

12.előadás: Adatbázisok-I.

dr. Hajas Csilla (ELTE IK) (2020)
<http://sila.hajas.elte.hu/>

Normalizálás (VM BCNF, FŐ VM 3NF)

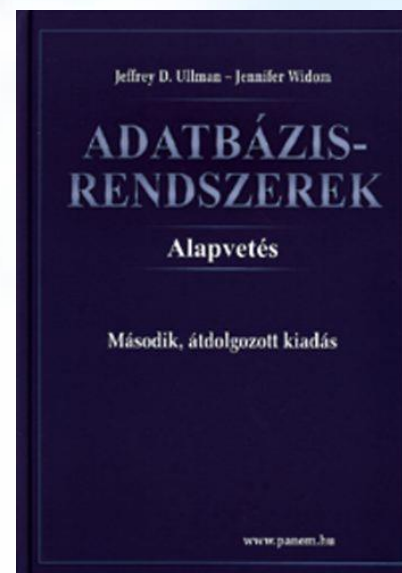
A mai témakörök a Tankönyvben:

3.3. Boyce-Codd normálforma és egy algoritmus BCNF-ra való felbontásra

3.4.1-3.4.3. Információ visszanyerése a komponensekből, és Chase-teszt a veszteségmentesség ellenőrzésére

3.4.4. Függőségek megőrzése

3.5. Harmadik normálforma és 3NF-szintetizáló algoritmus



(Tk.3.3.1.) Motiváció: Bevezető példa

- Mi lenne, ha a Sörivók és a Sörök, meg a köztük lévő Szeret illetve aKedvencSör kapcsolatokat az eddigi több tábla helyett egy táblába egyesítve tárolnánk?

Sörivó(név, cím, sör, gyártó, aKedvencSör)

név	cím	sör	gyártó	aKedvenc
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

- **Redundancia**, az információk feleslegesen ismétlődnek több sorban, hiszen mindenkinek csak egy lakcíme és egy aKedvencSöre lehet, továbbá a söröknek is csak egy gyártója lehet!

Hibás tervezés problémái

A rosszul tervezettség anomáliákat is eredményez
Sörivő(név, cím, sör, gyártó, aKedvencSör)

név	cím	sör	gyártó	aKedvenc
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

- **Módosítási anomália:** ha az A.B. gyártót B.A.-ra módosítjuk, megteesszük-e ezt minden sornál?
- **Törlési anomália:** Ha senki sem szereti a Bud sört, azzal töröljük azt az infót is, hogy ki gyártotta.
- **Beillesztési anomália:** és felvinni ilyen gyártót?

Relációs adatbázissémák tervezése

- Cél: redundancia csökkentése és az anomáliák megszüntetése.
 - **Redundancia**: az adatok felesleges ismétlődése
 - **Módosítási anomália**: egy adat egy előfordulását megváltoztatjuk, más előfordulásait azonban nem.
 - **Törlési anomália**: törléskor olyan adatot is elveszítünk, amit nem szeretnénk.
 - **Beillesztési anomália** : nem tudunk felvinni adatsorokat

(Tk.3.3.2.) Relációk felbontása

Az anomáliák megszüntetésének útja a relációk felbontása (dekompozíciója). R felbontása azt jelenti, hogy R attribútumait szétosztjuk két vagy több reláció sémáját alakítjuk ki belőlük:

- **Definíció:** $d=\{R_1, \dots, R_k\}$ az (R, F) **dekompozíciója**, ha nem marad ki attribútum, azaz $R_1 \cup \dots \cup R_k = R$.
Az adattábla felbontását projekcióval végezzük.
- **Megj.:** Tk.3.3.2. Relációk felbontása c. részhez, lásd 91.o. kiegészítés: a felbontás nemcsak két relációra, hanem több új relációsémára történhet!
- **Példa:** $R=ABCDE$ felbontása $d=\{AD, BCE, ABE\}$
ez pl. 3 tagú felbontás $R_1=AD$, $R_2=BCE$, $R_3=ABE$

Felbontásra vonatkozó elvárások

➤ Elvárások

(1) **A vetületek** legyenek jó tulajdonságúak, és a vetületi függőségi rendszere egyszerű legyen (normálformák: BCNF, 3NF, 4NF, később)

(2) **A felbontás** is jó tulajdonságú legyen, vagyis ne legyen információvesztés:

Veszteségmentes legyen a felbontás (VM)

(3) **Függőségek megőrzése** a vetületekben (FŐ)

➤ **Tételek** (ezekre nézünk majd algoritmusokat)

➤ Mindig van **VM BCNF-ra való felbontás**

➤ Mindig van **VM FŐ 3NF-ra való felbontás**

(Tk.3.3.3.) Boyce-Codd normálforma

➤ Definíció:

R reláció **Boyce-Codd normálformában**,
BCNF-ban van, ha

- minden $X \rightarrow Y$ **nem-triviális FF-re** R -ben
(nem-triviális, vagyis Y nem része X -nek)
- az X **szuperkulcs** (vagyis az X -ben szereplő attribútumok funkcionálisan meghatározzák a reláció összes többi attribútumát: $X \rightarrow R$)

Példa: nincs BCNF-ban

Sörivók(név, cím, kedveltSörök, gyártó, kedvencSör)

FF-ek: $név \rightarrow cím$ $kedvencSör$, $kedveltSörök \rightarrow gyártó$

- Itt egy kulcs van: { $név$, $kedveltSörök$ }.
- A baloldalak egyik FF esetén sem szuperkulcsok.
- Emiatt az *Sörivók* reláció nincs Boyce-Codd normálformában.

egy másik példa: nincs BCNF-ban

Sörök(név, gyártó, gyártóCím)

FF-ek: $név \rightarrow gyártó$, $gyártó \rightarrow gyártóCím$

- Az egyetlen kulcs { $név$ } .
- $név \rightarrow gyártó$ nem sérti a BCNF feltételét, de a $gyártó \rightarrow gyártóCím$ függőség igen.

Emiatt az *Sörök* reláció nincs BCNF-ban.

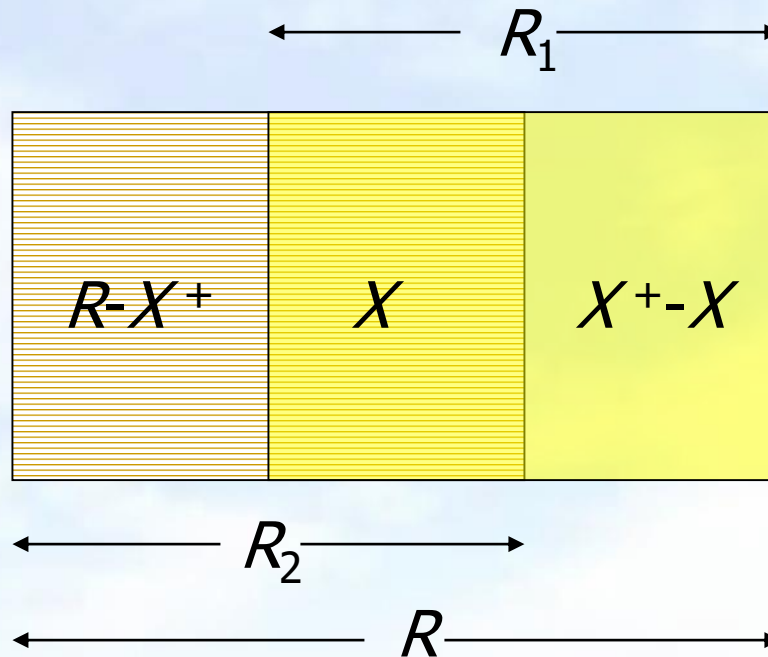
Példa BCNF-ra

- **Lemma:** Azt állítjuk, hogy bármely két attribútumból álló reláció BCNF-ban van.
- **Bizonyítás:** Tegyük fel, hogy ezek az attribútumok A és B.
- Nincs nem-triviális függőség. Ekkor a BCNF feltétel mindenképpen érvényes, hiszen csak a nem triviális függőségek sérthetik meg a feltételt. (Egy kulcs: {A,B}.)
- $A \rightarrow B$ fennáll, de $B \rightarrow A$ nem áll fenn. Ekkor egy kulcs: A, és minden nem triviális függőség tartalmazza A-t a bal oldalon (valójában a bal oldal csak A lehet). Tehát nem sérül a BCNF feltétel. (Szimmetrikusan úgy A és B csere)
- Mindkét $A \rightarrow B$ és $B \rightarrow A$ fennáll. Ekkor mindkét attribútum, A is és B is kulcs. Bármelyik függőséget nézzük, a két attribútum közül az egyik a bal oldalon áll, emiatt nem sértheti meg a BCNF feltételt.

(Tk.3.3.4.) BCNF-re való felbontás ---1

- Adott R reláció és F funkcionális függőségek.
- 1.) Ellenőrizzük, hogy R BCNF-ban van-e.
 - Ha igen, akkor készen vagyunk, és $\{R\}$ a válasz.
 - Van-e olyan $X \rightarrow Y$ FF, ami sérti a BCNF-t?
- 2.) Ha vannak BCNF-t megsértő függőségek, akkor $X \rightarrow Y$ legyen egy ilyen.
 - Kiszámítjuk X^+ -t:
 - Legyen $R_1 = X^+$ ez valódi részhalmaza R -nek, nem szerepel benne az összes attribútum, mivel X nem superkulcs (az $X \rightarrow Y$ sértette a BCNF-t).

BCNF-re való felbontás ---2



- **Folyt. 2.)** R -t helyettesítsük az alábbiakkal:
 - $R_1 = X^+$ az egyik relációséma és
 - $R_2 = X \cup (R - X^+)$ ebben legyenek benne X attribútumai és azok az R -beli attribútumok, amelyek nincsenek X^+ -ban.

BCNF-re való felbontás ---3

- 3.) Az F -beli funkcionális függőségeket vetítsük le az R_1 és R_2 reláció-sémákra, a vetületek legyenek rendre F_1 és F_2
- 4.) Rekurzívan folytassuk a felbontást R_1 -re illetve R_2 -re ennek az algoritmusnak a felhasználásával, amíg a kapott séma a BCNF-nak megfelelő nem lesz. A végeredmény a dekompozíciók eredményeinek uniója lesz.

Megj: Az eljárás biztosan véget ér, mert minden alkalommal a kapott relációk kevesebb attribútumot tartalmaznak és a 2 attribútumos már mindig BCNF.

Példa: BCNF dekompozíció ---1

Sörivók(név, cím, kedveltSörök, gyártó, kedvencSör)

$F = \text{név} \rightarrow \text{cím}, \text{név} \rightarrow \text{kedvencSör},$
 $\text{kedveltSörök} \rightarrow \text{gyártó}$

- Vegyünk $\text{név} \rightarrow \text{cím}$ FF-t:
- $\{\text{név}\}^+ = \{\text{név}, \text{cím}, \text{kedvencSör}\}.$
- A dekomponált relációsémák:
 1. Sörivók1(név, cím, kedvencSör)
 2. Sörivók2(név, kedveltSörök, gyártó)

Példa: BCNF dekompozíció ---2

- Meg kell néznünk, hogy az Sörivók1 és Sörivók2 táblák BCNF-ben vannak-e.
- Az FF-ek projektálása könnyű.
- A Sörivók1(név, cím, kedvencSör), az FF-ek név->cím és név->kedvencSör.
- Tehát az egyetlen kulcs: {név}, azaz Sörivók1 relációséma BCNF-ben van.

Példa: BCNF dekompozíció ---3

- Az $Sörivók2(\underline{név}, \underline{kedveltSörök}, gyártó)$ esetén
az egyetlen FF: $kedveltSörök \rightarrow gyártó$,
az egyetlen kulcs: $\{név, kedveltSörök\}$.
- Sérül a BCNF.
- $kedveltSörök^+ = \{kedveltSörök, gyártó\}$, a $Sörivók2$ felbontása:
 1. $Sörivók3(\underline{kedveltSörök}, gyártó)$
 2. $Sörivók4(\underline{név}, \underline{kedveltSörök})$

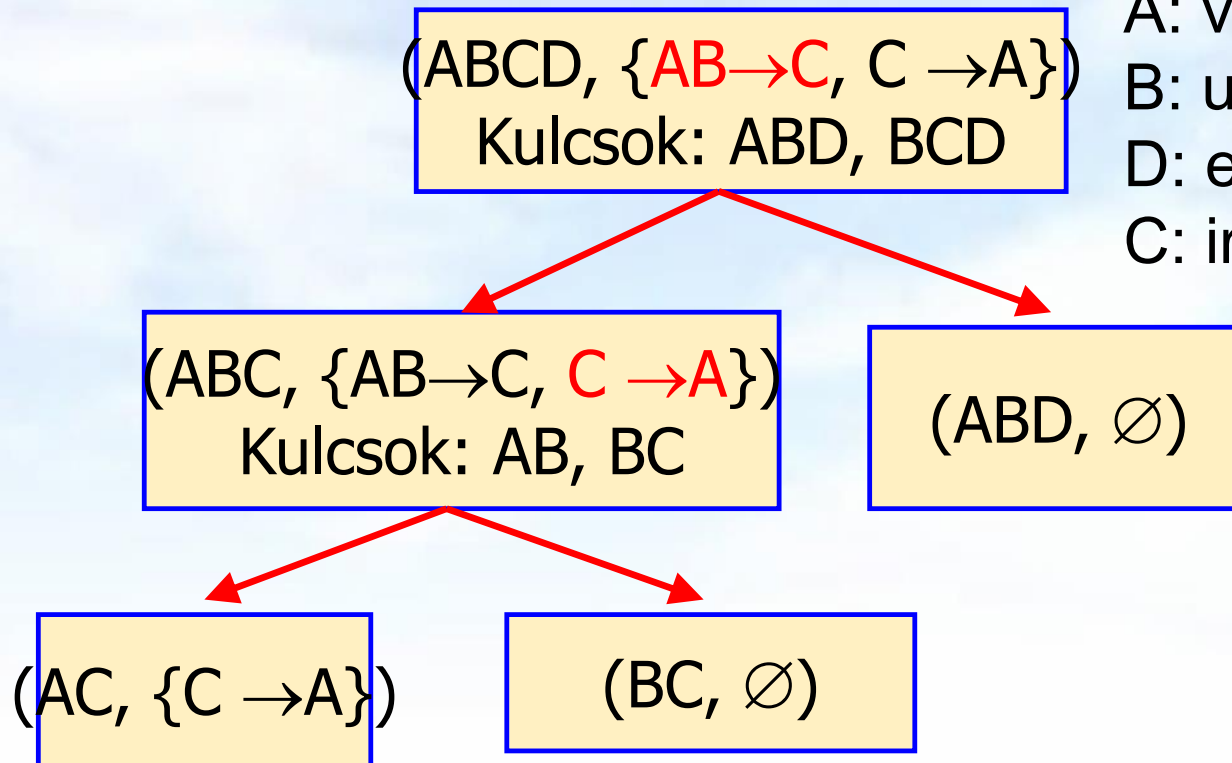
Példa: BCNF dekompozíció ---4

- Az *Sörivók* dekompozíciója tehát:
 1. *Sörivók1*(név, cím, kedvencSör)
 2. *Sörivók 3*(kedveltSörök, gyártó)
 3. *Sörivók 4*(név, kedveltSörök)
- A *Sörivók1* a sörivókról, a *Sörivók3* a sörökről, az *Sörivók4* a sörivók és kedvelt söreikről tartalmaz információt.

Példa: BCNF-ra való felbontás

$R=ABCD, F=\{AB \rightarrow C, C \rightarrow A\}$

Például R: lakcím
A: város, kerület
B: utca, házszám
D: emelet, ajtó
C: irányítószám



Tehát $d=(AC,BC,ABD)$ veszteségmentes BCNF dekompozíció.
(\emptyset azt jelenti, hogy csak a triviális függőségek teljesülnek a sémában.)

Tankönyv 3.5.2. feladata (111.o.)

- **Órarend adatbázis:** Kurzus(**K**), Oktató(**O**), Időpont(**I**), Terem(**T**), Diák(**D**), Jegy(**J**)
- **Feltételek:**
 - Egy kurzust csak egy oktató tarthat: $K \rightarrow O$.
 - Egy helyen egy időben egy kurzus lehet: $IT \rightarrow K$.
 - Egy időben egy tanár csak egy helyen lehet: $IO \rightarrow T$.
 - Egy időben egy diák csak egy helyen lehet: $ID \rightarrow T$.
 - Egy diák egy kurzust egy végső jeggyel zár: $KD \rightarrow J$.
- **R=KOITDJ** $F = \{K \rightarrow O, IT \rightarrow K, IO \rightarrow T, ID \rightarrow T, KD \rightarrow J\}$
- **Feladat:** Adjuk meg az algoritmussal egy BCNF dekompozícióját!

(Tk.3.4.) Felbontásra vonatkozó elvárások

- (1) **Anomáliák kiküszöbölése dekompozícióval**, és a vetületek legyenek jó tulajdonságúak, a vetületi függőségi rendszere egyszerű legyen (normálformák: BCNF, 3NF, 4NF)
- (2) **Veszteségmentes** legyen a felbontás, vagyis vissza tudjuk állítani az eredeti relációt a dekompozícióval kapott relációk soraiból.
- (3) **Függőségek megőrzése** a vetületekben (FŐ)
 - **BCNF-ra való felbontás algoritmus**
 - mindig veszteségmentes felbontást ad
 - De nem feltétlen függőségőrző a felbontás

(Tk.3.4.1.) Veszteségmentes szétvágás

- A fenti jelölésekkel: ha $r = \Pi_{R_1}(r) \bowtie \dots \bowtie \Pi_{R_k}(r)$ teljesül, akkor az előbbi összekapcsolásra azt mondjuk, hogy **veszteségmentes**.

(Itt r egy R sémájú reláció-előfordulást jelöl).

R

A	B	C
a	b	c
d	e	f
c	b	c

R_1

A	B
a	b
d	e
c	b

R_2

B	C
b	c
e	f

Egy példa: Nem veszteségmentes

- **Megjegyzés:** Könnyen belátható, hogy $r \subseteq \Pi_{R_1}(r) \bowtie \dots \bowtie \Pi_{R_k}(r)$ mindig teljesül.
- **Példa:** Vizsgáljuk meg $R(A,B,C)$ relációt, amelyre a $B \rightarrow A$ és $B \rightarrow C$ függőségek egyike sem teljesül. A szétvágás után keletkező relációk természetes összekapcsolásával 4 sort, vagyis két hamis sort kapunk, amelyek nem voltak az eredeti R -ben.

R

A	B	C
a	b	c
c	b	e

R_1

A	B
a	b
c	b

R_2

B	C
b	c
b	e

(Tk.3.4.2.) Chase-teszt VM ellenőrzése

- Példa: adott $R(A, B, C, D)$, $F = \{ A \rightarrow B, B \rightarrow C, CD \rightarrow A \}$ és az $R_1(A, D)$, $R_2(A, C)$, $R_3(B, C, D)$ felbontás. Kérdés veszteségmentes-e a felbontás?
- Vegyük $R_1 \bowtie R_2 \bowtie R_3$ egy $t = (a, b, c, d)$ sorát. Bizonyítani kell, hogy $t \in R$ egy sora. A következő tablót készítjük:

A	B	C	D
a	b_1	c_1	d
a	b_2	c	d_2
a_3	b	c	d

Itt pl. az (a, b_1, c_1, d) sor azt jelzi, hogy R -nek van olyan sora, aminek R_1 -re való levetítése (a, d) , ám ennek a B és C attribútumokhoz tartozó értéke ismeretlen, így egyáltalán nem biztos, hogy a t sorról van szó.

Chase-teszt VM ellenőrzése ---2

- Az F-beli függőségeket használva egyenlővé tesszük azokat a szimbólumokat, amelyeknek ugyanazoknak kell lennie, hogy valamelyik függőség ne sérüljön.
 - Ha a két egyenlővé teendő szimbólum közül az egyik index nélküli, akkor a másik is ezt az értéket kapja.
 - Két indexes szimbólum esetén a kisebbik indexű értéket kapja meg a másik.
 - A szimbólumok minden előfordulását helyettesíteni kell az új értékkel.
- Az algoritmus véget ér, ha valamelyik sor t-vel lesz egyenlő, vagy több szimbólumot már nem tudunk egyenlővé tenni.

Chase-teszt VM ellenőrzése ---3

A	B	C	D
a	b ₁	c ₁	d
a	b ₂	c	d ₂
a ₃	b	c	d

$A \rightarrow B$



A	B	C	D
a	b ₁	c ₁	d
a	b ₁	c	d ₂
a ₃	b	c	d

$B \rightarrow C$



A	B	C	D
a	b ₁	c	d
a	b ₁	c	d ₂
a ₃	b	c	d

$CD \rightarrow A$




A	B	C	D
a	b ₁	c	d
a	b ₁	c	d ₂
a	b	c	d

Chase-teszt VM ellenőrzése ---4

- Ha t szerepel a tablóban, akkor valóban R -nek egy sora, s mivel t -t tetszőlegesen választottuk, ezért a felbontás veszteségmentes.
- Ha nem kapjuk meg t -t, akkor viszont a felbontás nem veszteségmentes.
- **Példa:** $R(A, B, C, D)$, $F = \{ B \rightarrow AD \}$,
a felbontás: $R_1(A, B)$, $R_2(B, C)$, $R_3(C, D)$.

A	B	C	D
a	b	c_1	d_1
a_2	b	c	d_2
a_3	b_3	c	d

$B \rightarrow AD$



A	B	C	D
a	b	c_1	d_1
a	b	c	d_1
a_3	b_3	c	d

Itt az eredmény jó ellenpélda, hiszen az összekapcsolásban szerepel $t = (a, b, c, d)$, míg az eredeti relációban nem.

(Tk.3.4.4.) Függőségek megőrzése függőségőrző felbontás

- **Függőségőrző felbontás:** a dekompozíciókban érvényes függőségekből következzen az eredeti sémára kirótt összes függőség, vagyis
- **Definíció:** Adott (R, F) séma esetén $d=(R_1, \dots, R_k)$ felbontás akkor és csak akkor **függőségőrző**, ha minden F -beli függőség levezethető a vetületi függőségekből:

minden $X \rightarrow Y \in F$ esetén

$$\Pi_{R_1}(F) \cup \dots \cup \Pi_{R_k}(F) \vdash X \rightarrow Y$$

Emlékeztető: Függőségőrző felbontás

Itt mit értünk függőségek vetületén?

(Tk.3.2.8.) Függőségek vetítése

- **Definíció: Funkcionális függőségek vetítése**

Adott (R, F) , és $R_i \subseteq R$ esetén:

$$\Pi_{R_i}(F) := \{ X \rightarrow Y \mid F \vdash X \rightarrow Y, XY \subseteq R_i \}$$

- **Példa-1: $R=ABC$, $A \rightarrow B$ és $B \rightarrow C$ FF-kel.**
Nézzük meg az $R_1=AC$ -re való vetületet

Példa-1: Függőségek vetítése

- ABC , $A \rightarrow B$ és $B \rightarrow C$ FF-kel. Nézzük meg az AC -re való vetületet:
 - $A^+ = ABC$; ebből $A \rightarrow B$, $A \rightarrow C$.
 - Nem kell kiszámítani AB^+ és AC^+ lezárásokat.
 - $B^+ = BC$; ebből $B \rightarrow C$.
 - $C^+ = C$; semmit nem ad.
 - $BC^+ = BC$; semmit nem ad.
- A kapott FF-ek: $A \rightarrow B$, $A \rightarrow C$ és $B \rightarrow C$.
- AC -re projekció: $A \rightarrow C$.

Példa-2: Függőségőrző felbontás

- Milyen függőségek lesznek érvényesek a dekompozíció sémáiban?
- **Példa:** $R=ABC$, $F= \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$ vajon a $d= (AB, BC)$ felbontás megőrzi-e a $C \rightarrow A$ függőséget?
- AB -re projekció: $A \rightarrow B$ és $B \rightarrow A$ is.
- BC -re projekció: $B \rightarrow C$ és $C \rightarrow B$ is.
- $\Pi_{R_1}(F) \cup \dots \cup \Pi_{R_k}(F) \mid\!\!-\! C \rightarrow A$.

Függőségőrzés - veszteségmentesség

- A függőségőrzésből nem következik a veszteségmentesség:

$R=ABCD$, $F= \{A \rightarrow B, C \rightarrow D\}$, $d=\{AB, CD\}$
függőségőrző, de nem veszteségmentes.

- A veszteségmentességből nem következik a függőségőrzés

$R=ABC$, $F= \{AB \rightarrow C, C \rightarrow B\}$, $d=\{AC, BC\}$
veszteségmentes, de nem függőségőrző.

(Tk.3.5.) A harmadik normálforma motiváció

- Bizonyos FF halmazok esetén a felbontáskor elveszíthetünk függőségeket.
- $AB \rightarrow C$ és $C \rightarrow B$.
 - Példa: $A =$ utca, $B =$ város, $C =$ irányítószám.
- Két kulcs van: $\{A, B\}$ és $\{A, C\}$.
- $C \rightarrow B$ megsérti a BCNF-t, tehát AC , BC -re dekomponálunk.
- A probléma az, hogy AC és BC sémákkal nem tudjuk kikényszeríteni $AB \rightarrow C$ függőséget.

A harmadik normálforma (definíció)

- A harmadik normálformában (3NF) úgy módosul a BCNF feltétel, hogy az előbbi esetben már nem kell dekomponálnunk (megőrizzük a függőséget)
- **Definíció:** Egy attribútum **elsődleges attribútum (prímattribútum)**, ha legalább egy kulcsnak eleme.
- **Definíció:** Az **R reláció 3NF-ban van** akkor és csak akkor, ha valahányszor létezik egy **$X \rightarrow A$** nem-triviális függőség, akkor
 - vagy **X superkulcs**
 - vagy **A prímattribútum.**

Példa: 3NF

- Az előző példában $AB \rightarrow C$ és $C \rightarrow B$ FF-ek esetén a kulcsok AB és AC .
- Ezért A , B és C mindegyike **prím**.
- Habár $C \rightarrow B$ megsérti a BCNF feltételét, de a 3NF feltételét már nem sérti meg.

A 3NF és BCNF felbontások

- A dekompozícióknak két fontos tulajdonsága:
 1. **Veszteségmentes összekapcsolás**: ha a részsémákra vetített relációkból természetes összekapcsolással az eredetit kapjuk vissza.
 2. **Függőségek megőrzése**: a vetített relációk segítségével is kikényszeríthetőek az előre megadott függőségek.
- Az (1) tulajdonság teljesül a BCNF esetében.
- A 3NF (1) és (2)-t is teljesíti.
- A BCNF esetén (2) sérülhet (*utca-város-irszám*)

3NF-szintetizáló algoritmus

- Hogyan bontható fel egy R reláció olyan relációk halmazává, amelyekre teljesülnek az alábbiak:
 - A felbontásban szereplő relációk mindegyike 3NF-ban van
 - A felbontásnak veszteségmentes összekapcsolása van
 - A felbontásnak függőségmegőrző tulajdonsága van.
- Az algoritmushoz ismételjük át a minimális bázis fogalmát és előállítását, mert erre lesz szükség!

(Tk.3.2.7.) Minimális bázis (definíció)

Amikor egy relációhoz választhatunk, hogy milyen ekvivalens funkcionális függőségi hz-t használunk, amelyek reprezentálják egy reláció teljes f.f. hz-t. Az F minimális bázis, ha az olyan f.f.-ekből áll, amelyre három („minimális”) feltétel igaz:

1. F összes függőségének jobb oldalán egy attribútum van.
2. Ha bármelyik F -beli függőséget elhagyjuk, a fennmaradó halmaz már nem bázis.
3. Ha bármelyik F -beli funkcionális függőség bal oldaláról elhagyunk egy vagy több attribútumot, akkor az eredmény már nem marad bázis.

Minimális bázist kiszámító algoritmus

1.) Kezdetben G az üreshalmaz.

Minden $X \rightarrow Y \in F$ helyett vegyük az $X \rightarrow A$ funkcionális függőségeket, ahol $A \in Y - X$).

Megj.: Ekkor minden G -beli függőség $X \rightarrow A$ alakú.

2.) Minden $X \rightarrow A \in G$ -re, ha $X \rightarrow A \in (G - \{X \rightarrow A\})^+$, vagyis ha elhagyjuk az $X \rightarrow A$ függőséget G -ből, az még mindig következik a maradékból, akkor $G := G - \{X \rightarrow A\}$.

Megj.: Elhagyható függőségeket mind elhagyjuk.

3.) Minden $X \rightarrow A \in G$ -re, amíg van olyan $B \in X$ -re $A \in (X - B)^+$ a G -szerint, vagyis $(X - B) \rightarrow A$ teljesül, akkor $X := X - B$.

Megj.: E lépés után minden baloldal minimális lesz.

Mohó algoritmus minimális bázis előállítására

1. Jobb oldalak minimalizálása:

$X \rightarrow A_1, \dots, A_k$ függőséget cseréljük le az
 $X \rightarrow A_1, \dots, X \rightarrow A_k$ k darab függőségre.

2. A halmaz minimalizálása:

Hagyjuk el az olyan $X \rightarrow A$ függőségeket, amelyek a bázist nem befolyásolják, azaz

while F változik

if $(F - \{X \rightarrow A\})^* = F^*$ then $F := F - \{X \rightarrow A\}$;

3. Bal oldalak minimalizálása:

Hagyjuk el a bal oldalakból azokat az attribútumokat, amelyek a bázist nem befolyásolják, azaz

while F változik

for all $X \rightarrow A \in F$

for all $B \in X$

if $((F - \{X \rightarrow A\}) \cup \{(X - \{B\}) \rightarrow A\})^* = F^*$ then $F := (F - \{X \rightarrow A\}) \cup \{(X - \{B\}) \rightarrow A\}$

Példa: minimális bázis kiszámítása

- Az algoritmusban különböző sorrendben választva a függőségeket, illetve attribútumokat, különböző minimális bázist kaphatunk.
- $F = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, A \rightarrow C, C \rightarrow A\}$
 $(F - \{B \rightarrow A\})^* = F^*$, mivel $F - \{B \rightarrow A\} \vdash B \rightarrow A$
 $F := F - \{B \rightarrow A\}$
 $(F - \{A \rightarrow C\})^* = F^*$, mivel $F - \{A \rightarrow C\} \vdash A \rightarrow C$
 $F := F - \{A \rightarrow C\} = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$ **minimális bázis**, mert több függőség és attribútum már nem hagyható el.
- $F = \{A \rightarrow B, B \rightarrow A, B \rightarrow C, A \rightarrow C, C \rightarrow A\}$
 $(F - \{B \rightarrow C\})^* = F^*$, mivel $F - \{B \rightarrow C\} \vdash B \rightarrow C$
 $F := F - \{B \rightarrow C\} = \{A \rightarrow B, B \rightarrow A, A \rightarrow C, C \rightarrow A\}$ **is minimális bázis**, mert több függőség és attribútum már nem hagyható el.

Példa: minimális bázis kiszámítása

- Az algoritmusban különböző sorrendben választva a függőségeket, illetve attribútumokat, különböző minimális bázist kaphatunk.
- $F = \{AB \rightarrow C, A \rightarrow B, B \rightarrow A\}$
 $(F - \{AB \rightarrow C\} \cup \{A \rightarrow C\})^* = F^*$, mivel
 $(F - \{AB \rightarrow C\}) \cup \{A \rightarrow C\} \vdash AB \rightarrow C$ és $F \vdash A \rightarrow C$.
 $F := (F - \{AB \rightarrow C\} \cup \{A \rightarrow C\}) = \{A \rightarrow C, A \rightarrow B, B \rightarrow A\}$ **minimális bázis**, mert több függőség és attribútum már nem hagyható el.
- $F = \{AB \rightarrow C, A \rightarrow B, B \rightarrow A\}$
 $(F - \{AB \rightarrow C\} \cup \{B \rightarrow C\})^* = F^*$, mivel
 $(F - \{AB \rightarrow C\}) \cup \{B \rightarrow C\} \vdash AB \rightarrow C$ és $F \vdash B \rightarrow C$.
 $F := (F - \{AB \rightarrow C\} \cup \{B \rightarrow C\}) = \{B \rightarrow C, A \rightarrow B, B \rightarrow A\}$ **is minimális bázis**, mert több függőség és attribútum már nem hagyható el.

3NF-szintetizáló algoritmus ---1

- Algoritmus **függőségőrző 3NF dekompozíció** előállítására:
- Input: (R, F)
 - Legyen $G := \{X \rightarrow A, X \rightarrow B, \dots, Y \rightarrow C, Y \rightarrow D, \dots\}$ az **F minimális bázisa**.
 - Legyen **S** az R sémának G-ben nem szereplő attribútumai. Ha van olyan függőség G-ben, amely R összes attribútumát tartalmazza, akkor legyen $d := \{R\}$, különben legyen $d := \{S, XA, XB, \dots, YC, YD, \dots\}$ (így ez egy felbontás).
 - Amennyiben a fenti relációk attribútumhalmazának egyike sem szuperkulcs R-ben, adjunk még egy relációt az eredményhez, melynek sémája **K legyen az R egy kulcsa**, és legyen $d := \{K, S, XA, XB, \dots, YC, YD, \dots\}$ (így VM felbontás)

3NF-szintetizáló algoritmus ---2

- Algoritmus függőségörző és veszteségmentes 3NF redukált (kevesebb tagból álló) dekompozíció előállítására:
- Input: (R, F)
 - Legyen $G := \{X \rightarrow A, X \rightarrow B, \dots, Y \rightarrow C, Y \rightarrow D, \dots\}$ az F minimális bázisa.
 - Ha van olyan függőség G -ben, amely R összes attribútumát tartalmazza, akkor legyen $d := \{R\}$.
 - Legyen S az R sémának G -ben nem szereplő attribútumai. Legyen K az R egy kulcsa, és legyen $d := \{K, S, XAB \dots, \dots, YCD \dots, \dots\}$.
 - Ha K része valamelyik sémának, akkor K -t elhagyhatjuk.

Miért működik?

3NF-szintetizáló algoritmus:

- **Veszteségmentes összekapcsolás:** a Chase algoritmussal ellenőrizhető (a kulcsból létrehozott séma itt lesz fontos).
- **Függőségőrzés:** minden FF megmarad a minimális bázisból.
- **3NF:** a minimális bázis tulajdonságaiból következik.

Kérdés / Válasz

- **Köszönöm a figyelmet! Kérdés/Válasz?**
- **Feladatok:** Tankönyv 3.fejezetében az egyes szakaszok után található gyakorló feladatok