{\text{BEOSZTÁS}\} \rightarrow \{\text{FIZETÉS}\}.$$

a4.) Az **A4** (egyesítési) szabályból,  $f_4$ -ből és  $f_8$ -ból

$$f_9 = \{\text{BEOSZTÁS}\} \rightarrow \{\text{FIZ\_OSZTÁLY}, \text{FIZETÉS}\}.$$

a5.) Az **A6** (dekompozíciós) szabályból és  $f_3$ -ból

$$f_{10} = K_1 \rightarrow \{\text{BEOSZTÁS}\}.$$

a6.) Az **A3** (transzitivitási), az **A4** (egyesítési) szabályokból,  $f_7$ -ből,  $f_{10}$ -ból és  $f_9$ -ból

$$f_{11} = K_1 \rightarrow \{\text{ADÓSZÁM}, \text{LAKHELY}, \text{BEOSZTÁS}, \text{FIZ\_OSZTÁLY}, \text{FIZETÉS}\}.$$

a7.) Ismét alkalmazva az **A4** (egyesítési) szabályokat, valamint  $f_6$ -ot és  $f_{11}$ -et

$$f_{12} = K_1 \rightarrow A,$$

tehát  $K_1$  szuperkulcs.

a8.) Végül az **A3** (transzitivitási) szabályból,  $f_2$ -ből és  $f_{11}$ -ből adódódik

$$f_{13} = K_2 \rightarrow A,$$

tehát  $K_2$  is szuperkulcs.

b1.) Mivel a  $K_1$  szuperkulcs semelyik részére (jelen esetben egyetlen attribútumára sem) írható fel egyetlen függőség sem, így  $K_1$  kulcs.

b2.) Mivel a  $K_2$  szuperkulcs egyetlen attribútumból áll, így nyilván kulcs.

c.) A  $K_1$  attribútumhalmaz csak a  $K_2$  attribútumhalmaztól függ (az  $f_2$  függőségen keresztül), amelyik pedig csak  $K_1$ -től függ (az  $f_1$  függőség szerint), így e kettőn kívül további kulcsok nem lehetnek a  $\mathbf{K}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  kulcshalmazban.

**Függőségcsalád**

Legyen  $A$  egy attribútumhalmaz, és  $\mathbf{F} = \mathbf{F}\{A\}$  egy ezzel kompatibilis logikai-struktúra. Ekkor az  $\mathbf{F}$  logikai-struktúra *függőségcsaládot* alkot,

**F1.** ha  $\mathbf{F}$  *reflexív*,

azaz minden  $Y \subseteq X \subseteq A$  esetén

$$X \rightarrow Y \in \mathbf{F},$$

**F2.** ha **F** zárt a bővítésre,

azaz minden  $V \subseteq A$  esetén

ha  $X \rightarrow Y \in \mathbf{F}$ , akkor  $X \cup V \rightarrow Y \cup V \in \mathbf{F}$ , és

**F3.** ha **F** tranzitív,

azaz minden  $X, Y, V \subseteq A$  esetén

ha  $X \rightarrow Y \in \mathbf{F}$  és  $Y \rightarrow V \in \mathbf{F}$ , akkor  $X \rightarrow V \in \mathbf{F}$ .

*Megjegyzés*

Mivel a függőségcsaládok **F1-F3** tulajdonságai megfelelnek az **A1-A3** alapvető Armstrong-szabályoknak, így nyilván egy függőségcsalád minden olyan függőséget tartalmaz, mely belőle az Armstrong szabályokkal képezhető. Így kimondható egy fontos állítás:

**Állítás**

A függőségcsaládok burokstruktúrát alkotnak, tehát zártak az Armstrong-bővítésekre.

*Bizonyítás*

(Lásd az iménti megjegyzés utolsó bekezdését!)

*Megjegyzés*

Az előbbi megjegyzés értelmében elegendő csupán az **A1-A3** Armstrong-szabályokat alkalmazni a logikai-struktúrák ekvivalens bővítésére.

**Példa**

Tekintsük a korábbi  $\mathbf{R}(\underline{A})$  relációsémának azt az  $\mathbf{S}(\underline{B})$  alsémáját, melyben

$B = \{ \text{BEOSZTÁS}, \text{FIZ\_OSZTÁLY}, \text{FIZETÉS} \}$ .

Ekkor a korábbi példa függőségei közül csak az alábbiak maradnak meg:

$g_1 = \{ \text{BEOSZTÁS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}$ , (ez volt az  $f_4$ )

$g_2 = \{ \text{FIZ\_OSZTÁLY} \} \rightarrow \{ \text{FIZETÉS} \}$ , (ez volt az  $f_5$ )

ahol az áttekinthetőség miatt új jelöléseket vezettünk be.

*Feladat*

Állítsunk elő Armstrong-bővítésekkel az  $F = \{ g_1, g_2 \}$  logikai-struktúrából egy függőségcsaládot.

*Megoldás*

Az  $F$  logikai-struktúrát az **F3** tranzitivitási követelmény miatt az **A3** Armstrong-szabály alapján ki kell bővíteni a

$g_3 = \{ \text{BEOSZTÁS} \} \rightarrow \{ \text{FIZETÉS} \}$

függőséggel, az **F1** reflexivitási követelmény miatt az **A1** szerint a

$g_4 = \{ \text{BEOSZTÁS} \} \rightarrow \{ \text{BEOSZTÁS} \}$ ,

$g_5 = \{ \text{FIZ\_OSZTÁLY} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}$ , és a

$g_6 = \{ \text{FIZETÉS} \} \rightarrow \{ \text{FIZETÉS} \}$

függőségekkel, az **F2** zártági követelmény miatt az **A2** szerint a

$g_7 = \{ \text{BEOSZTÁS}, \text{FIZ\_OSZTÁLY} \} \rightarrow \{ \text{BEOSZTÁS} \}$ ,

$g_8 = \{ \text{BEOSZTÁS}, \text{FIZETÉS} \} \rightarrow \{ \text{BEOSZTÁS} \}$ ,

$g_9 = \{ \text{FIZ\_OSZTÁLY}, \text{BEOSZTÁS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}$ ,

$g_{10} = \{ \text{FIZ\_OSZTÁLY}, \text{FIZETÉS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}$ ,

$g_{11} = \{ \text{FIZETÉS}, \text{FIZ\_OSZTÁLY} \} \rightarrow \{ \text{FIZETÉS} \}$ ,

$g_{12} = \{ \text{FIZETÉS, BEOSZTÁS} \} \rightarrow \{ \text{FIZETÉS} \}$  , továbbá  
 $g_{13} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{BEOSZTÁS} \}$  ,  
 $g_{14} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}$  ,  
 $g_{15} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{FIZETÉS} \}$  , valamint  
 $g_{16} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY} \}$  ,  
 $g_{17} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{BEOSZTÁS, FIZETÉS} \}$  ,  
 $g_{18} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY, FIZETÉS} \}$  , végül  
 $g_{19} = \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \} \rightarrow \{ \text{BEOSZTÁS, FIZ\_OSZTÁLY, FIZETÉS} \}$

függőségekkel. Az így kapott

$\mathbf{F} = \{ g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}, g_{11}, g_{12}, g_{13}, g_{14}, g_{15}, g_{16}, g_{17}, g_{18}, g_{19} \}$   
 logikai-struktúrát a funkcionális-függőség definíciója szerint már csak a zérusfüggőségekkel kell kiegészíteni ahhoz, hogy függőségcsalád legyen (tehát az olyanokkal, melyekben a "→" jel valamelyik, vagy mindkét oldalán üres halmaz áll).

## Függőség-generátor

Egy  $H \subseteq \mathbf{P}\{A\}$  logikai-struktúrát *függőséggenerátornak* nevezünk, ha létezik olyan  $\mathbf{F}\{A\}$  függőségcsalád, mely  $H$ -ből Armstrong-bővítésekkel előállítható, azaz

$$\text{Gen}(H) = \mathbf{F}\{A\} .$$

### Állítás

1. Minden függőségcsalád egyben függőség-generátor is, és
2. minden  $F$  logikai-struktúrához létezik olyan  $\mathbf{F}\{A\}$  függőségcsalád, melynek  $F$  a függőség-generátora, és ez nem más, mint az  $F$  burokstruktúrája, vagyis az  $\mathbf{F}\{F\}$  logikai-struktúra, tehát  $\text{Gen}(F) = \mathbf{F}\{F\}$ .

### Bizonyítás

A fenti állítások a definíciókból közvetlenül következnek.

### Példa

Tekintsük az iménti  $\mathbf{S}\langle \underline{B} \rangle$  relációsémát. Ekkor az  $F = \{ g_1, g_2 \}$  logikai-struktúra függőség-generátor, hiszen Armstrong-bővítések segítségével előállítható belőle az  $\mathbf{F}$  függőségcsalád.

## Logikai-struktúrák ekvivalenciája

Korábban logikai-struktúrákat ekvivalenseknek neveztünk, ha séma-kompatibilisek, azaz valamely  $H_1$  és  $H_2$  logikai-struktúrák, azaz

$$H_1 \equiv H_2 , \text{ ha } H_1 \# H_2 .$$

Később beláttuk, hogy ezt úgy is mondhatjuk, hogy valamely  $H_1$  és  $H_2$  logikai-struktúrák ekvivalensek, ha burokstruktúrájuk azonos, azaz

$$H_1 \equiv H_2 , \text{ ha } \mathbf{F}\{H_1\} = \mathbf{F}\{H_2\} .$$

Most az Armstrong-bővítésekre vonatkozó megállapítások alapján azt is mondhatjuk, hogy valamely  $H_1$  és  $H_2$  logikai-struktúrák akkor ekvivalensek, ha ugyanazt a függőségcsaládot generálják, azaz

$$H_1 \equiv H_2 , \text{ ha } \text{Gen}(H_1) = \text{Gen}(H_2) .$$

*Megjegyzés*

A logikai-struktúrák ekvivalenciájának jelentőségét már említettük (például az Armstrong-szabályok bevezetésénél). Tudjuk az ekvivalencia fogalma lehetővé teszi azt, hogy egy feladat megoldása szempontjából azonos adatbázist definiáló logikai-struktúrák közül kiválaszthassuk azt, amelyik az adatbázis megvalósításának szempontjából hasznosabb.

Az alábbiakban néhány olyan speciális tulajdonságú logikai-struktúrát mutatunk be, melyeknek kifejezetten a gyakorlati alkalmazások szempontjából van jelentősége.

Későbbiekben, az úgynevezett normalizálásokkal kapcsolatban látni fogjuk, hogy még egyéb követelményeknek is meg kell felelniük a gyakorlati szempontokból legjobbnak tekinthető logikai-struktúrák.

**Irredundáns függőség-generátor**

Legyen  $F = F\{A\}$  egy függőségcsalád. Ekkor egy  $H \subseteq F$  logikai-struktúra az  $F$  függőségcsalád *irredundáns függőség-generátora*, ha

1.  $Gen(H) = F$ ,  
vagyis a  $H$  logikai-struktúra generátora az  $F$  függőségcsaládnak, és
2. minden  $J \subseteq H$  és  $Gen(J) = F$  esetén  $J = H$ ,  
vagyis a  $H$  bármely elemi függőségének elhagyásával már nem generátora  $F$ -nek.

**Minimális függőség-generátor**

Legyen  $F = F\{A\}$  egy függőségcsalád. Ekkor egy  $H \subseteq F$  logikai-struktúra az  $F$  függőségcsalád *minimális függőség-generátora*, ha

1.  $Gen(H) = F$ , és
2. az  $F$  függőségcsaládnak nem létezik nála kisebb elemszámú függőség-generátora.

**A függőség-generátorok néhány tulajdonsága**

Az alábbi állítások bizonyíthatóak, azonban a részletes algebrai bizonyítások helyett inkább mintapéldákat adunk. Mivel ezek úgynevezett egzisztencia-állítások (azaz valaminek a létezését állítják), így a példák „bizonyító erejűek”.

**Állítás** (A minimális és az irredundáns függőség-generátorok kapcsolata)

Egy irredundáns függőség-generátor nem biztos, hogy minimális, de egy minimális függőség-generátor mindig irredundáns.

**Példa**

Tekintsük a korábbi példa  $R\langle A \rangle$  relációsémáját, és a hozzá megadott  $F = \{f_1, f_2, f_3, f_4, f_5\}$  logikai-struktúrát. Ez nyilván egy irredundáns függőség-generátor, hiszen bármelyik elemi függőség elhagyása után már tudjuk az előbbivel ekvivalens függőségcsaládot generálni az  $R\langle A \rangle$  relációsémában.

Hozzuk létre ezután az  $F_3 = \{f_2, f_3^{(1)}, f_5\}$  logikai-struktúrát oly módon, hogy

$f_3^{(1)} = \{NÉV, SZÜL\_DÁTUM\} \rightarrow \{ADÓSZÁM, LAKHELY, BEOSZTÁS\}$

legyen. Ez is irredundáns függőség-generátor, ám belátható, hogy ez még minimális is, hiszen ennél kevesebb függőségből álló ekvivalens logikai-struktúra már biztos nem létezik.

**Állítás** (A minimális és az irredundáns függőség-generátorok száma)

Egy függőscsaládhoz több ekvivalens minimális függőség-generátor, és több ekvivalens irredundáns függőség-generátor is tartozhat.

**Példa**

Tekintsük a korábbi példa  $\mathbf{R}(\underline{A})$  relációsémáját, és a hozzá megadott  $F = \{f_1, f_2, f_3, f_4, f_5\}$  logikai-struktúrát, amelyről az előző példából már tudjuk, hogy irredundáns függőség-generátor. Hozzuk létre az  $F_4 = \{f_1, f_2, f_3^{(2)}, f_4, f_5\}$  logikai-struktúrát oly módon, hogy

$$f_3^{(2)} = \{ \text{ADÓSZÁM} \} \rightarrow \{ \text{LAKHELY, BEOSZTÁS} \}$$

legyen. Ez a logikai-struktúra is irredundáns függőség-generátor, és nyilvánvalóan ekvivalens az eredeti  $F$  függőség-generátorral.

Ezek után hozzuk létre az  $F_5 = \{f_1, f_3^{(3)}, f_4, f_5\}$  logikai-struktúrát oly módon, hogy

$$f_3^{(3)} = \{ \text{ADÓSZÁM} \} \rightarrow \{ \text{NÉV, SZÜL\_DÁTUM, LAKHELY, BEOSZTÁS} \}$$

legyen. Ez a logikai-struktúra is minimális függőség-generátor, és nyilvánvalóan ekvivalens az előző példabeli  $F_3$  függőség-generátorral.

## A relációalgebra és a függőségalgebra kapcsolata

Az eddig bevezetett fogalmak tükrében szemléltethető a relációalgebra és a függőségalgebra „hasonlósága”. A „metafogalmakat” (a fogalmak jellegének megnevezését) intuitív módon használjuk, hiszen a célunk csupán felkelteni a hasonlóság felismerésének érzetét.

Tekintsük először a relációsémát. A relációalgebra legfontosabb fogalmainak *hierarchiája* a következő. Az *alapfogalom* a  $\mathbf{D}(\underline{A})$  Descartes-szorzat, melynek *kiindulási fogalma* az  $\underline{A}$  attribútumvektor, *tartalmazott fogalma* a rekord, *származtatott fogalma* az  $\mathbf{R}(\underline{A})$  relációséma, mely a Descartes-szorzatnak a függőségek által korlátozott rekordjait tartalmazza, és *gyakorlati alapfogalma* az  $R(\underline{A})$  reláció, mely (a rekordismétlésektől eltekintve) a relációséma egy részhalmaza (vagyis a relációséma redukált részhalmaza).

### A RELÁCIÓALGEBRA FOGALOMHIERARCHIÁJA:

a fogalom jellege	a fogalom	jelölése	kapcsolata az alapfogalommal
alapfogalom	Descartes-szorzat	$\mathbf{D}(\underline{A})$	$\mathbf{D}(\underline{A}) = \times \underline{A}$
kiindulási fogalom	Attribútumvektor	$\underline{A}$	A $\mathbf{D}(\underline{A})$ Descartes-szorzat az $\underline{A}$ attribútumvektor teljes érték-tartománya által meghatározott összes vektort tartalmazza
tartalmazott fogalom	Rekord	$\underline{r}$	$\underline{r} \in \mathbf{D}(\underline{A})$
származtatott fogalom	Relációséma	$\mathbf{R}(\underline{A})$	$\mathbf{R}(\underline{A}) \subseteq \mathbf{D}(\underline{A})$ , ahol $\mathbf{R}(\underline{A})$ relációséma a $\mathbf{D}(\underline{A})$ Descartes-szorzat „célszerűen” kiválasztott rekordjait tartalmazza (korlátozás a logikai-struktúra által)
gyakorlati alapfogalom	Reláció	$R(\underline{A})$	$R(\underline{A}) \subseteq^R \mathbf{R}(\underline{A})$ , a reláció a gyakorlatban megvalósított relációséma

Tekintsük ezután a függőségsémát. Az  $\mathbf{F}\{A\}$  függőségsalád az  $A$  attribútumhalmaz részhalmazai fölötti, vagyis a  $2^A$  halmaz fölötti unáris függvények halmazának, vagyis a  $\mathbf{P}\{A\}$  hatványfüggőségnek (a logikai-struktúrában megadott függőségek Armstrong-bővítése által kijelölt) függőségeit tartalmazza, és így minden, a gyakorlatban megadott, az  $A$  attribútumhalmazon értelmezett  $F\{A\}$  függőség-halmaz, mint logikai-struktúra ennek valamelyik részhalmaza. Tehát az *alapfogalom* a hatványfüggőség, *kiindulási fogalom* az attribútumhalmaz, *tartalmazott fogalom* a függőség, és *gyakorlati alapfogalom* a logikai-struktúra, mely természetesen a hatványfüggőség részhalmaza. Legfontosabb *származtatott fogalom* a függőségsalád.

#### ***A FÜGGŐSÉGALGEBRA FOGALOMHIERARCHIÁJA:***

<b>A fogalom jellege</b>	<b>a fogalom</b>	<b>jelölése</b>	<b>kapcsolata az alapfogalommal</b>
alapfogalom	Hatványfüggőség	$\mathbf{P}\{A\}$	$\mathbf{P}\{A\} = 2^A \times 2^A$
kiindulási fogalom	Attribútumhalmaz	$A$	A $\mathbf{P}\{A\}$ hatványfüggőség az $A$ attribútumhalmaz fölötti összes függőséget tartalmazza
tartalmazott fogalom	Függőség	$f$	$f \in \mathbf{P}\{A\}$
származtatott fogalom	Függőségsalád	$\mathbf{F}\{A\}$	$\mathbf{F}\{A\} \subseteq \mathbf{P}\{A\}$ , ahol $\mathbf{F}\{A\}$ függőségsalád a $\mathbf{P}\{A\}$ hatványfüggőség „célszerűen” kiválasztott függőségeit tartalmazza (korlátozás a logikai-struktúra Armstrong bővítése által)
gyakorlati alapfogalom	Logikai-struktúra	$F\{A\}$	$F\{A\} \subseteq \mathbf{F}\{A\}$ , és $Gen(F\{A\}) = \mathbf{F}\{A\}$ . A logikai-struktúra a függőségsalád gyakorlatban használt részhalmaza.

A fenti táblázatokból láthatóan a relációalgebra és a függőségalgebra legfontosabb fogalmi párba állíthatóak a bennük szereplő fogalmak felépítése és használata alapján.

## **Függőségalgebrai műveletek**

Már említettük a függőségalgebra hasonlóságát a relációalgebrához, és az imént e két rendszer fogalmi hierarchiájának hasonlóságát is bemutattuk. Az előző fejezet relációalgebrai műveletei után most megvizsgáljuk a függőségalgebra műveleteit is.

Az előző fejezetben azt mondtuk, hogy akkor beszélhetünk algebrai struktúráról, ha

- adott egy objektumhalmaz,
- adott ezen értelmezett műveletek halmaza, valamint
- e műveletekre nézve az objektumhalmaz zárt, vagyis bármely műveletvégzés eredménye is eleme az objektumhalmaznak.

A függőségalgebrára vonatkozóan ez azt jelenti, hogy a függőségek halmazának zártnak kell lennie a függőségeken definiált műveletekre nézve.

Gyakorlati okokból a függőség-algebrai műveletektől nem csupán azt várjuk el, hogy a műveletek eredményei függőségek legyenek, hanem azt is, hogy ugyanannak a függőség-családnak legyenek elemei. Az alábbiakban definiált műveletek megfelelnek ennek a követelménynek is.

E műveletek az Armstrong-szabályokból vezethetők le. Definiáljuk függőségek egyesítését a bővítési szabály alapján, a függőségek szorzását a tranzitivitási szabály alapján, sőt a dekompozíciós szabály alapján még a függőségek felbontását is.

Az alábbiakban az egyes műveletek definiálása után bemutatjuk a műveletek bevezetésének indoklását is. Az ennek során alkalmazott formális bizonyítások során az áttekinthetőség érdekében nagyfokú tömörségre törekedtünk.

Legyen ezek után  $\mathbf{F}\{A\}$  az  $A$  attribútumhalmazzal kompatibilis függőségek egy családja, és  $X, Y, V, Z \subseteq A$ . Ekkor  $\mathbf{F}\{A\}$ -ban érvényesek az alábbi műveletek, és  $\mathbf{F}\{A\}$  zárt is e műveletekre nézve. A bizonyításokban **A1-A6** módon az azonos jelölésű Armstrong-szabályokra hivatkozunk.

### Függőségek egyesítése

$$X \rightarrow Y \cup V \rightarrow Z \triangleq X \cup V \rightarrow Y \cup Z.$$

#### Állítás

$$X \rightarrow Y, V \rightarrow Z \models X \cup V \rightarrow Y \cup Z,$$

azaz indokolt a függőségek egyesítésének fenti definíciója.

#### Bizonyítás

$$X \rightarrow Y, \mathbf{A2} \models X \cup V \rightarrow Y \cup V \quad (1)$$

$$\mathbf{A1} \models Y \cup V \rightarrow Y \quad (2)$$

$$(1), (2), \mathbf{A3} \models X \cup V \rightarrow Y \quad (3)$$

$$V \rightarrow Z, \mathbf{A2} \models X \cup V \rightarrow X \cup Z \quad (4)$$

$$\mathbf{A1} \models X \cup Z \rightarrow Z \quad (5)$$

$$(4), (5), \mathbf{A3} \models X \cup V \rightarrow Z \quad (6)$$

$$(3), (6), \mathbf{A4} \models X \cup V \rightarrow Y \cup Z \blacksquare \quad (7)$$

### Függőségek szorzata

$$X \rightarrow Y \otimes Y \rightarrow Z \triangleq X \rightarrow Z.$$

#### Állítás

$$X \rightarrow Y, Y \rightarrow Z \models X \rightarrow Z,$$

azaz indokolt a függőségek szorzatának fenti definíciója.

#### Bizonyítás

Az **A3** tranzitivitási szabályból közvetlenül következik. ■

### Függőség felosztása

$$X \rightarrow \Phi[Y] \triangleq \{X \rightarrow Y_i \mid Y_i \in \Phi[Y]\},$$

ahol  $\Phi[Y]$  az  $Y$  halmaz valamely felosztása.

**Állítás**

$$X \rightarrow Y, \Phi[Y] \models X \rightarrow Y_i, Y_i \in \Phi[Y],$$

azaz indokolt a függőség felosztásának fenti definíciója.

**Bizonyítás**

Az **A6** dekompozíciós szabályból közvetlenül következik. ■

## 7. FEJEZET

# A relációs adatbázis tervezése

## Alapfogalmak

### Elsődleges és másodlagos attribútumok

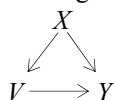
Egy relációséma attribútumai közül azokat, melyek legalább egy kulcsban szerepelnek *elsődleges attribútumoknak*, a többieket *másodlagos attribútumoknak* nevezzük.

## Speciális függőségek

### Teljes és tranzitív függőség

1. Legyen  $R\langle A \rangle$  egy relációséma,  $X, Y \subseteq A$ , és  $X \rightarrow Y$ . Ekkor a  $Y$  *teljesen függ*  $X$ -től az  $R\langle A \rangle$ -ban, ha minden  $X' \subset X$  esetén  $X' \nrightarrow Y$ .
2. Legyen  $R\langle A \rangle$  egy relációséma,  $X, Y \subseteq A$ , és  $X \rightarrow Y$ . Ekkor az  $Y$  *tranzitívan függ*  $X$ -től, ha létezik olyan  $V \subseteq A$ , melyre
  - 2.1.  $X \rightarrow V$  és  $V \rightarrow Y$ , miközben
  - 2.2.  $V \nrightarrow X$  és  $Y \nrightarrow V$ .

A fentieket grafikusán jelölve:



#### Megjegyzés

Nyilván egy relációséma attribútumhalmaza teljes függésben van a kulcstól.

#### Példa

Tekintsük a „Kulcs meghatározása az Armstrong-szabályok segítségével” pontbeli példa  $R\langle A \rangle$  relációsémáját. Az ott bevezetett

$$f_1 = \{ \text{NÉV, SZÜL\_DÁTUM} \} \rightarrow \{ \text{ADÓSZÁM} \}$$

függőség nyilván egy teljes függőség.

**Példa**

Tekintsük az imént említett példa  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémáját, valamint az ott bevezetett alábbi

$$\begin{aligned} f_2 &= \{ \text{ADÓSZÁM} \} \rightarrow \{ \text{NÉV, SZÜL\_DÁTUM} \}, \\ f_3 &= \{ \text{NÉV, SZÜL\_DÁTUM} \} \rightarrow \{ \text{LAKHELY, BEOSZTÁS} \}, \\ f_4 &= \{ \text{BEOSZTÁS} \} \rightarrow \{ \text{FIZ\_OSZTÁLY} \}, \\ f_5 &= \{ \text{FIZ\_OSZTÁLY} \} \rightarrow \{ \text{FIZETÉS} \}, \\ f_8 &= \{ \text{BEOSZTÁS} \} \rightarrow \{ \text{FIZETÉS} \} \end{aligned}$$

függőségeket, valamint az ezekből „A függőség-generátorok néhány tulajdonsága” pontban származtatott

$$f_3^{(2)} = \{ \text{ADÓSZÁM} \} \rightarrow \{ \text{LAKHELY, BEOSZTÁS} \}$$

függőséget.

Könnyen belátható, hogy az  $f_8$  és az  $f_3^{(2)}$  függőségek tranzitívak, mégpedig az  $f_8$  a  $\{ \text{FIZ\_OSZTÁLY} \}$  attribútumhalmazon és az  $f_4, f_5$  függőségeken keresztül, az  $f_3^{(2)}$  pedig a  $\{ \text{NÉV, SZÜL\_DÁTUM} \}$  attribútumhalmazon és az  $f_2, f_3$  függőségeken keresztül.

**Belső-függőség**

Legyen  $\mathbf{R}\langle\mathbf{A}\rangle$  egy relációséma,  $\mathbf{K}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  a kulcsok halmaza, továbbá  $X, Y \subseteq \mathbf{A}$ . Ekkor az  $X \rightarrow Y$  funkcionális-függőséget *belső-függőségnek* nevezzük, ha minden  $K \in \mathbf{K}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  esetén  $X \cap K = \emptyset$ . A belső függőség tehát a nem-kulcs (azaz másodlagos) attribútumoktól való függést jelenti.

**Állítás** (A tranzitivitási kényszer szabálya)

Egy  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémában legyen  $\mathbf{K}\{\mathbf{R}\langle\mathbf{A}\rangle\}$  a kulcsok halmaza, továbbá  $K, X, Y \subseteq \mathbf{A}$ ,  $K \in \mathbf{K}\{\mathbf{R}\langle\mathbf{A}\rangle\}$ ,  $X \rightarrow Y$  és  $K \cap X = \emptyset$ . Ekkor az  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémában van tranzitív függés, mégpedig  $K \rightarrow X \rightarrow Y$  alakban.

*Bizonyítás*

Mivel  $K$  egy kulcs, így  $K \rightarrow \mathbf{A}$ . A dekompozíciós szabály (A6) miatt ekkor  $K \rightarrow X$  és  $K \rightarrow Y$  is teljesül. ■

A fenti állítás tehát azt mondja ki, hogy ha egy relációsémában van belső függés, akkor ott tranzitív függés is van.

**Relációséma dekompozíciója attribútumvektor-halmaz szerint**

A relációalgebrai műveletek között bevezettük a relációk dekompozícióját, mely relációsémára (mint speciális relációra) nyilván ugyanígy értelmezhető:

Legyen  $A$  egy attribútumhalmaz, és  $B = \{\underline{B}_1, \underline{B}_2, \dots, \underline{B}_n\}$  egy attribútumvektor-halmaz, ahol minden  $\underline{B}_i \in B$  esetén  $B_i \subseteq A$  (tehát bármely  $\underline{B}_i, \underline{B}_j \in B$  esetén  $B_i \cap B_j \neq \emptyset$  megengedett), és

$$\bigcup_{\underline{B}_i \in B} B_i = A.$$

Ekkor egy  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémának a  $B$  attribútumvektor-halmazra vonatkozó dekompozíciója

$$\mathbf{R}\langle\mathbf{A}\rangle/B \triangleq \{ \mathbf{R}_i \mid \mathbf{R}_i = \mathbf{R}\langle\mathbf{A}\rangle|_{\underline{B}_i}, \underline{B}_i \in B \},$$

ahol természetesen minden  $\underline{B}_i \in B$  esetén  $B_i$  a  $\underline{B}_i$  vektor alaphalmaza. A továbbiakban egy  $\mathbf{R}\langle\mathbf{A}\rangle$  relációséma valamely dekompozícióját

$$\varphi[\mathbf{R}\langle\mathbf{A}\rangle]$$

módon is fogjuk jelölni.

Ha  $\mathbf{R}\langle\mathbf{A}\rangle \in \varphi[\mathbf{R}\langle\mathbf{A}\rangle]$ , akkor a  $\varphi[\mathbf{R}\langle\mathbf{A}\rangle]$ -t *triviális dekompozíciónak* nevezzük.

#### Megjegyzés

Vegyük észre, hogy a definíció szerint a dekompozícióban szereplő relációsémák alaphalmazainak lehetnek közös elemei.

Relációséma dekompozíciója és az előző két fejezetben tárgyalt sémafelosztás (osztályozás) között az a különbség, hogy míg a sémafelosztással egy relációsémát rekordok halmazára (mégpedig diszjunkt halmazára), addig a dekompozícióval egy relációsémát *alsémáinak halmazára* (mégpedig esetleg közös attribútumot is tartalmazó alsémáinak halmazára) bontunk fel. Szemléletesen azt lehetne mondani, hogy a sémafelosztással „vízszintesen”, míg a dekompozícióval „függőlegesen” bontjuk fel a relációsémát (pontosabban a tábláját).

### Relációséma dekompozíciója funkcionális függőség szerint

Legyen  $\mathbf{R}\langle\mathbf{A}\rangle$  egy relációséma, és  $f_{X,Y} = X \rightarrow Y$  egy függőség, ahol  $X, Y \subseteq A$ , és  $X \cap Y = \emptyset$ . Ekkor valamely  $\mathbf{R}\langle\mathbf{A}\rangle$  relációsémának az  $f_{X,Y}$  függőségre vonatkozó dekompozíciója

$$\mathbf{R}\langle\mathbf{A}\rangle / f_{X,Y} \triangleq \{ \mathbf{R}\langle\mathbf{A}\rangle|_S, \mathbf{R}\langle\mathbf{A}\rangle|_T \},$$

ahol

$$S = A \setminus Y, \text{ és } T = X \cup Y,$$

továbbá az  $S$  illetve  $T$  vektorok az  $S$  illetve  $T$  halmazok tetszőleges vektorai.

#### Megjegyzés

Mivel  $S \cup T = (A \setminus Y) \cup (X \cup Y) = X \cup A \setminus Y \cup Y = X \cup A = A$ , így a függőség szerinti dekompozíció tulajdonképpen egy speciális attribútumvektor-halmaz szerinti dekompozíció.

Ha  $T = A$ , akkor a  $\varphi[\mathbf{R}\langle\mathbf{A}\rangle] = \mathbf{R}\langle\mathbf{A}\rangle / f_{X,Y}$  nyilván egy triviális dekompozíció.

### Relációséma veszteségmentes dekompozíciója

Egy relációséma valamely  $X \rightarrow Y$  funkcionális függősége szerint való dekompozíciója tehát azt jelenti, hogy helyettesítjük két (egymáshoz az  $X$  attribútumhalmaz révén kapcsolódó) alsémájával. E helyettesítés célja, hogy a származtatott relációsémák valamilyen szempontból kellemesebb használatot biztosítsanak a gyakorlatban, mint az eredeti relációséma (például elkerülhető legyen az úgynevezett törlési anomália, lásd később).

Az alapvető kérdés természetesen az, hogy egy relációséma dekompozíciója során nem veszítünk-e el információt. Vajon a származtatott relációsémák hordozzák-e ugyanazokat az információkat, kapcsolatokat, mint az eredeti? Más szóval: vajon veszteségmentes-e egy relációséma valamely dekompozíciója. Végül pedig a legfontosabb kérdés: miként tudunk meggyőződni arról, hogy egy dekompozíció veszteségmentes-e.

Az alábbiakban ellenőrzésként azt vizsgáljuk, hogy a származtatott relációsémák természetes összekapcsolása visszaadja-e az eredeti relációsémát.

Ezekután tehát egy  $\mathbf{R}\langle\mathbf{A}\rangle$  relációséma valamely  $\varphi[\mathbf{R}\langle\mathbf{A}\rangle]$  dekompozícióját *veszteségmentesnek* nevezzük, ha

$$\mathbf{R}\langle \underline{B}_1 \rangle \bowtie \dots \bowtie \mathbf{R}\langle \underline{B}_n \rangle = \mathbf{R}\langle \underline{A} \rangle,$$

ahol minden  $\mathbf{R}\langle \underline{B}_i \rangle \in \varphi[\mathbf{R}\langle \underline{A} \rangle]$ .

### Példa

Az alábbiakban bemutatjuk ugyanannak a relációsémának két dekompozícióját.

Legyen az alábbi  $\mathbf{R}\langle \underline{A} \rangle$  relációséma az  $\underline{A} = \langle A, B, C \rangle$  attribútumvektoron értelmezve, és legyen  $f_{\{C\},\{A\}} = \{C\} \rightarrow \{A\}$  a funkcionális függősége. Hasonlítsuk össze a  $\varphi_1[\mathbf{R}\langle \underline{A} \rangle] = \{ \mathbf{R}'\langle A, B \rangle, \mathbf{R}'\langle B, C \rangle \}$ , valamint a  $\varphi_2[\mathbf{R}\langle \underline{A} \rangle] = \mathbf{R}\langle \underline{A} \rangle / f_{\{C\},\{A\}}$  dekompozíciókat a veszteségmentesség szempontjából, ahol  $\mathbf{R}'\langle A, B \rangle = \mathbf{R}\langle \underline{A} \rangle \llcorner_{\langle A, B \rangle}$ , és  $\mathbf{R}'\langle B, C \rangle = \mathbf{R}\langle \underline{A} \rangle \llcorner_{\langle B, C \rangle}$ .

Először is állapítsuk meg, hogy az  $\mathbf{R}\langle \underline{A} \rangle$  relációséma  $f_{\{C\},\{A\}}$  függőség szerinti dekompozíciója  $\varphi_2[\mathbf{R}\langle \underline{A} \rangle] = \{ \mathbf{R}''\langle B, C \rangle, \mathbf{R}''\langle A, C \rangle \}$ , ahol például  $\mathbf{R}''\langle B, C \rangle = \mathbf{R}\langle \underline{A} \rangle \llcorner_{\langle B, C \rangle}$ .

$\mathbf{R}\langle \underline{A} \rangle: \underline{ABC}$		$\mathbf{R}'\langle A, B \rangle: \underline{AB}$	$\mathbf{R}'\langle B, C \rangle: \underline{BC}$	$\mathbf{R}^{(1)}: \underline{ABC}$
1 3 5		1 3	3 5	1 3 5
1 4 6	$\varphi_1[\mathbf{R}\langle \underline{A} \rangle]$	1 4	4 6	1 3 7
2 3 7	$\Rightarrow$	2 3	3 7	1 4 6
2 4 8		2 4	4 8	1 4 8
				2 3 5
				2 3 7
				2 4 6
				2 4 8

$\mathbf{R}\langle \underline{A} \rangle: \underline{ABC}$		$\mathbf{R}''\langle B, C \rangle: \underline{BC}$	$\mathbf{R}''\langle A, C \rangle: \underline{AC}$	$\mathbf{R}^{(2)}: \underline{ABC}$
1 3 5		3 5	1 5	1 3 5
1 4 6	$\varphi_2[\mathbf{R}\langle \underline{A} \rangle]$	4 6	1 6	1 4 6
2 3 7	$\Rightarrow$	3 7	2 7	2 3 7
2 4 8		4 8	2 8	2 4 8

Az ellenőrzések eredményei:

$$\mathbf{R}^{(1)} = \mathbf{R}'\langle A, B \rangle \bowtie \mathbf{R}'\langle B, C \rangle,$$

$$\mathbf{R}^{(2)} = \mathbf{R}''\langle B, C \rangle \bowtie \mathbf{R}''\langle A, C \rangle.$$

Láthatóan  $\mathbf{R}^{(1)} \neq \mathbf{R}\langle \underline{A} \rangle$ , míg  $\mathbf{R}^{(2)} = \mathbf{R}\langle \underline{A} \rangle$ , tehát a funkcionális függőség szerinti  $\varphi_2[\mathbf{R}\langle \underline{A} \rangle] = \mathbf{R}\langle \underline{A} \rangle / f_{\{C\},\{A\}}$  dekompozíció veszteségmentes. Ennek alapján (bizonyítás nélkül) kimondjuk az alábbi állítást.

### Állítás

Egy relációsémának a vele kompatibilis funkcionális függőségek szerinti dekompozíciói veszteségmentesek.

### Megjegyzés

A fenti állítás jelentőségét az adja, hogy a relációsémák konkrét tartalmától függetlenül lehetővé teszi egy dekompozíció veszteségmentességének (és így a később bemutatásra kerülő normalizálási folyamat egyes lépéseinek) ellenőrzését.

Nyilván a gyakorlat számára érdektelenek a triviális dekompozíciók. Ám attól, hogy egy relációséma valamely függőség szerint csak triviális módon dekomponálható, még lehetséges, hogy létezik olyan dekompozíciója, mely veszteségmentes.

## Relációséma függőségőrző dekompozíciója

Figyeljünk fel arra, hogy az előző példában az  $\mathbf{R}\langle \underline{A} \rangle$  relációséma  $\varphi_1[\mathbf{R}\langle \underline{A} \rangle]$  dekompozíciójából nem csupán az eredeti  $\mathbf{R}\langle \underline{A} \rangle$  relációséma nem volt visszaállítható (éppen ezért neveztük veszteségesnek e felbontást), hanem a felbontás során „elveszítettük” az  $f_{\{C\},\{A\}}$  funkcionális függőséget is. Vegyük észre, hogy ez nem abból következik, hogy a „visszaállított”  $\mathbf{R}^{(1)}$  relációsémában nem teljesül az  $f_{\{C\},\{A\}}$  (akár teljesülhetne is), hanem abból, hogy sem az  $\mathbf{R}'\langle A, B \rangle$ , sem az  $\mathbf{R}'\langle B, C \rangle$  relációsémákon nem értelmezhető az  $f_{\{C\},\{A\}}$  függőség. Miért fontos ez?

Tegyük fel, hogy valamilyen okból az  $\mathbf{R}\langle \underline{A} \rangle$  relációsémát szeretnénk kiegészíteni az  $\langle 1, 3, 7 \rangle$  rekorddal. Megtehetjük? Nem, mivel az  $f_{\{C\},\{A\}}$  függőség ezt megakadályozza. Mint mondtuk a függőségek fejezik ki ismereteinket a világról, tehát nyilván elvárjuk, hogy egy dekompozíció ezeket megőrizze. Mi a helyzet tehát az  $\mathbf{R}'\langle A, B \rangle$  és az  $\mathbf{R}'\langle B, C \rangle$  relációsémákkal? Ki tudjuk-e bővíteni ezeket az  $\langle 1, 3, 7 \rangle$  rekorddal, pontosabban annak e relációsémákra értelmezhető vetületeivel, vagyis az  $\langle 1, 3, 7 \rangle \ll_{\langle A, B \rangle} = \langle 1, 3 \rangle$  és az  $\langle 1, 3, 7 \rangle \ll_{\langle B, C \rangle} = \langle 3, 7 \rangle$  rekordokkal? Nos, a válasz nyilván az, hogy igen, azaz lehetséges hamis rekordok bevitele az alsémákba! Mi is akadályozná ezt meg, hiszen az  $f_{\{C\},\{A\}}$  függőség nem értelmezhető az  $\mathbf{R}'\langle A, B \rangle$  és az  $\mathbf{R}'\langle B, C \rangle$  relációsémákon, és nyilván nem is tudjuk e függőséget sem egy, sem több olyan függőséggel helyettesíteni, mely az általa kifejezett *megszorítást* érvényesíteni tudná.

Érdekes kérdés, hogy miért „veszett el” az  $f_{\{C\},\{A\}}$  függőség. Láthatóan a dekompozícióbeli alsémák egyikébe került a függőség magattribútuma, a másikba pedig a képattribútuma, vagyis a dekompozíció „megtörte” a függőséget.

Végül is tehát megállapíthatjuk, hogy a  $\varphi_1[\mathbf{R}\langle \underline{A} \rangle]$  dekompozíció eredménye két olyan alséma ( $\mathbf{R}'\langle A, B \rangle$  és  $\mathbf{R}'\langle B, C \rangle$ ), melyekből az eredeti relációséma nem állítható helyre, és az eredeti relációsémára érvényes függőséget sem őrizték meg. E dekompozícióra tehát kimondható, hogy nem veszteségmentes, és nem őrizte meg az  $f_{\{C\},\{A\}}$  függőséget.

Ha azonban megvizsgáljuk a  $\varphi_2[\mathbf{R}\langle \underline{A} \rangle]$  dekompozíciót (melyről tudjuk, hogy az  $\mathbf{R}\langle \underline{A} \rangle$  relációséma  $f_{\{C\},\{A\}}$  függőség szerinti dekompozíciója), akkor azt láthatjuk, hogy az nem csupán veszteségmentes, de az  $f_{\{C\},\{A\}}$  függőséget is megőrizte. Ezekután fogalmazzuk meg pontosabban e függőségőrzést.

Egy  $\mathbf{R}\langle \underline{A} \rangle$  relációséma valamely  $\varphi[\mathbf{R}\langle \underline{A} \rangle]$  dekompozícióját *függőségőrzőnek* nevezzük, ha az  $\mathbf{R}\langle \underline{A} \rangle$  relációsémára értelmezett összes függőség (minden  $f \in F\{\mathbf{R}\langle \underline{A} \rangle\}$  függőség) legalább egy  $\mathbf{R}\langle \underline{B_i} \rangle \in \varphi[\mathbf{R}\langle \underline{A} \rangle]$  alsémára teljesül.

Az alábbi állítások azt mondják ki, hogy mitől függ egy dekompozíció függőségőrző képessége.

### Állítás

1. Egy relációsémának valamely vele kompatibilis funkcionális függőség szerinti dekompozíciója *e funkcionális függőséget megőrzi*.
2. Ha egy relációsémának valamely függőségre vonatkozóan csak triviális dekompozíciója létezik, akkor *nincs* olyan nem-triviális dekompozíciója, mely e függőséget megőrizné.

### Megjegyzés

Ha tehát egy relációséma valamely dekompozíciójának veszteségmentessége és függőségőrző képessége között keresünk kapcsolatot, akkor általánosságban az alábbiakat mondhatjuk. Egy dekompozíció

- lehet veszteségmentes és függőségörző,
- lehet veszteségmentes és nem függőségörző,
- lehet veszteséges és függőségörző, végül
- lehet veszteséges és nem függőségörző.

A fenti állítások alapján azonban láthatóan van lehetőség veszteségmentes és függőségörző dekompozícióra, ha relációsémánkat függőség szerint dekomponáljuk.

A kimondott állításokat bizonyítás helyett egy példán szemléltetjük.

### Példa

A fenti példában a  $\varphi_1[\mathbf{R}(\underline{A})]$  dekompozíció, mint láttuk, veszteséges, és nem örzi meg az  $f_{\{C\},\{A\}}$  függőséget, a  $\varphi_2[\mathbf{R}(\underline{A})]$  dekompozíció pedig veszteségmentes és meg is örzi az  $f_{\{C\},\{A\}}$  függőséget. Nézzünk egy olyan esetet, ahol a dekompozíció veszteségmentes, ám mégse függőségörző.

Tekintsük a fenti  $\mathbf{R}(\underline{A})$  relációsémára vonatkozóan az  $f_{\{A,B\},\{C\}} = \{A, B\} \rightarrow \{C\}$  funkcionális függőséget. (Ez láthatóan fennáll az  $\mathbf{R}(\underline{A})$  relációsémára – azaz kompatibilis vele –, tehát feltehetjük, hogy valamilyen valódi ismeret kapcsolódik hozzá.) Mivel  $\varphi_3[\mathbf{R}(\underline{A})] = \mathbf{R}(\underline{A})/f_{\{A,B\},\{C\}} = \{ \mathbf{R}(\underline{A})|_{\langle A,B \rangle}, \mathbf{R}(\underline{A})|_{\langle A,B,C \rangle} \}$ , azaz  $\mathbf{R}(\underline{A}) \in \varphi_3[\mathbf{R}(\underline{A})]$ , így  $\varphi_3[\mathbf{R}(\underline{A})]$  egy triviális dekompozíció. Az előbbi esethez hasonlóan a  $f_{\{A,B\},\{C\}}$  függőséget sem tudjuk más függőséggel/függőségekkel egyenértékű módon helyettesíteni, a fenti állítás alapján tehát azt várjuk, hogy  $\mathbf{R}(\underline{A})$ -nak e függőségre vonatkozóan nem lesz függőségörző dekompozíciója. Végig vizsgálhatnánk a lehetséges dekompozíciókat, ám most elégedjünk meg a fenti, a veszteségmentesség szempontjából sikeres  $\varphi_2[\mathbf{R}(\underline{A})]$  dekompozíció ellenőrzésével.

Kíséréljük bevinni például az  $\langle 1, 4, 5 \rangle$  rekordot az  $\mathbf{R}(\underline{A})$  relációsémába. Láthatóan nem engedi az  $f_{\{A,C\},\{B\}}$  függőség! Nézzük ezután, hogy a  $\varphi_2[\mathbf{R}(\underline{A})]$  dekompozícióbeli alsémák bővíthetők-e ennek a rekordnak a megfelelő vetületeivel, vagyis az  $\langle 1, 4, 5 \rangle|_{\langle B,C \rangle} = \langle 4, 5 \rangle$  és az  $\langle 1, 4, 5 \rangle|_{\langle A,C \rangle} = \langle 1, 5 \rangle$  rekordokkal? Bizony bővíthetők, vagyis az  $f_{\{A,C\},\{B\}}$  függőségre vonatkozóan a  $\varphi_2[\mathbf{R}(\underline{A})]$  dekompozícióba is lehetséges hamis rekordot bevinni, tehát e függőségre nézve ez a (veszteségmentes) dekompozíció nem függőségörző.

## Normalizálás

A relációs adatmodellből felépülő adatbázisok belső logikai szerkezetét, valamint a függőségek és a relációséma kapcsolatát a gyakorlatban azért vizsgáljuk, hogy ilymódon optimális adatbázismoddelt tervezhessünk. Nagy kérdés persze, hogy mire is szeretnénk optimalizálni.

Általában *minimális redundanciára* optimalizálunk. Ennek nem az a fő célja, hogy kisebb legyen az adatbázis helyigénye (valamikor azért ez is igen fontos szempont volt), hanem az, hogy elkerülhessük a redundanciából származó különböző rendellenes működéseket, melyeket összefoglalóan csak *adatbázis-anomáliáknak* nevezünk. A redundancia és a függőség kapcsolatát a függőség tulajdonságainál már megemlített „három adatból a negyediket” jósolhatóság szemlélteti.

A minimális redundancia érdekében olyan relációséma-transzformációkat kívánunk végrehajtani, melyek révén egy redundáns relációsémából több, kevésbé redundáns, de összességében az eredetivel logikailag egyező relációséma keletkezik. E felbontással szemben – a korábbiak alapján – elvárjuk, hogy

1. a származtatott relációsémák együttesen kevésbé legyenek redundánsak, mint az eredeti (azaz kevesebb adatot tartsanak),
2. az egyes származtatott relációsémák kisebb fokszerűek legyenek (azaz kevesebb attribútumot tartsanak),
3. a származtatott relációsémákból veszteségmentesen visszaállítható legyen az eredeti relációséma, továbbá
4. az eredeti relációsémára megfogalmazott függőségek mindegyike teljesüljön a származtatott relációsémák legalább egyikére (természetesen külön-külön).

A fenti követelmények azt a természetes igényt fejezik ki, hogy a redundancia-csökkenésével együttjáró előnyök (kisebb memória-igény, anomáliák elkerülése) ne torzítsák el az adatbázist. Ám mindezen követelmények teljesülése esetén is szembe kell néznünk azzal a ténnyel, hogy az eredeti relációséma felbontásával létrejövő több relációséma időigényesebbé teszi az adatbázis-lekérdezéseket.

A másik optimalizációs szempont a *minél gyorsabb működés* biztosítása. Tekintettel arra, hogy egy adatbáziskezelő-rendszer időkritikus működése a lekérdezés (hiszen az adatfeltöltés munkája nyilván tervezhetőbb), így a gyors működés követelményét hatékony lekérdezést biztosító adatbázismodellel tudjuk megvalósítani. Ám könnyen belátható, hogy a lekérdezés hatékonysága romlik, ha több relációsémát kell kezelnie a rendszernek. Tehát a gyors lekérdezés és az irredundáns tárolás egymással ellentétes követelmények.

Amikor egy adatbázismodelt megtervezünk, figyelembe kell tehát venni azt is, hogy milyen célra készül. Egy repülőtéren helyfoglalási rendszert, melyben az adatok állandóan változnak, az adatbázis-anomáliák elkerülése érdekében célszerű minimális redundanciára optimalizálni, míg egy vállalati döntéstámogató rendszert, ahol a tárolt adatok a lekérdezésekhez képest elég stabilnak tekinthetők, viszont változatos és bonyolult elemző-lekérdezésekre lehet számítani, célszerű minél kevesebb relációsémából (adattáblából) felépíteni.

Az alábbiakban vázlatosan bemutatunk egy olyan relációséma transzformációs módszert, melynek segítségével redundanciamentessé, a korábban már említett adatbázis-anomáliáktól mentessé tehetjük az adatbázisunkat. E tevékenységet *normalizálásnak* nevezzük, és a relációs adatmodellel rendelkező adatbázisok hagyományos tervezésének ez a legjellemzőbb fázisa. *Teljes mértékben normalizálnak* nevezzük egy adatbázist, ha annak bármely (a funkcionális függőségei által megengedett) feltöltöttsége esetén sincs egyetlen adattáblán belül lehetőség a „három adatból a negyediket” típusú jósolhatóságra. Ezt több lépésben érhetjük el. A normalizálás egyes lépéseit első, második, stb. *normálformára hozásnak* nevezzük. Nem mindig cél azonban a teljes normalizálás, sőt vannak esetek, amikor szándékosan redundanciát viszünk vissza a modellbe (azaz denormalizálunk), mivel a „túlnormalizálás” esetleg nagyon lerontja hatékonyságot, túl merevvé teszi az adatbázist, stb.

Végül megjegyezzük, hogy az adatbázis-tervezés legfontosabb eleme, a függőségek kijelölése alapvetően intuitív tevékenység. Nem is lehet ez másként, hiszen ismereteinket a világról a függőségek fejezik ki. A kulcsok meghatározása, és maga a teljes normalizálási folyamat a függőségeken alapszik. Ha a függőségeket rosszul írtuk fel, akkor a még oly szakszerű normalizálás is hibás adatbázist eredményez.

Mint mondtuk bemutatásunk vázlatos lesz. Ennek oka alapvetően az, hogy könyvünk célja nem az adatbázis-tervezés, hanem elsősorban az elméleti alapok, valamint az adatbázis-használat bemutatása, másrészt pedig az, hogy e témának igen kiterjedt szakirodalma van ma már magyar nyelven is (lásd az irodalomjegyzéket).

## Normálformák

### Az ősmodell (alapmodell)

*Ősmodellnek* nevezzük azt az adatbázismodellt, melyet az adatgyűjtéskor építünk fel, és az ennek során felmerült összes attribútumot tartalmazza.

#### Megjegyzés

Amikor létre akarunk hozni egy adatbázist, akkor az első lépés az adatgyűjtés. Bár ez a tevékenység is célratörő (például csak olyan személyektől kérünk tájékoztatást – úgynevezett interjút –, akik nyilvánvalóan tájékozottak a munkafolyamatokról, az adatok beszerzéséről, feldolgozásáról és felhasználásáról), azonban ekkor még sokmindent feljegyzünk, beépítünk a modellbe (biztos, ami biztos alapon), amit az elemző átgondolás után már kihagyunk. Persze az is előfordulhat, hogy amikor az elkészült adatbázismodellt egyeztetjük a megrendelővel, akkor ő még módosítást, kiegészítést kér, ami miatt esetleg a tervezés egyes fázisait újra kell kezdenünk.

Emlékeztetőül: *adatbázismodellnek* nevezzük relációsémák véges halmazát. (Az, hogy mely relációsémák alkotják az adatbázismodellt, éppen a tervezési folyamat során dől el.)

#### Példa

Az alábbiakban egy mintapéldán keresztül fogjuk a tervezési folyamatot szemléltetni. Példánkban egy katalógus alapján építünk fel egy adatbázismodellt azzal a céllal, hogy egy könyv kiválasztásához és megrendeléséhez szükséges lekérdezések hatékonyak legyenek.

Az ilyen típusú adatbázisok jellemzője tehát a tartalom alapvető stabilitása, valamint a gyors lekérdezés biztosítása. Ennek érdekében egyrészt elkerüljük a „túlnormalizálást”, másrészt általában lehetőséget adunk a hiányos (esetenként a hibás) információk alapján való keresésre is. Ennek a legegyszerűbb *logikai módja* az, ha az összetett adatokat részeikre bontva külön tároljuk (például több szerző esetén minden szerzőt külön adatként), *fizikai módja* pedig a részszöveg (alstring) szerint való keresés (például megengedjük, hogy a szerzőket a nevüknek csak egy részlete alapján keressük). Mi a továbbiakban csak a logikai módszerekkel fogunk foglalkozni.

Lássuk tehát a katalógus egy részletét!

...

#### ALGORITMUS-ELMÉLET

**Műszaki Könyvkiadó** (1033 Budapest, Szentendrei út 89/93.)

*Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullmann*: Számítógép-algoritmusok tervezése és analízise. (ISBN: 963 10 4323 1, megjelent: 1982, 487 oldal, 117 Ft)

...

#### ADATBÁZIS-KEZELÉS

**IDG Kiadó** (1012 Budapest, Márvány u. 17.)

*Halassy Béla*: Az adatbázis-tervezés alapjai és titkai. (ISBN: 963 8287 012, megjelent: 1994, 379 oldal, 1280 Ft)

...

...

**Kossuth Kiadó** (1043 Budapest, Csányi L. u. 36.)*Dave Ensor, Ian Stevenson*: Oracle-tervezés. (ISBN: 963 09 4122 8, megjelent: 2000, 525 oldal, 3600 Ft)

...

**LSI Oktatóközpont** (1037 Budapest, Bécsi út 324.)*Szelezsán János*: Adatbázisok. (ISBN: 963 577 189 4, megjelent: 1997, 218 oldal, 941 Ft)

...

**Panem Kiadó** (1147 Budapest, Öv u. 146.)*Jeffrey D. Ullmann, Jennifer Widom*: Adatbázisrendszerek alapvetés. (ISBN: 963 545 190 3, megjelent: 1998, 507 oldal, 3490 Ft)*Hector Garcia-Molina, Jeffrey D. Ullmann, Jennifer Widom*: Adatbázisrendszerek megvalósítása. (ISBN: 963 545 280 2, megjelent: 2001, 682 oldal, 5990 Ft)

...

Írjuk fel a fenti katalógusrészlet alapján a **KATALÓGUS** relációsémát:

**KATALÓGUS**(ISBN, KÖNYV\_CÍME, **SZERZŐ**(ISBN, NÉV), TÉMA, KIADÁSI\_ÉV,  
OLDALSZÁM, ÁR, KIADÓ, KIADÓ\_CÍME)

Láthatóan kissé átrendeztük az attribútumokat. Nyilván az ISBN számot előre vesszük, hiszen ez egyértelmű azonosítója, kulcsa minden könyvnek. Ezt kihangsúlyozandó alá is húztuk (amint ezt a továbbiakban is megteesszük a kulcsattribútumokkal a relációsémákban).

Figyeljünk fel arra, miként fejeztük ki a **KATALÓGUS** relációsémában azt a tényt, hogy egy könyvnek több szerzője is lehet. Nyilvánvaló, hogy itt egy többértékű attribútumról van szó, amelynek az elemei tehát nem egyszerűek, hanem összetettek.

Korábban már utaltunk arra, hogy egy elem egyszerű, vagy összetett jellege, mindig nézőpont kérdése. Az a megállapítás, hogy egy halmaznak az elemei egyszerűek, a gyakorlatban mindössze annyit jelent, hogy nem érdekelnek bennünket annak részletei.

A példánkban a **SZERZŐ** attribútumot oly módon valósíthatjuk meg egyszerű elemként, hogy az összes szerző nevét (úgy, ahogy az a katalógusban szerepel) egyetlen adatként tároljuk (egyetlen karaktersorozatként). Nem olyan nagy szentségtörés ez, mint első hallásra gondolnánk, hiszen ha meg szeretnénk valósítani a hiányos, vagy hibás információ alapján való keresést, akkor ez a legjobb megoldás. Gondoljunk csak arra, hogy az *Ullmann* nevű szerző *Jeffrey D. Ullmann* szerint való keresése nem túlzottan felhasználóbarát megoldás (ráadásul még egy angol anyanyelvű olvasónak is problémát jelenthet a dupla betűk megfelelő alkalmazása). Amint azonban már említettük nem célunk a fizikai megoldások taglalása, annál inkább a logikai megoldások bemutatása.

A „több szerző” probléma megoldásaként tehát az „egy szerző – egy adat” módszert választottuk, vagyis a **SZERZŐ** attribútumot egy **SZERZŐ**(ISBN, NÉV) *beágyazott relációsémával* helyettesítettük. Adott könyvhöz tartozó szerzők egyértelmű azonosíthatósága érdekében, a szerzők neve elé minden rekordba elhelyeztük a könyv ISBN számát is (mely természetesen e sémában csak egy szerző nevével együtt alkot kulcsot).

## Nulladik normálforma, 0NF

Egy  $R\langle A \rangle$  relációséma *nulladik normálformájú*, ha *lehetnek* benne összetett attribútumok. (Egy attribútum összetett, ha nem egyszerűek az elemei.) A *nulladik normálformában szereplő attribútumok már a végleges adatbázismodell attribútumai*. A nulladik normálformát 0NF módon is jelöljük.

A szakirodalomban az összetett attribútumokat *többsértékű attribútumoknak*, vagy *ismétlődő csoportoknak* is nevezik.

### Példa (folytatás)

Az előállított KATALÓGUS relációséma SZERZŐ attribútuma nyilván összetett, mivel egy beágyazott relációséma. (Megjegyezzük, hogy a definíció nem követeli meg, csupán megengedi az összetett attribútumot, tehát a későbbiekben előállítandó magasabb normálformájú relációsémák egyben nulladik normálformájúak is lesznek.)

A gyakorlatban a nulladik normálforma legnagyobb jelentősége, hogy ebben tisztázzuk; mely attribútumokra is tartunk igényt a későbbi modellben.

Tekintettel arra, hogy viszonylag elhanyagolható információ egy könyv oldalainak száma (bár kétségtelen, hogy egy vastag könyvért hajlamosak vagyunk magasabb árat fizetni), döntünk úgy, hogy az oldalszámot kihagyjuk a modellből.

Ezekután a KATALÓGUS relációséma 0NF alakja:

KATALÓGUS(ISBN, KÖNYV\_CÍME, SZERZŐ(ISBN, NÉV), TÉMA, KIADÁSI\_ÉV,  
ÁR, KIADÓ, KIADÓ\_CÍME)

A további vizsgálatokban ebből fogunk kiindulni.

## Első normálforma, 1NF

Egy  $R\langle A \rangle$  relációséma *első normálformájú*, ha *nem* lehetnek benne összetett attribútumok. Az első normálformát 1NF módon is jelöljük.

### Megjegyzés

Az első normálformára (1NF alakra) hozás során a normalizálás egyik fő céljaként kitűzött redundancia-csökkentés helyett általában inkább még növeljük is a redundanciát.

Tekintsük például egy szakértői nyilvántartás alábbi részletét:

...	SZAKÉRTŐ_NEVE	SZAKKÉPESÍTÉS	...
...	...	...	...
...	Tóth Bálint	villamosmérnök, informatikus	...
...	...	...	...

A SZAKKÉPESÍTÉS attribútum azonban ebben a formában logikailag összetettnek tekinthető. Tekinethetnének a korábban mondottak szerint persze egyszerűnek is, ám az általunk választott logikai módszer szerint célszerű összetett adatként tekinteni, és részeire bontva külön tárolni. Ezt megtehetjük az alábbi módon:

...	SZAKÉRTŐ_NEVE	SZAKKÉPESÍTÉS	...
...	...	...	...
...	Tóth Bálint	villamosmérnök	...

...	Tóth Bálint	informatikus	...
...	...	...	...

Ily módon a SZAKKÉPESÍTÉS már egyszerű attribútum, azonban az így kapott relációséma redundanciája nyilván megnőtt.

### Példa (folytatás)

Térjünk vissza a korábbi példánkhoz. Mivel a KATALÓGUS relációséma SZERZŐ attribútuma, mint beágyazott relációséma összetett, tehát ki kell emelnünk. Kérdés, hogy megtehetjük-e, nem történik-e információvesztés!

Először azonban azt kell tisztáznunk, hogy miért nem jártunk el a KATALÓGUS relációséma esetén a SZERZŐ attribútummal oly módon, mint az imént a szakértői nyilvántartásban a SZAKKÉPESÍTÉS attribútummal, azaz miért nem elégedtünk meg a többértékű SZERZŐ attribútum értékei szerint történő rekordtöbbszörözéssel.

Az összetett attribútumok (ismétlődő csoportok) megszüntetése esetén a legfontosabb kérdés az, hogy mit is tekintünk az adatbázis objektumának.

A szakértői nyilvántartó rendszerben az adatbázis objektuma egy szakértő. Ennek értelmében, ha valakinek több szakképesítése van, akkor ő minden szakképesítése szerint nyilván külön objektuma e nyilvántartásnak.

A katalógusban azonban egy könyv attól, hogy több szerzője van, még egyetlen objektum, melyet az ISBN szám azonosít. Ha a SZAKKÉPESÍTÉS attribútumhoz hasonlóan a SZERZŐ attribútum szerint is megsokszoroznánk a KATALÓGUS relációséma rekordjait, akkor az ISBN attribútum a továbbiakban már nem volna kulcs, és minden könyv annyi objektumként jelenne meg, ahány szerzője van. Ez természetesen elfogadhatatlan, tehát a rekordtöbbszörözés nem járható út a KATALÓGUS relációséma első normálformára hozásában. Marad tehát az általunk alkalmazott beágyazott relációséma.

Ezekután visszatérhetünk arra a kérdésre, hogy az első normálformára hozás során kiemelhetjük-e ezt a SZERZŐ(ISBN, NÉV) beágyazott relációsémát a KATALÓGUS relációsémából.

A relációsémák dekompozíciójánál azt láttuk, hogy ha egy  $f = \{C\} \rightarrow \{A\}$  funkcionális függőséggel definiált  $R(A, B, C)$  relációsémát a függőség magattribútuma (C) szerint bontunk fel, akkor az így kapott  $\phi[R(\underline{A})] = \{R''(B, C), R''(A, C)\}$  dekompozíció veszteségmentes lesz, vagyis ebből visszaállítható az eredeti relációséma.

Ennek alapján feltételezhetjük, hogy a kulcs mentén történő dekompozíció általában veszteségmentes (az állítás igaz, de most nem bizonyítjuk), márpedig a fenti KATALÓGUS relációséma

KATALÓGUS'(ISBN, KÖNYV\_CÍME, TÉMA, KIADÁSI\_ÉV, ÁR, KIADÓ, KIADÓ\_CÍME),  
SZERZŐ'(ISBN, NÉV)

relációsémákra való dekompozíciója az ISBN attribútum (azaz kulcs) mentén történt. Ennek belátásához elég, ha felírjuk az eredeti KATALÓGUS relációsémához tartozó függőséget:

$$f_{\text{KATALÓGUS}} = \{ \text{ISBN} \} \rightarrow \{ \text{KÖNYV\_CÍME}, \text{SZERZŐ}, \text{TÉMA}, \text{KIADÁSI\_ÉV}, \text{ÁR}, \text{KIADÓ}, \text{KIADÓ\_CÍME} \}.$$

A kapott KATALÓGUS' és SZERZŐ relációsémák már láthatóan 1NF alakúak.

## Második normálforma, 2NF

Egy 1NF  $R\langle A \rangle$  relációséma *második normálformájú*, ha minden másodlagos attribútum a relációséma bármely kulcsától teljesen függ. Jelölésére a 2NF alakot is használjuk.

### Megjegyzés

A normalizálási folyamatnak ebben a lépésében tehát két teendőnk van. Az egyik, hogy kiemeljük önálló függőségként a részleges függőségeket, másrészt pedig, hogy az eredeti függőségből töröljük a kiemelt függőség képattribútumait. Természetesen mindenekelőtt fel kell írunk a vizsgált relációsémát meghatározó függőségeket.

Vegyük észre, hogy a két attribútumból álló relációsémák, valamint az egy attribútumból álló (úgynevezett *egyszerű*) kulccsal rendelkező relációsémák mindig 2NF alakúak, sőt (mint majd látni fogjuk) teljesen normalizáltak. Ezeket így már nem kell tovább alakítanunk.

### Példa (folytatás)

A fentiek szerint először is megállapíthatjuk, hogy a

$SZERZŐ\langle \underline{ISBN}, \underline{NÉV} \rangle$

relációsémánk 2NF alakú, vele tehát nincs teendőnk. Mivel a

$KATALÓGUS'\langle \underline{ISBN}, KÖNYV\_CÍME, TÉMA, KIADÁSI\_ÉV, \hat{ÁR}, KIADÓ, KIADÓ\_CÍME \rangle$ ,

relációsémánk egyszerű kulccsal rendelkezik, így nyilván ez is 2NF alakú. Gyakorlásképpen azért írjuk fel a függőségét, mely nyilván az alábbi:

$$f_1 = \{ \underline{ISBN} \} \rightarrow \{ KÖNYV\_CÍME, TÉMA, KIADÁSI\_ÉV, \hat{ÁR}, KIADÓ, KIADÓ\_CÍME \}.$$

### Megjegyzés

Ha nem használtuk volna az ISBN számot, akkor a könyvek egyértelmű azonosításához a

$\{ KÖNYV\_CÍME, SZERZŐ, KIADÁSI\_ÉV \}$

kulcsra lett volna szükségünk, és az  $f_1$  függőség helyett az

$$f_{11} = \{ KÖNYV\_CÍME, SZERZŐ, KIADÁSI\_ÉV \} \rightarrow \{ TÉMA, \hat{ÁR}, KIADÓ, KIADÓ\_CÍME \}$$

függőséget írhattuk volna fel. Mivel itt a kulcs összetett, így már valóban meg kellene vizsgálnunk a relációséma függőségét abból a szempontból, hogy nem tartalmaz-e részleges függőséget (Ezúttal az egyszerűség kedvéért tételezzük fel, hogy a SZERZŐ attribútum nem összetett.) Nos, egy könyv témáját a könyv címe és szerzője egyértelműen meghatározza (a történelmi „átértékelésektől” eltekintve a kiadás éve ebben nem játszik szerepet). Ezt a részleges függést kiemelve, az alábbi függőségeket kapnánk

$$f_{111} = \{ KÖNYV\_CÍME, SZERZŐ, KIADÁSI\_ÉV \} \rightarrow \{ \hat{ÁR}, KIADÓ, KIADÓ\_CÍME \},$$

$$f_{112} = \{ KÖNYV\_CÍME, SZERZŐ \} \rightarrow \{ TÉMA \},$$

melyek által meghatározott

$KATALÓGUS_{111}\langle \underline{KÖNYV\_CÍME}, \underline{SZERZŐ}, \underline{KIADÁSI\_ÉV}, \hat{ÁR}, KIADÓ, KIADÓ\_CÍME \rangle$  és

$KATALÓGUS_{112}\langle \underline{KÖNYV\_CÍME}, \underline{SZERZŐ}, TÉMA \rangle$

relációsémák már láthatóan valóban 2NF alakúak.

### Harmadik normálforma, 3NF

Egy 2NF  $R\langle A \rangle$  relációséma *harmadik normálformájú*, ha egyetlen másodlagos attribútum sem függ tranzitív módon valamely kulcstól. Jelölésére a 3NF alakot is használjuk.

#### Megjegyzés

A belső-függőség definícióját követő állítás (a tranzitivitási kényszer szabálya) szerint, ha egy relációsémában van belső függés (azaz nem-kulcs, tehát másodlagos attribútumtól való függés), akkor ott tranzitív függés is van. Tehát a tranzitív függés (és így a redundancia) felszámolása érdekében meg kell vizsgálnunk van-e belső-függőség. Ha van, akkor azt a részleges függőség megszüntetésénél mondottak szerint ki kell emelnünk.

#### Példa (folytatás)

Vizsgáljuk meg tehát az

$$f_1 = \{ \text{ISBN} \} \rightarrow \{ \text{KÖNYV\_CÍME}, \text{TÉMA}, \text{KIADÁSI\_ÉV}, \text{ÁR}, \text{KIADÓ}, \text{KIADÓ\_CÍME} \}$$

függőséget, tartalmaz-e belső függőségeket.

Könnyen észrevehető, hogy a

$$f_2 = \{ \text{KIADÓ} \} \rightarrow \{ \text{KIADÓ\_CÍME} \}$$

egy belső-függőség, tehát  $f_1$ -ből kiemelve kapjuk az

$$f_3 = \{ \text{ISBN} \} \rightarrow \{ \text{KÖNYV\_CÍME}, \text{TÉMA}, \text{KIADÁSI\_ÉV}, \text{ÁR}, \text{KIADÓ} \}$$

függőséget. Láthatóan  $f_2$  és  $f_3$  már nem tartalmaznak további belső függéseket, tehát a hozzájuk tartozó

**KIADÓK** $\langle \underline{\text{KIADÓ}}, \text{KIADÓ\_CÍME} \rangle$ , és

**KATALÓGUS** $\langle \underline{\text{ISBN}}, \text{KÖNYV\_CÍME}, \text{TÉMA}, \text{KIADÁSI\_ÉV}, \text{ÁR}, \text{KIADÓ} \rangle$

relációsémák már 3NF alakúak. A definícióból következően 3NF alakú a korábban már létrehozott

**SZERZŐ** $\langle \underline{\text{ISBN}}, \underline{\text{NÉV}} \rangle$

relációséma is.

Ily módon elő is állítottuk adatbázisunk 3NF alakú

**Szakkatalógus** = { **KATALÓGUS**, **SZERZŐ**, **KIADÓK** }

adatbázismodelljét, amely tehát három relációsémából (a gyakorlati megvalósítás során három adattáblából) áll.

Emlékeztetünk arra, hogy az egyes relációsémák közötti kapcsolatokat az idegen kulcsok biztosítják. (Ezek olyan attribútumai egy relációsémának, melyek egy másik relációsémában kulcsot alkotnak.)

## Boyce-Codd normálforma, BCNF

A tranzitív függőséget (és így a redundanciát) a relációséma 3NF alakja még nem zárja ki teljesen, hiszen elsődleges attribútum tranzitív függése még fennállhat. Célszerű ezért egy újabb, a Boyce-Codd normálforma bevezetése, melyet BCNF módon is jelölünk és az alábbi módon értelmezünk.

Egy 3NF  $R\langle A \rangle$  relációséma *Boyce-Codd normálformájú*, ha kulcs valódi része nem függ más attribútumtól (azaz nincs kulcstörés). Másképpen ezt úgy is mondhatnánk, hogy egyáltalán nincs tranzitív függés.

### Megjegyzés

Az egyszerű (egyetlen attribútumot tartalmazó) kulccsal rendelkező 3NF relációsémák értelemszerűen BCNF alakúak.

A szakirodalomban további normálformákat is definiálnak, az érdeklődőknek javasoljuk az *Ullmann-Widom* szerzőpáros „Adatbázisrendszerek” című könyvét (lásd az irodalomjegyzéket).

### Állítás

A BCNF alakú relációsémák nem tartalmaznak (funkcionális függőségből származó) redundanciát.

### Bizonyítás

Mivel egy BCNF alakú relációséma már nem tartalmaz tranzitív függést, így nincs kitalálható, „három adatból a negyedik” típusú, azaz redundáns adat.

### Példa

Könnyen belátható, hogy az előző példasorozat eredményeként BCNF normálformát kaptunk, ezért e normálforma bemutatása érdekében tekintsünk egy másik példát.

Legyen egy iskolai nyilvántartás relációsémája a

$$\{ \text{TANTÁRGY, TANÁR, DIÁK} \}$$

attribútumhalmazon értelmezett

$$R\langle \text{TANTÁRGY, TANÁR, DIÁK} \rangle.$$

#### 1. lépés.

Először foglaljuk össze a rendelkezésünkre álló ismereteket.

- 1.1. Egyetlen attribútum sem határozza meg egyértelműen az összes többi.
- 1.2. A tantárgy és a tanár együttesen nem határozzák meg a diákot.
- 1.3. A diákoknak minden tanáruk csak egy tantárgyat tanít (ebben az iskolában).
- 1.4. Általában is igaz, hogy minden tanár csak egy tantárgyat tanít (ebben az iskolában).
- 1.5. A diákoknak minden tantárgyat csak egyetlen tanár tanít (ez elég általános).

#### 2. lépés.

Írjuk fel a fenti ismereteket függőségekként. Először azonban – ha triviális is – írjuk fel az

$$f_0 = \{ \text{TANTÁRGY, TANÁR, DIÁK} \} \rightarrow \{ \text{TANTÁRGY, TANÁR, DIÁK} \}$$

egységfüggőséget, mivel ez kifejezi azt, hogy a

$$\{ \text{TANTÁRGY, TANÁR, DIÁK} \}$$

attribútumhalmaz kulcs. A *dekompozíciós Armstrong-szabály* értelmében az  $f_0$  függőség egyenértékű módon felbontható az

$$f_{01} = \{ \text{TANTÁRGY, TANÁR, DIÁK} \} \rightarrow \{ \text{TANTÁRGY} \},$$

$$f_{02} = \{ \text{TANTÁRGY, TANÁR, DIÁK} \} \rightarrow \{ \text{TANÁR} \}, \text{ és}$$

$$f_{03} = \{ \text{TANTÁRGY, TANÁR, DIÁK} \} \rightarrow \{ \text{DIÁK} \}$$

függőségekre. Tekintsük át ezekután az 1.1.-1.5. ismeretekből következő függőségeket.

2.1. Az 1.1.-ből nem következik függőség, mindössze annyi, hogy egyszerű kulcs nem lesz megfelelő.

2.2. Az 1.2. szerint a  $\{ \text{TANTÁRGY, TANÁR} \}$  összetett kulcsként szintén nem megfelelő.

2.3. Az 1.3.-ból már következik az alábbi függőség:

$$f_1 = \{ \text{TANÁR, DIÁK} \} \rightarrow \{ \text{TANTÁRGY} \}.$$

2.4. Az 1.4. függőségként felírva:

$$f_2 = \{ \text{TANÁR} \} \rightarrow \{ \text{TANTÁRGY} \}.$$

2.5. Végül az 1.5. is felírható függőségként:

$$f_3 = \{ \text{TANTÁRGY, DIÁK} \} \rightarrow \{ \text{TANÁR} \}.$$

Nyilvánvalóan még számos további függőséget felírhatnánk, azonban ezek – hasonlóan az  $f_0, f_{01}, f_{02}, f_{03}$  függőségekhez – triviálisak volnának, vagyis nem tükröznének újabb ismereteket vizsgálatunk tárgyáról.

### 3. lépés.

Elemezzük a kapott függőségeket. Láthatóan az  $f_1$  és  $f_2$  függőségek miatt az **R** relációséma nem 2NF alakú, az  $f_2$  és  $f_3$  függőségek miatt pedig nem BCNF alakú.

### 4. lépés.

Normalizálás. Az  $f_1$  és  $f_3$  függőségek alapján a

$$\{ \text{TANÁR, DIÁK} \}, \text{ és a}$$

$$\{ \text{TANTÁRGY, DIÁK} \}$$

attribútumhalmazok (alternatív) kulcsai az **R**(TANTÁRGY, TANÁR, DIÁK) relációsémának.

Elvégezve a bevezetett normalizálásokat az alábbi dekompozíciót kapjuk

$$\phi[\mathbf{R}(\text{TANTÁRGY, TANÁR, DIÁK})] = \{ \mathbf{R}_1(\text{TANTÁRGY, TANÁR}), \\ \mathbf{R}_2(\text{TANÁR, DIÁK}) \},$$

melyben nyilván

$$\mathbf{K}\{\mathbf{R}_1\} = \{ \text{TANÁR} \}, \text{ és}$$

$$\mathbf{K}\{\mathbf{R}_2\} = \{ \text{TANÁR, DIÁK} \}.$$

A fenti  $\phi$  dekompozíció egyben az iskolai nyilvántartás adatbázismodellje is.

### Megjegyzés

Vizsgáljuk meg a fenti  $\phi$  dekompozíciót. (Megjegyezzük, hogy a „Relációséma veszteségmentes dekompozíciója” pontnál bemutatott példa **R**(A) relációsémája A = (TANTÁRGY, DIÁK, TANÁR), valamint  $\phi_2 = \phi$  összerendeléssel a jelen példához hasonlít.) Könnyen belátható, hogy a  $\phi$  dekompozíció veszteségmentes, és az  $f_2$  függőséget közvetlenül megőrzi (**R**<sub>1</sub>-ben).

Tekintsük az  $f_1$  függőséget. Bármely konkrét (TANÁR<sub>0</sub>, DIÁK<sub>0</sub>) értékpáros az **R**<sub>2</sub> relációsémában nyilván csak egyszer fordulhat elő, az **R**<sub>1</sub>-beli TANÁR attribútumra vonatkozó  $f_2$

függőség miatt pedig e pár  $TANÁR_0$  értékéhez csak egyetlen  $TANTÁRGY_0$  érték tartozhat, tehát teljesül az  $f_1$  függőség is.

Figyeljünk fel arra, hogy az  $f_1$  függőséget nem közvetlenül őrzi meg a  $\phi$  dekompozíció. Nem is teheti, hiszen az általa tartalmazott alsémák egyikére sem értelmezhető  $f_1$ . Ezt az  $R_1$  és az  $R_2$  relációsémák együttesen őrzik meg. Azt mondhatjuk tehát, hogy az  $f_1$  függőséget a  $\phi$  dekompozíció *strukturálisan őrzi meg*. Ez egy új jelenség, tárgyalásunk keretei azonban nem engedik meg ennek részletes kifejtését.

Végül tekintsük az  $f_3$  függőséget. Ez nyilvánvalóan nem értelmezhető a  $\phi$  dekompozíció által tartalmazott alsémákban, de a  $\phi$  ezt strukturálisan sem őrzi meg, vagyis az  $f_3$  függőség „elveszett”. Mi ennek a következménye? Mint korábban láttuk a nem függőségőrző dekompozíciók esetén lehetséges „hamis” rekordok felvitele, vagyis bevihetünk az alsémákba olyan rekordokat, melyeket az eredeti relációsémán értelmezett függőségek megtiltanának.

Például, ha a Kiss Dénes diáknak Tóth Pál a matematika tanára, akkor az  $R(TANTÁRGY, TANÁR, DIÁK)$  relációsémára épülő adattáblába az  $f_3$  függőség nem engedi felvinni a („matematika”, „Kara Péter”, „Kiss Dénes”) rekordot, míg e rekord („matematika”, „Kara Péter”) illetve („Kara Péter”, „Kiss Dénes”) vetületei az  $R_1$  illetve  $R_2$  alsémák tábláiba minden nehézség nélkül bevihetők. Az pedig, hogy baj történt, csak akkor derül ki, amikor lekérdezzük a diákok adatait és a

DIÁK	TANTÁRGY	TANÁR
...	...	...
Kiss Dénes	matematika	Tóth Pál
Kiss Dénes	matematika	Kara Péter
...	...	...

listát kapjuk, mely már nyilvánvalóan ellentmondásban van az  $f_3$  függőséggel.

A normalizálás ebben az esetben tehát megsértette az adatbázis *integritását*, vagyis elvesztette az eredeti modellben megfogalmazott függőségek egy részét. (A függőségeket az Oracle-rendszerben majd *megszorításoknak* nevezzük.)

A gyakorlatban az ilyen esetekben mindig el kell gondolkodnunk azon, hogy vajon nem történt-e „túlnormalizálás”. Tény azonban, hogy az esetleges adatvesztések (melyek a következő pontban bemutatásra kerülő adatbázis-anomáliák révén érhetik az adatbázist) általában visszariasztják a tervezőket a denormalizálásoktól, és az adatbázis-modell integritási sérüléseit inkább más módon korrigálják. (Például az adatbevitelkor aktivizálódó speciális eljárások, az úgynevezett triggererek segítségével többtáblás megszorításként valósítják meg a dekompozíció során „elveszett” megszorításokat.) Ezekkel a III. részben, a PL/SQL nyelv kapcsán fogunk foglalkozni.

## Adatbázis-anomáliák

A fentiekben áttekintettük az adatbázis-tervezés alapvető lépéseit, fogalmait. Az adatbázis-modell 2NF, 3NF és BCNF alakra hozása érzékelhető redundancia-csökkenéssel járt (az 1NF még nem!), és ez összhangban volt eredeti célkitűzésünkkel. A „Normalizálás” alfejezet bevezetőjében azonban utaltunk arra is, hogy a normalizálással bizonyos rendellenes adatbázis-működések is ki szeretnénk zárni. Eljött az ideje annak, hogy – mintegy a fejezet

lezárásaként – áttekintsük e rendellenességeket (más szóval anomáliákat), továbbá egyben azt is megvizsgáljuk, hogy az általunk bevezetett eszközök segítségével e rendellenességek kiküszöbölhetőkké váltak-e.

Induljunk ki az 1NF alakból, hiszen az olyan relációsémák, melyek nincsenek 1NF alakban (tehát összetett attribútumokat, azaz ismétlődő csoportokat tartalmaznak) eleve alkalmatlanok arra, hogy adattáblákkal reprezentáljuk.

Az alábbiak során tekintsük a korábban már bevezetett

**KATALÓGUS'**(ISBN, KÖNYV\_CÍME, TÉMA, KIADÁSI\_ÉV, ÁR, KIADÓ, KIADÓ\_CÍME), relációsémát, melyről tudjuk, hogy 2NF alakú (tehát 1NF alakú is), de nem 3NF alakú, és így nem BCNF alakú.

## Módosítási anomália

Módosítási anomáliának nevezzük azt a jelenséget, amikor az adatbázis valamely adatának a megváltoztatása során történő hiba az adatbázis konzisztenciáját (azaz ellentmondásmentességét) veszélyezteti.

Tegyük fel, hogy valamelyik kiadó elköltözik, tehát módosítani szeretnénk e kiadó címét. Ha az adatbázisunk modellje a fenti **KATALÓGUS'** relációséma, akkor már önmagában kellemetlen, hogy ezt mindazon rekordon el kell végezni, amelyben az adott kiadó szerepelt (ezt a redundancia okozza). Az viszont már az adatbázis konzisztenciáját veszélyezteti, ha ez a módosítási folyamat bármilyen okból (például egy áramkimaradás, vagy rendszerösszeomlás miatt) megszakad, ugyanis így egyszerre szerepelhet az adatbázisban a régi és az új cím. Abban az esetben pedig, ha ez a módosítás nem egy megfelelő SQL utasítással, hanem „kézzel” (tehát közvetlen adatátírással) történik, akkor még az is előfordulhat, hogy tévesztés miatt különböző címadatok kerülnek az adatbázisba.

Könnyen belátható, hogy a redundancia, vagyis a részleges függések, illetve a tranzitív függések megszüntetésével, tehát az adatbázis BCNF alakjában a módosítási anomália veszélye elhárul.

Feltehető ezután a kérdés, hogy egy BCNF alakú adatbázisban a módosítás nem okozhat-e hibát? Természetesen okozhat! Hibás adatot bármikor be lehet vinni egy adatbázisba, ez ellen nincs védelem. Ám egy irredundáns adatbázisban ez nem okoz inkonzisztenciát, ilyen módon ha észre vesszük a hibát, könnyen ki tudjuk javítani. Nem az a végzetes baj tehát, ha például egy kiadó címét rosszul írjuk (csak észre vesszük előbb-utóbb), hanem az, ha ez a cím különbözőképpen szerepel. Ezt kijavítani egy nagy adatbázisban (mely erre az adatra nézve nem irredundáns) szinte lehetetlen.

## Törlési anomália

Törlési anomália fordul elő olyankor, amikor valamely fölöslegesnek, vagy hibásnak minősült adat törlése során információvesztés történik.

Gondoljunk ismét a fenti **KATALÓGUS'** relációsémára. Tegyük fel, hogy a könyvesboltból (melynek ez a katalógusa) kifogy egy olyan könyv, melynek kiadója csak ezzel a könyvvel szerepelt a kínálatban. Nagy baj ez, mivel az által, hogy kitöröltük a könyvet a katalógusból, kitöröltük a kiadójának adatait is, úgyhogy most már rendelni sem tudunk belőle. Mi is okozta ezt a rendellenességet? Láthatóan az, hogy a kiadó adatait együtt tároltuk a könyv

adataival. A könyv és a kiadója két különböző objektum, melyek mindegyike fontos az adatbázisban, tehát különböző relációban (külön adattáblában) kell tárolni őket. A külön történő tárolás egy belső függést számolt fel, tehát ennek az anomáliának az elhárításához az adatbázis 3NF alakja szükséges.

### **Bővítési anomália**

Bővítési anomália következik be olyankor, amikor valamely adat bevitele azért bizonyul lehetetlennek, mert rá vonatkozóan nincs kulcs.

Tegyük fel, hogy fenti **KATALÓGUS** relációsémát alkalmazó könyvesbolt vezetője együttműködési szerződést ír alá valamelyik kiadóval. Ezt követően szeretné beírni az adatbázisba ennek a kiadónak az adatait, hogy majd rendelni tudjanak tőlük (tegyük fel, hogy ehhez az adatbázishoz kapcsolódik egy rendeléskezelő program is). Az előbbihez hasonló problémával kell szembenéznie, csak most nem elveszik a cím, hanem már be se tudja írni. Ennek az az oka, hogy a **KATALÓGUS** relációsémájú adatbázis objektumai könyvek, melyeknek az ISBN számuk a kulcsadatuk, így enélkül nem lehet adatot bevinni. Márpedig az új kiadó adataihoz nem rendelhető ISBN szám. A problémát ezúttal is az okozza, hogy egy rekordban két különböző objektum adatait tároljuk, de a rekord kulcsa csak az egyik objektumnak kulcsa. Miután a kiadó független e kulcstól, így az adataival önmagában nem lehet elvégezni a bővítést. A belső-függések felszámolása, külön relációséma megalkotása, vagyis az adatbázis 3NF alakra hozása nyilvánvalóan ezt a problémát is megoldja.

