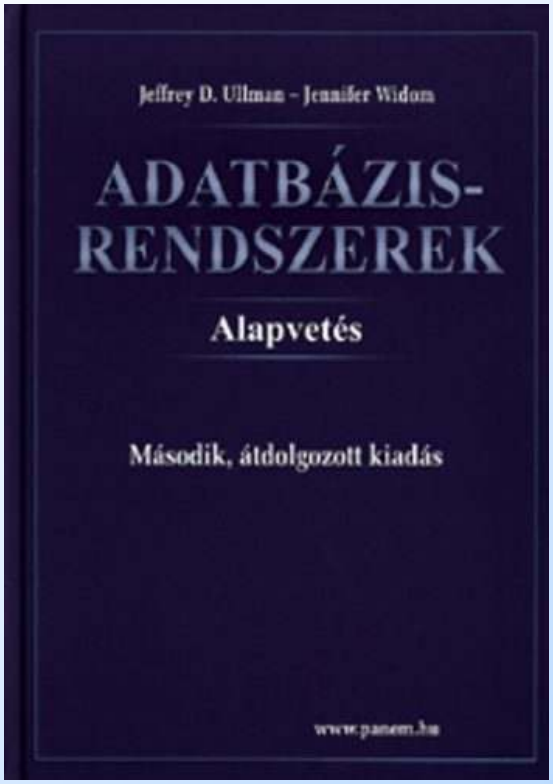
Relációs adatbázisok tervezése
4.rész (3NF szintetizáló algoritmus)



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

3.4.4. Függőségek megőrzése
3.5. Harmadik normálforma

(Jeffrey D. Ullman, 2007 slides alapján
Dr. Kiss Attila előadásainak felhasználásával)

Third Normal Form -- Motivation

- There is one structure of FD's that causes trouble when we decompose.
- *AB* -> *C* and *C* -> *B*.
  - Example: *A* = street address, *B* = city, *C* = zip code.
- There are two keys, {*A*,*B*} and {*A*,*C*}.
- *C* -> *B* is a BCNF violation, so we must decompose into *AC*, *BC*.

We Cannot Enforce FD's

- The problem is that if we use *AC* and *BC* as our database schema, we cannot enforce the FD *AB -> C* by checking FD's in these decomposed relations.
- Example with *A* = street, *B* = city, and *C* = zip on the next slide.

An Unenforceable FD

street	zip
545 Tech Sq.	02138
545 Tech Sq.	02139

city	zip
Cambridge	02138
Cambridge	02139

Join tuples with equal zip codes.

street	city	zip
545 Tech Sq.	Cambridge	02138
545 Tech Sq.	Cambridge	02139

Although no FD's were violated in the decomposed relations, FD street city -> zip is violated by the database as a whole.

3NF Let's Us Avoid This Problem

- 3rd Normal Form (3NF) modifies the BCNF condition so we do not have to decompose in this problem situation.
- An attribute is *elsődleges attribútum (prime)* if it is a member of any key.
- *X* -> *A* violates 3NF if and only if *X* is not a superkey, and also *A* is not prime.

Example: 3NF

- In our problem situation with FD's *AB* -> *C* and *C* -> *B*, we have keys *AB* and *AC*.
- Thus *A*, *B*, and *C* are each prime.
- Although *C* -> *B* violates BCNF, it does not violate 3NF.

Normálformák (3NF)

- Adott (R,F) esetén $A \in R$ az R elsődleges attribútuma F-re nézve, ha A szerepel az R valamelyik F-re vonatkoztatott kulcsában.
- A 3NF több redundanciát enged meg, mint a BCNF.
- A következő két definíció ekvivalens.
- 1. Definíció. R relációséma 3. normálformában (3NF-ben) van az F-re nézve, ha tetszőleges $XA \subseteq R$, $A \notin X$ és $F \vdash X \rightarrow A$ esetén $F \vdash X \rightarrow R$ (azaz X az R szuperkulcsa F-re nézve) vagy A az R elsődleges attribútuma F-re nézve.

- 2. Definíció. R relációséma 3. normálformában (3NF-ben) van az F-re nézve, ha tetszőleges $XA \subseteq R$, $A \notin X$ és $X \rightarrow A \in F$ esetén $F \vdash X \rightarrow R$ (azaz X az R szuperkulcsa F-re nézve) vagy A az R elsődleges attribútuma F-re nézve.

Normálformák (3NF)

- Adott (R,F) és d=(R1,...,Rk) esetén d az R 3NF dekompozíciója, ha minden i-re Ri 3NF a Π_{Ri}(F)-re nézve.

- R=ABC, F={AB→C,C→A}

  R kulcsai: AB, BC

  Láttuk, hogy nincs BCNF-ben, és nincs függőségőrző BCNF dekompozíciója.

- R elsődleges attribútumai: A,B,C, így R 3NF-ben van F-re nézve, tehát nem kell tovább dekomponálni.

- Következmény: Mivel a definíciót gyengítettük, így ha (R,F) BCNF, akkor (R,F) 3NF is.

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

Normálformák (3NF)

Mohó algoritmus minimális bázis előállítására:

1. **Jobb oldalak minimalizálása:**

   $X \to A1,\ldots,Ak$ függőséget cseréljük le az

   $X \to A1, \ldots, X \to Ak$ k darab függőségre.

2. **A halmaz minimalizálása:**

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

   while F változik

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

3. **Bal oldalak minimalizálása:**

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

   while F változik

   for all $X \to A \in F$

   for all $B \in X$

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

- Belátható, hogy a 3. lépés nem rontja el a halmaz minimalizálást, így minimális bázist kapunk.

Normálformák (3NF)

- Az algoritmusban különböző sorrendben választva a függőségeket, illetve attribútumokat, különböző minimális bázist kaphatunk.

- F={A→B, B→A, B→C, A→C, C→A}
  (F-{B→A})*=F*, mivel F-{B→A} |— B→A
  F:=F-{B→A}
  (F-{A→C})*=F*, mivel F-{A→C} |— A→C
  F:=F-{A→C}= {A→B,B→C,C→A} minimális bázis, mert több függőség és attribútum már nem hagyható el.

- F={A→B, B→A, B→C, A→C, C→A}
  (F-{B→C})*=F*, mivel F-{B→C} |— B→C
  F:=F-{B→C}={A→B,B→A,A→C,C→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→C, A→B, B→A}**

  (F-{AB→C}∪{A→C})*=F*, mivel

  (F-{AB→C})∪{A→C} |— AB→C és F |— A→C.

  F:=(F-{AB→C}∪{A→C})= {A→C,A→B,B→A} minimális bázis, mert több függőség és attribútum már nem hagyható el.

- **F={AB→C, A→B, B→A}**

  (F-{AB→C}∪{B→C})*=F*, mivel

  (F-{AB→C})∪{B→C} |— AB→C és F |— B→C.

  F:=(F-{AB→C}∪{B→C})= {B→C,A→B,B→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→A,X→B,...,Y→C,Y→D,....} 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,...,YC,YD,...}.

Normálformák (3NF)

- **Algoritmus függőségőrző és veszteségmentes 3NF dekompozíció előállítására:**
- **Input: (R,F)**
  - **Legyen G:={X→A,X→B,...,Y→C,Y→D,....} 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,...,YC,YD,...}.

Normálformák (3NF)

- **Algoritmus függőségőrző és veszteségmentes 3NF redukált (kevesebb tagból álló) dekompozíció előállítására:**
- **Input: (R,F)**
  - **Legyen G:={X→A,X→B,...,Y→C,Y→D,....} 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...,...,YCD...,...}.
  - **Ha K része valamelyik sémának, akkor K-t elhagyhatjuk.**

What 3NF and BCNF Give You

- There are two important properties of a decomposition:
  1. *Lossless Join* : it should be possible to project the original relations onto the decomposed schema, and then reconstruct the original.
  2. *Dependency Preservation* : it should be possible to check in the projected relations whether all the given FD's are satisfied.

3NF and BCNF -- Continued

- We can get (1) with a BCNF decomposition.
- We can get both (1) and (2) with a 3NF decomposition.
- But we can't always get (1) and (2) with a BCNF decomposition.
  - street-city-zip is an example.

3NF Synthesis Algorithm

- We can always construct a decomposition into 3NF relations with a lossless join and dependency preservation.
- Need *minimal basis* for the FD's:
  1. Right sides are single attributes.
  2. No FD can be removed.
  3. No attribute can be removed from a left side.

Constructing a Minimal Basis

1. Split right sides.
2. Repeatedly try to remove an FD and see if the remaining FD's are equivalent to the original.
3. Repeatedly try to remove an attribute from a left side and see if the resulting FD's are equivalent to the original.

3NF Synthesis – (2)

- One relation for each FD in the minimal basis.
  - Schema is the union of the left and right sides.
- If no key is contained in an FD, then add one relation whose schema is some key.

Example: 3NF Synthesis

- Relation R = ABCD.
- FD's *A->B* and *A->C*.
- Decomposition: AB and AC from the FD's, plus AD for a key.

Why It Works

- Preserves dependencies: each FD from a minimal basis is contained in a relation, thus preserved.
- Lossless Join: use the chase to show that the row for the relation that contains a key can be made all-unsubscripted variables.
- 3NF: hard part – a property of minimal bases.

Tankönyv 3.5.2. feladata (111.o.)

- Órarend adatbázis: Kurzus(K), Oktató(O), Időpont(I), Terem(T), Diák(D), Csoport(C)
- Feltételek: …
- R=KOITDJ F= {K→O, IT→K, IO→T, ID→T, KD→C}
- Feladat: Igazoljuk, hogy a megadott funkcionális függőségek minimális bázist alkotnak! Használjuk a 3NF-szintetizáló algoritmust, hogy megtaláljuk R veszteségmentes összekapcsolási és függőségőrző tulajdonságú dekompozícióit 3NF-relációkra. Van-e ezek között olyan reláció, amelyik nincs BCNF-ban?