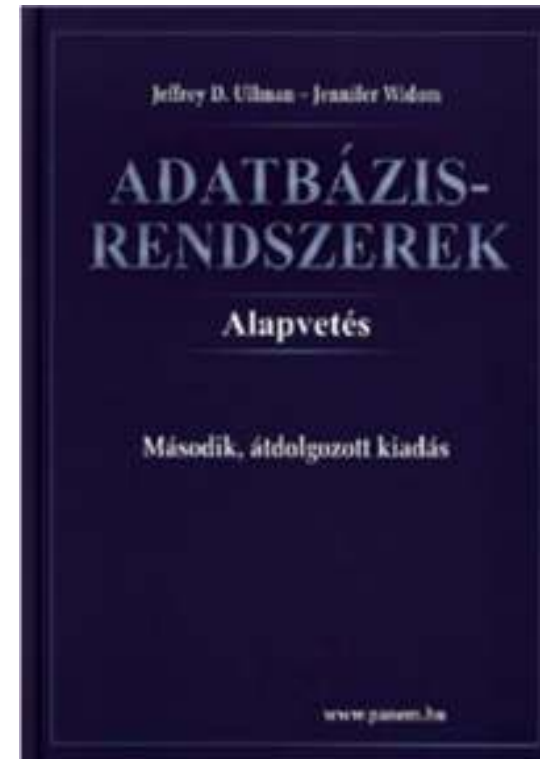


Relációs adatbázisok tervezése

3.rész Normálformák

Tankönyv: Ullman-Widom:
Adatbázisrendszerek Alapvetés
Második, átdolgozott kiadás,
Panem, 2009

3.3. Relációs adatbázissémák
tervezése, Boyce-Codd (BCNF)
3.5. Harmadik normálforma (3NF)



Boyce-Codd normálforma

- **Definíció:** R reláció **Boyce-Codd normálformában, BCNF-ban van**, ha
 - minden $X \rightarrow Y$ **nemtriviális FF-re** R -ben (nemtriviális, vagyis Y nem része X -nek)
 - az X **szuperkulcs** (szuperkulcs, vagyis tartalmaz kulcsot).

Példa

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 BCNF normálformában.

Még egy példa

Sörök(név, gyártó, gyártóCím)

FF-ek: név->gyártó, gyártó->gyártóCím

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

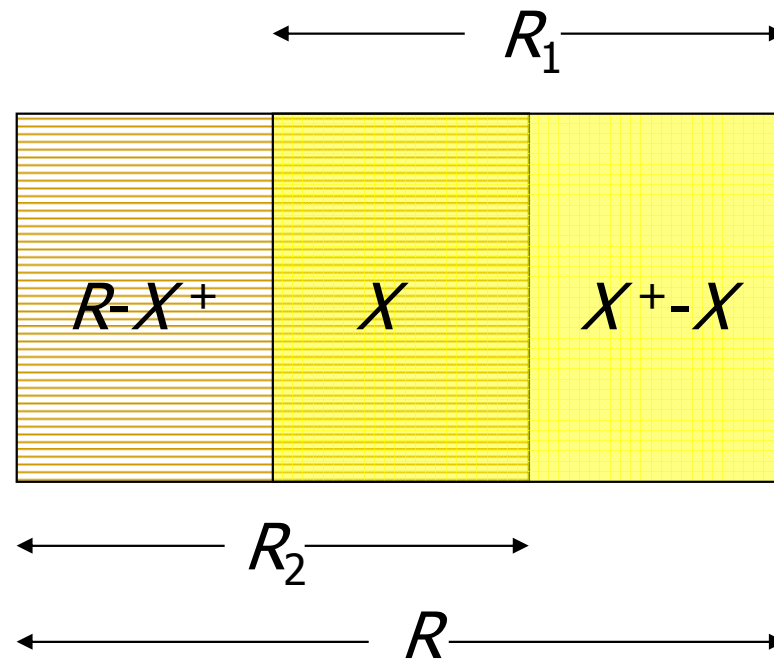
BCNF-re való felbontás ---1

- Adott R reláció és F funkcionális függőségek.
- Van-e olyan $X \rightarrow Y$ FF, ami sérti a BCNF-t?
 - Ha van olyan következmény FF F -ben, ami sérti a BCNF-t, akkor egy F -beli FF is sérti.
- Kiszámítjuk X^+ -t:
 - Ha itt nem szerepel az összes attribútum, X nem superkulcs.

BCNF-re való felbontás ---2

- R -t helyettesítsük az alábbiakkal:
 1. $R_1 = X^+$.
 2. $R_2 = R - (X^+ - X)$.
- *Projektáljuk* a meglévő F -beli FF-eket a két új relációsémára.

Dekomponálási kép



Példa: BCNF dekompozíció ---1

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

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

- Vegyük $\text{név} \rightarrow \text{cím}$ 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

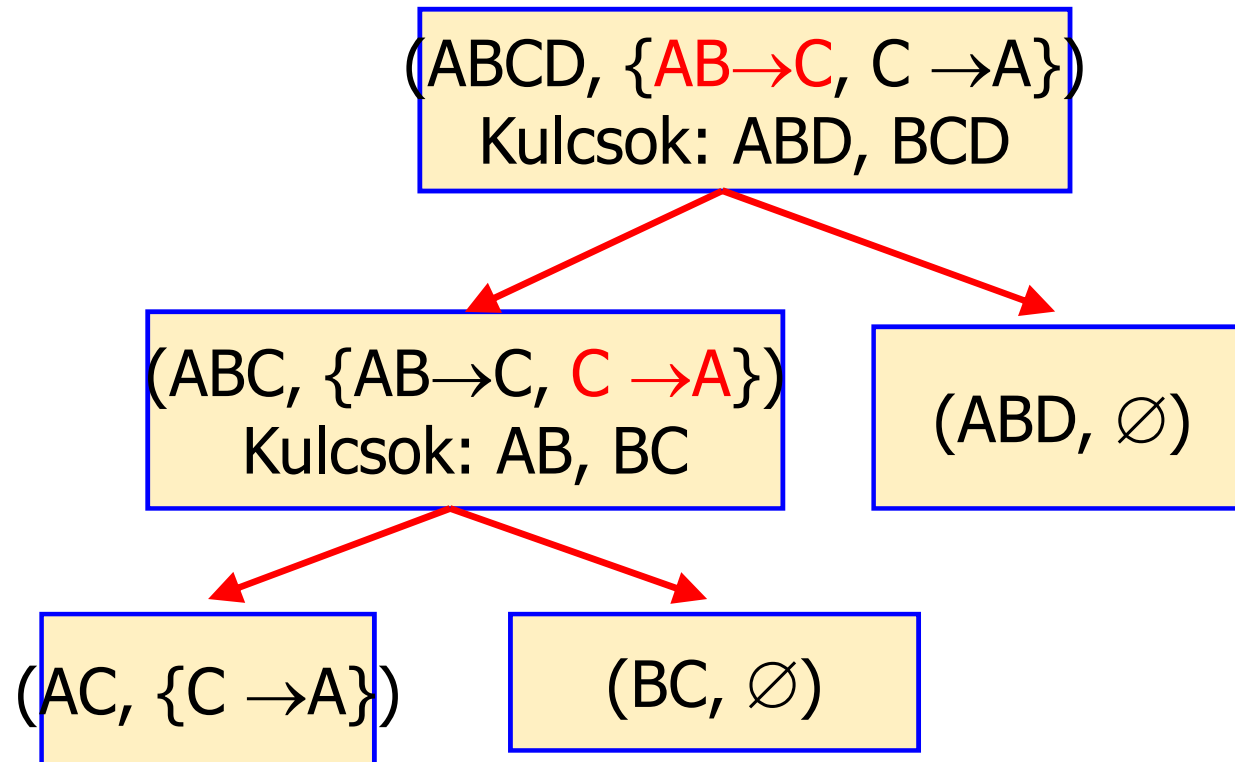
- Az *Sörivók* dekompozíciója tehát:
 1. *Sörivók1*(név, cím, kedvencSör)
 2. *Sörivók 3*(kedveltSörök, gyártó)
 3. *Sörivók 4*(név, kedveltSörök)
- A *Sörivók1* a sörivókról, a *Sörivók3* a sörökről, az *Sörivók4* a sörivók és kedvelt söreikről tartalmaz információt.

Miért működik a BCNF?

- (R, F) esetén ha R_1, \dots, R_k egy veszteségmentes felbontás, S_1, S_2 pedig R_1 veszteségmentes felbontása, akkor $S_1, S_2, R_2, \dots, R_k$ is veszteségmentes felbontás.
- Könnyen ellenőrizhető, hogy a fenti R_1, R_2 veszteségmentes. (Miért? Chase-teszttel)
- **Feladat-1:** Minden két attribútumú séma BCNF-ban van (mert nincs olyan FF, ami sértené a BCNF definíciót)
- **Feladat-2:** Bizonyítsuk be, hogy ha a fenti $R(A, B, C)$ reláció esetén $B \rightarrow C$ teljesül, akkor az $R_1(A, B), R_2(B, C)$ felbontás mindig veszteségmentes.
- Az algoritmus tehát valóban veszteségmentes felbontást ad, ám sajnos **exponenciális** lépésszámú is lehet a függőségek vetítése miatt.

Példa: BCNF-ra való felbontás

R=ABCD, **F={AB→C, C→A}**



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!

A 3normálforma -- motiváció

- Bizonyos FF halmazok esetén a felbontáskor elveszíthetünk függőségeket.
- $AB \rightarrow C$ és $C \rightarrow B$.
 - Példa1: $A =$ utca, $B =$ város, $C =$ irányítószám.
 - Példa2: $A =$ oktató, $B =$ időpont, $C =$ kurzus.
- 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 probléma megoldása: 3NF

- 3. normálformában (3NF) úgy módosul a BCNF feltétel, hogy az előbbi esetben nem kell dekomponálnunk.
- Egy attribútum **elsődleges attribútum (prím)**, ha legalább egy kulcsnak eleme.
- $X \rightarrow A$ megsérti 3NF-t akkor és csak akkor, ha X nem szuperkulcs és A nem prím.

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.

Miért hasznos 3NF és BCNF?

- A dekompozícióknak két fontos tulajdonsága lehet:
 1. *Veszteségmentes összekapcsolás* : ha a projektált relációkat összekapcsoljuk az eredetit kapjuk vissza.
 2. *Függőségek megőrzése* : a projektált relációk segítségével is kikényszeríthetők 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*)

Minimális bázist kiszámító algoritmus

1. Kezdetben G az üreshalmaz.
2. Minden $X \rightarrow Y \in F$ helyett vegyük az $X \rightarrow A$ függőségeket, ahol $A \in Y - X$.
Megj.: Ekkor minden G -beli függőség $X \rightarrow A$ alakú.
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 $X := X - B$.
Megjegyzés: E lépés után minden baloldal minimális lesz.
4. 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 $G := G - \{X \rightarrow A\}$.
Megjegyzés: Végül nem marad több elhagyható függőség

Normálformák (3NF)

- **Definíció:** $F^* := \{X \rightarrow Y \mid F \vdash X \rightarrow Y\}$ F-ből levezethető összes függőség (F levezethetőség szerinti lezártja).
- Nézzük meg, hogy lehet minimálisra csökkenteni egy F függőségi halmazt.
- G az F **minimális bázisa** (másképpen minimális fedése), ha
 - $F^* = G^*$ (**bázis**, vagy másképpen fedés),
 - G minden függőségében a jobb oldalak egyeleműek, (**jobb oldalak minimálisak**)
 - G-ből nem hagyható el függőség, hogy F bázisa maradjon, (**minimális halmaz**)
 - G függőségeinek bal oldala nem csökkenthető, hogy F bázisa maradjon (**bal oldalak minimálisak**).

Normálformák (3NF)

- **Kérdések:**
 - Minden F-nek létezik minimális bázisa?
 - Egyértelmű-e a minimális bázis?
 - Hogyan lehet adott F esetén meghatározni egy minimális bázist, lehetőleg hatékonyan?

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 $F := (F - \{X \rightarrow A\}) \cup \{(X - \{B\}) \rightarrow A\}$

Normálformák (3NF)

- 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.

Normálformák (3NF)

- 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.

Normálformák (3NF)

- 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
 $d := \{S, XA, XB, \dots, YC, YD, \dots\}$.

Normálformák (3NF)

- Algoritmus függőségörző és **veszteségmentes** 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 K az R egy kulcsa**, és legyen $d := \{K, S, XA, XB, \dots, YC, YD, \dots\}$.

Normálformák (3NF)

- 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.
 - 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 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:

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