Relációs adatbbázis tervezés

Funkcionális függőségek

Felbontások

Normálformák

Funkcionális függőségek

- X -> Y az R relációra vonatkozó megszorítás, miszerint ha két sor megegyezik X összes attribútumán, Y attribútumain is meg kell, hogy egyezzenek.
  - Jelölés: X, Y, Z,… attribútum halmazokat; A, B, C,… attribútumokat jelöl.
  - Jelölés: {A,B,C} attribútum halmaz helyett ABC-t írunk.

Jobboldalak szétvágása (ff)

- X->$A_1A_2 \ldots A_n$ akkor és csak akkor teljesül R relációra, ha X->A_1, X->A_2,…, X->A_n is teljesül R-en.
- Példa: A->BC ekvivalens A->B és A->C függőségek kettősével.
- Baloldalak szétvágására nincs általános szabály.
- Általában FF-k jobboldalán egyetlen attribútum szerepel majd.

Példa: FF

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

- FF-k, amelyek vszleg teljesülnek:
  1. név -> cím kedvencSör
     - Ez az FF ugyanaz, mint név -> cím és név -> kedvencSör.
  2. kedveltSörök -> gyártó.

Példa: egy lehetséges előfordulás

név	cím	kedveltSörök	gyártó	kedvencSör
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

Mert név -> cím

Mert név -> kedvencSör

Mert kedveltSörök -> gyártó

Relációk kulcsai

- *K* *szuperkulcs* *R* relációra, ha *K* funkcionálisan meghatározza *R* attribútumait.
- *K* *kulcs* *R*-en, ha *K* szuperkulcs, de egyetlen valódi részhalmaza sem szuperkulcs.

Példa: szuperkulcs

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

- {név, kedveltSörök} szuperkulcs, hiszen a két attribútum meghatározza funkcionálisan a maradék attribútumokat.
  - név -> cím kedvencSör
  - kedveltSörök -> gyártó

Példa: kulcs

- {név, kedveltSörök} kulcs, hiszen sem {név}, sem {kedveltSörök} nem szuperkulcs.
  - név -> gyártó; kedveltSörök -> cím nem teljesülnek.
- Az előbbin kívül nincs több kulcs, de számos szuperkulcs megadható még.
  - Minden olyan halmaz, amit tartalmazza {név, kedveltSörök}-t.

Kis kombinatorika

- Feladat: R relációnak legyenek $A_1,\ldots, A_n$ az attribútumai. Adjuk meg n függvényeként, hogy R-nek hány szuperkulcsa van, ha

  (a) csak A_1 kulcs,

  (b) A_1 és A_2 kulcsok,

  (c) $\{A_1, A_2\}$, $\{A_3, A_4\}$ kulcsok,

  (d) $\{A_1, A_2\}$, $\{A_1, A_3\}$ kulcsok.

Hogyan kaphatjuk meg a kulcsokat?

1. Szimplán megadunk egy *K* kulcsot.
   - Az FF-k *K* -> *A* alakúak, ahol *A* „végigmegy” az összes attribútumon
2. Vagy: megadjuk az FF-ket, és ezekből következtetjük ki a kulcsokat.

Még egy természetesen adódó FF

- Példa: az „ugyanabban az időben nem lehet két előadás ugyanabban a teremben” lefordítva:

  idő terem -> előadás.

FF-k kikövetkeztetése

- Legyenek X_1 -> A_1, X_2 -> A_2,…, X_n -> A_n adott FF-ek, szeretnénk tudni, hogy Y -> B teljesül-e olyan relációkra, amire az előbbi FF-ek teljesülnek.
  - Példa: A -> B és B -> C teljesülése esetén A -> C biztosan teljesül.
- Ez az adatbázis sémájának megtervezésekor lesz majd fontos.

Armstrong-axiómák I.

- (A1) Reflexitivitás: ha $Y \subseteq X \subseteq R$, akkor $X \rightarrow Y$. Az ilyen függőségeket triviális függőségeknek nevezzük.
- (A2) Bővítés: ha $X \rightarrow Y$ teljesül, akkor tetszőleges $Z \subseteq R$-ra $XZ \rightarrow YZ$ teljesül.
- (A3) Tranzitivitás: ha $X \rightarrow Y$ és $Y \rightarrow Z$, akkor $X \rightarrow Z$.

- Példák a *személy (sz_ig_szám, TAJ, név, anyja_neve, születés, kor, fizetés)* tábla esetén:
  - (A1) (név, születés) → név
  - (A2) születés → kor, akkor (születés, név) → (kor, név)
  - (A3) TAJ → születés, születés → kor, akkor TAJ → kor.

Példa levezetésre

- Legyen R = ABCD és F = { $A \rightarrow C$, $B \rightarrow D$ }:

1. $A \rightarrow C$ adott.
2. $AB \rightarrow ABC$ (A2) alapján.
3. $B \rightarrow D$ adott.
4. $ABC \rightarrow ABCD$ (A2) alapján.
5. $AB \rightarrow ABCD$ (A3) alapján 2-ből és 4-ből.

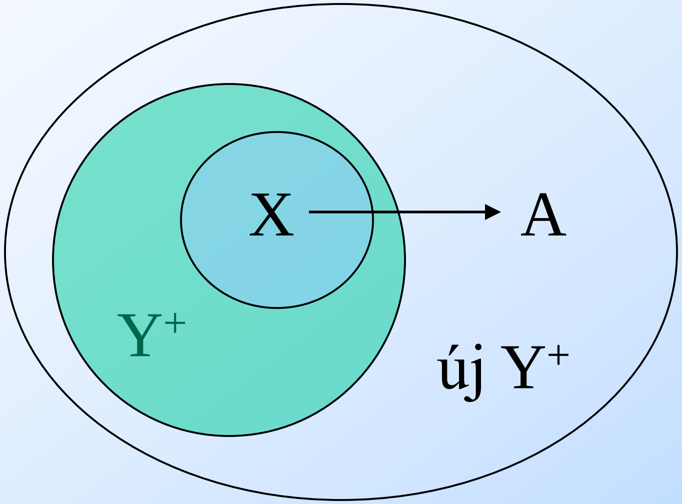
- Példa: bizonyítsuk be levezetéssel, hogy { $X \rightarrow Y$, $XY \rightarrow Z$ }-ből következik { $X \rightarrow Z$ }.

Újabb feladat

- Feladat: mutassuk meg, hogy az alábbiak nem érvényes szabályok funkcionális függőségekre:
  - ha $A \rightarrow B$, akkor $B \rightarrow A$,
  - ha $AB \rightarrow C$ és $A \rightarrow C$, akkor $B \rightarrow C$,
  - ha $AB \rightarrow C$, akkor $A \rightarrow C$ vagy $B \rightarrow C$.

Lezárás

- Egy egyszerűbb út, ha kiszámítjuk Y *lezártját*, jelölésben Y^+.
- Kiindulás: $Y^+ = Y$.
- Indukció: Olyan FF-ket keresünk, melyeknek a baloldala már benne van Y^+-ban. Ha $X \to A$ ilyen, A-t hozzáadjuk Y^+-hoz.


X
A
Y+
új Y+

Mi is az a lezárt?

- Adott *R* séma és *F* FF halmaza mellett, X^+ az összes olyan *A* attribútum halmaza, amire *X->A* következik *F*-ből.

A lezárást kiszámító algoritmus „helyes” I.

- Az algoritmus „tényleg” X^+-t számítja ki. Vagyis:
  1. Ha az A attribútum valamely *j*-re belekerül X^j-be, akkor A valóban eleme X^+-nak.
  2. Másfelől, ha A $\in X^+$, akkor létezik olyan *j*, amire A belekerül X^j-be.
- Megjegyzés: az első állítás könnyen bizonyítható indukcióval.

A lezárást kiszámító algoritmus „helyes” II.

- (2) Tegyük fel, hogy A $\notin X^+$, tehát nem létezik olyan *j*, amire A belekerül X^j-be.

	X^+ elemei	más attribútumok
t	111 … 111	000 … 000
s	111 … 111	111 … 111

- Ekkor az előbbi előfordulásra minden F-beli függőség teljesül. (Miért?)
- $X \rightarrow A$ viszont nem teljesül.

Feladatok

1. R = SEBADNM, F = { S $\rightarrow$ EBAD, D $\rightarrow$ NM }, mi az S és D attribútumok F-re vonatkozó lezártja?
2. R = HTUSDK, F = { H $\rightarrow$ T, U $\rightarrow$ HS, HD $\rightarrow$ KU }, adjuk meg az összes F-szerinti kulcsát R-nek és lássuk be, hogy csak a megadottak kulcsok.
3. Bizonyítsuk be, hogy $(X^+)^+ = X^+$.

Az összes következmény FF megtalálása

- Motiváció: „normalizálás”, melynek során egy reláció sémát több sémára bonthatunk szét.
- Példa: *ABCD* FF-k: *AB* -> *C*, *C* -> *D* és *D* -> *A*.
  - Bontsuk fel *ABC* és *AD*-re. Milyen FF-k teljesülnek *ABC* –n?
  - Nem csak *AB* -> *C*, de *C* -> *A* is!

Miért?

ABCD $a_1 b_1 c d_1$ $a_2 b_2 c d_2$

$d_1 = d_2$ mert
C -> D

ebből

$a_1 = a_2$ mert
D -> A

ABC $a_1 b_1 c$ $a_2 b_2 c$

Emiatt, ha két vetített sor C-n megegyezik, akkor A-n is, azaz: *C -> A*.

Alapötlet

1. Induljunk ki a megadott FF-ekből és keressük meg az összes *nem triviális* FF-t, ami a megadott FF-ekből következik.
   - Nem triviális = a jobboldalt nem tartalmazza a bal.
2. Csak azokkal az FF-kel foglalkozzunk, amelyekben a projektált séma attribútumai szerepelnek.

Exponenciális algoritmus

1. Minden X attribútumhalmazra számítsuk ki X^+-t.
2. Adjuk hozzá a függőségeinkhez $X \rightarrow A$-t minden A-ra $X^+ - X$-ből.
3. Dobjuk ki $XY \rightarrow A$-t, ha $X \rightarrow A$ is teljesül.
   - Mert $XY \rightarrow A$ $X \rightarrow A$-ból minden esetben következik.
4. Végül csak azokat az FF-ket használjuk, amelyekben csak a projektált attribútumok szerepelnek.

Néhány trükk

- Az üreshalmaznak és az összes attribútum halmazának nem kell kiszámolni a lezártját.
- Ha X^+ = az összes attribútum, akkor egyetlen X- t tartalmazó halmaznak sem kell kiszámítani a lezártját.

Példa: FF-k projekciója

- ABC, $A \text{->} B$ és $B \text{->} C$ FF-kel. Projektáljunk AC-re.
  - $A^+ = ABC$; ebből $A \text{->} B$, $A \text{->} C$.
    - Nem kell kiszámítani AB^+ és AC^+ lezárásokat.
  - $B^+ = BC$; ebből $B \text{->} C$.
  - $C^+ = C$; semmit nem ad.
  - $BC^+ = BC$; semmit nem ad.

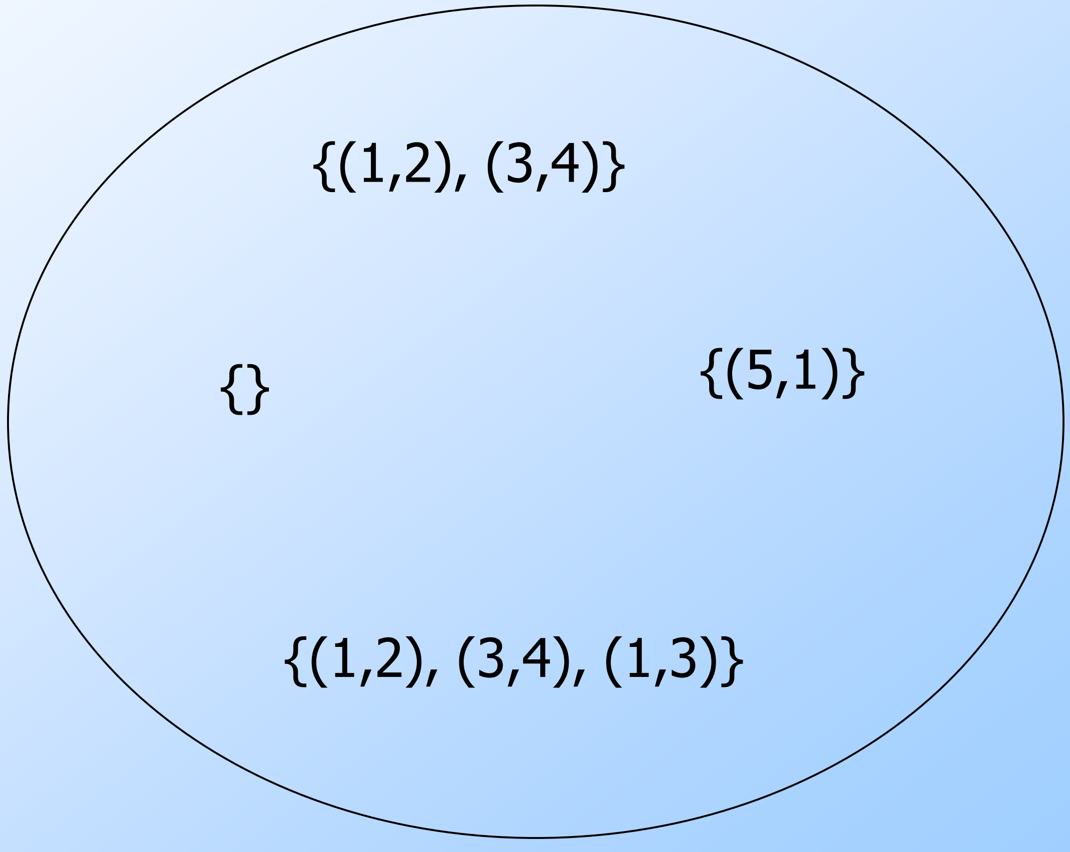
Példa -- folytatása

- A kapott FF-ek: $A \to B$, $A \to C$ és $B \to C$.
- AC-re projekció: $A \to C$.

Az FF-k geometriai reprezentációja

- Vegyük egy reláció összes lehetséges *előfordulásainak* halmazát.
- Azaz az összes olyan sorhalmazt, mely sorok komponensei a „megfelelőek”.
- Minden ilyen halmaz egy pont a térben.

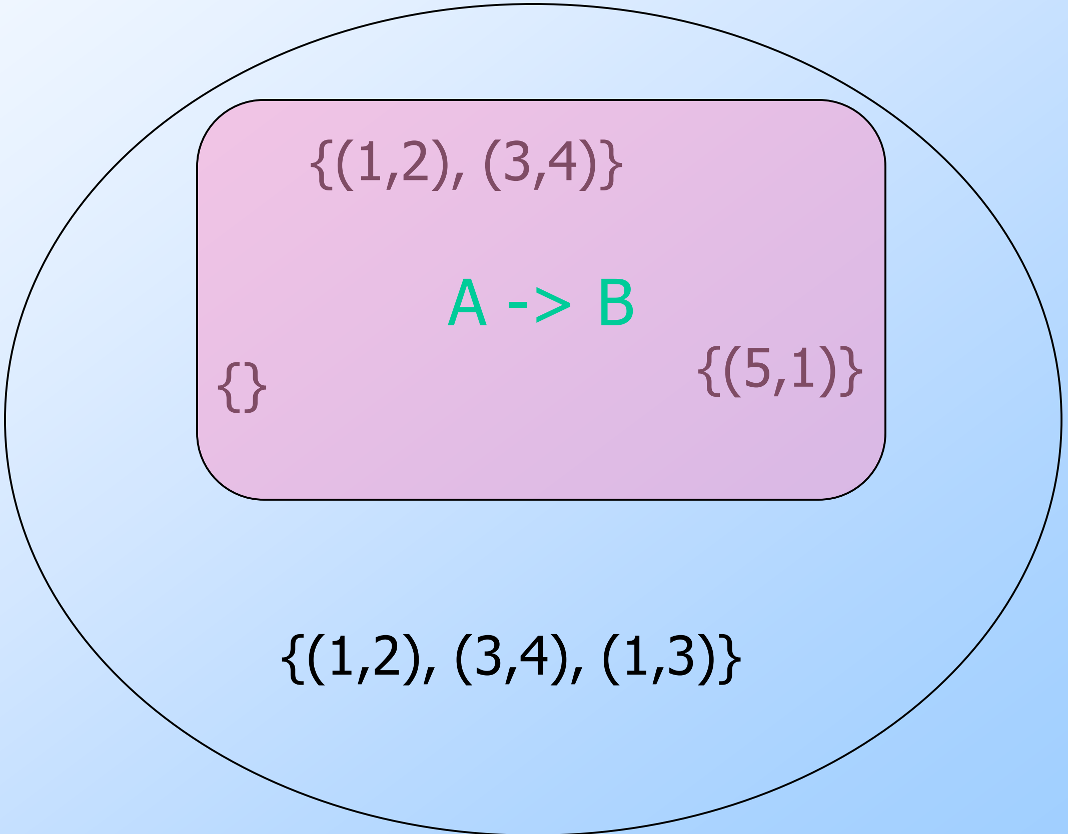
Példa: R(A,B)



Egy FF az előfordulásoknak egy részhalmaza

- Minden *X* -> *A* FF megadható azon előfordulások részhalmazaként, mely teljesíti FF-t.
- Így minden FF egy régióval jellemezhető a térben
- A triviális FF-k azok, melyeknél ez a régió a teljes tér.
  - Példa: *A* -> *A*.

Példa: A -> B R(A,B) fölött

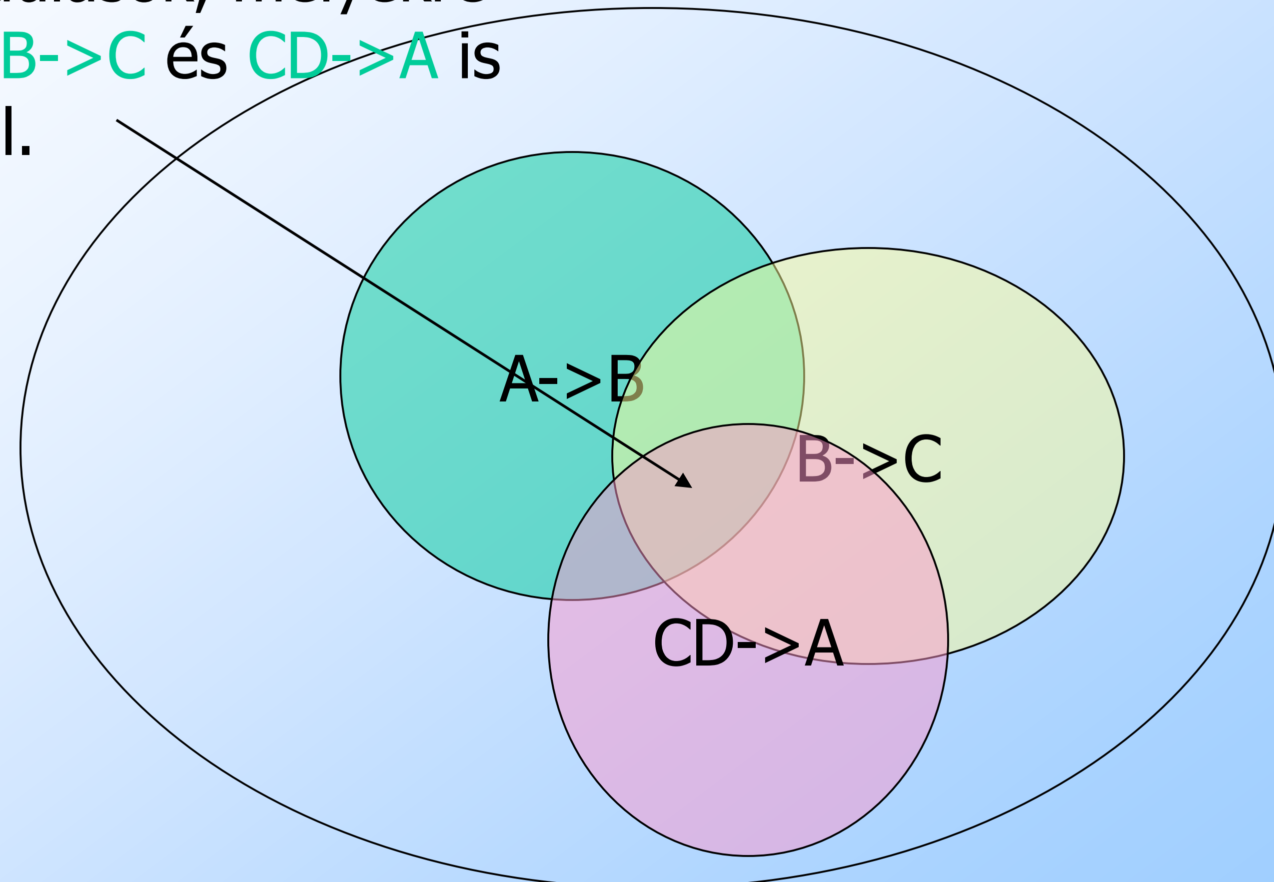


FF-k halmazának reprezentálása

- Ha egy-egy FF előfordulásoknak egy halmazával reprezentálható, akkor az FF-ek halmaza az előbbi halmazok metszetével lesz egyenlő.
  - Azaz a metszet = azon előfordulások, amelyekre mindegyik FF teljesül.

Példa

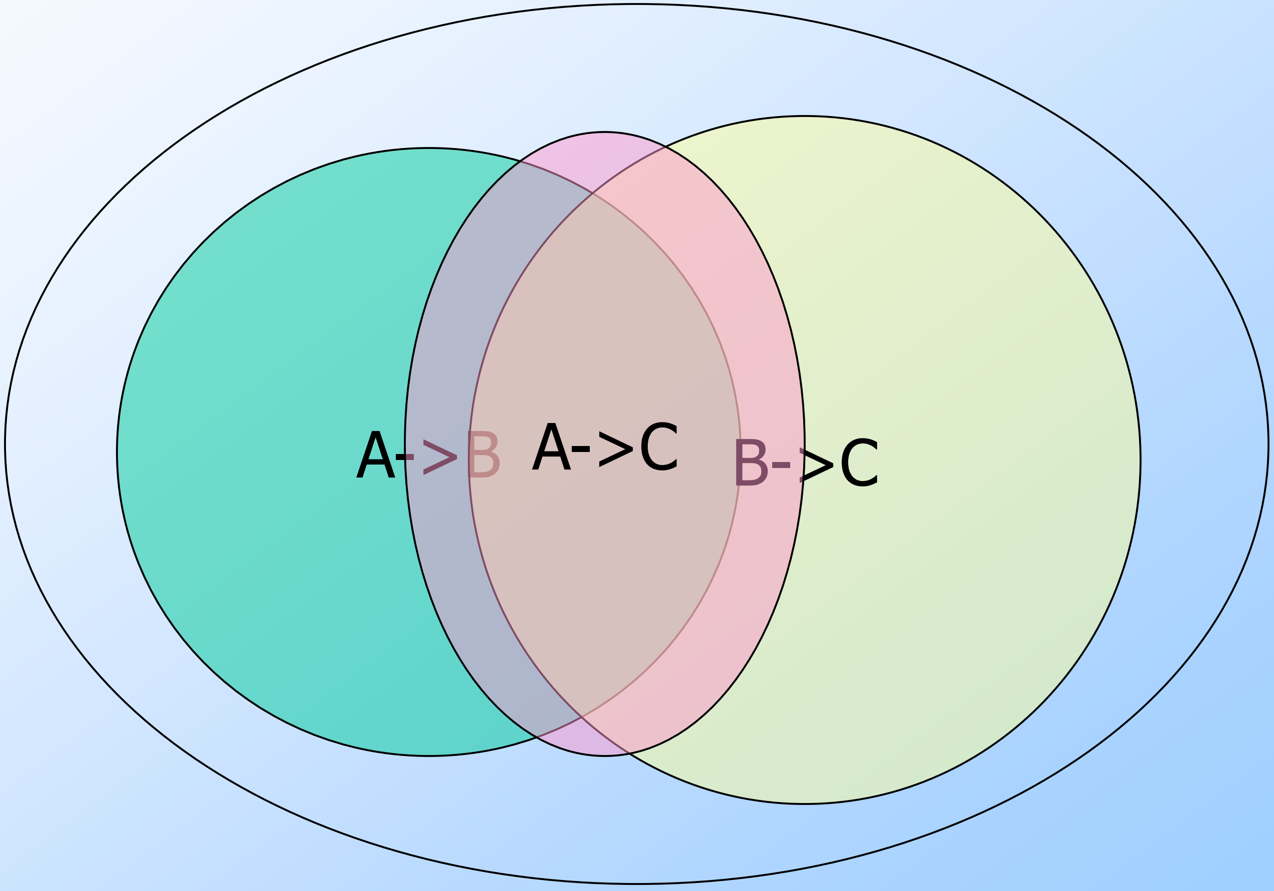
Előfordulások, melyekre A->B, B->C és CD->A is teljesül.



FF-k következtetése

- Ha FF Y -> B következik X_1 -> A_1,…, X_n -> A_n FF-ekből, akkor az Y -> B régiójának tartalmaznia kell az X_i -> A_i FF-ekhez tartozó régiók metszetét.
  - Azaz: minden előfordulás, ami teljesíti X_i -> A_i -t, Y -> B -t is teljesíti.
  - Ugyanakkor ha egy előfordulásra teljesül Y -> B, X_i -> A_i nem feltétlen teljesül.

Példa



Relációs sémák tervezése

- Cél: az anomáliák és a redundancia megszüntetése.
  - *Módosítási anomália* : egy adat egy előfordulását megváltoztatjuk, más előfordulásait azonban nem.
  - *Törlési anomália* : törléskor olyan adatot is elveszítünk, amit nem szeretnénk.

Példa: rosszul tervezett séma

Főnökök(<u>név</u>, cím, <u>kedveltSörök</u>, gyártó, kedvencSör)

név	cím	kedveltSörök	gyártó	kedvencSör
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	???	WickedAle	Pete's	???
Spock	Enterprise	Bud	???	Bud

Redundáns adat, a ??? helyén a név -> cím kedvencSör és kedveltSörök -> gyártó FF-ek felhasználásával tudjuk, mi szerepel.

A rosszul tervezettség anomáliákat is eredményez

név	cím	kedveltSörök	gyártó	kedvencSör
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

- Módosítási anomália: ha Janeway-t *Karcsira* módoítjuk, megtesszük-e ezt minden sornál?
- Törlési anomális: Ha senki sem szereti a Bud sört, azt sem tudjuk, ki gyártotta.

Boyce-Codd normálforma

- *R* reláció *BCNF* normálformában van, ha minden *X* -> *Y* nemtriviális FF-re *R*-ben *X* szuperkulcs.
  - *Nemtriviális:* *Y* nem része *X*-nek.
  - *Szuperkulcs:* tartalmaz kulcsot (ő maga is lehet kulcs).

Példa

Főnökök(<u>név</u>, cím, <u>kedveltSörök</u>, gyártó, kedvencSör)

FF-ek: név->cím kedvencSör, kedveltSörök->gyártó

- Itt egy kulcs van: {név, kedveltSörök}.
- A baloldalak egyik FF esetén sem szuperkulcsok.
- Emiatt az *Főnökök* 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

- Adott *R* reláció és *F* funkcionális függőségek.
- Van-e olyan $X \text{->} Y$ 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 szuperkulcs.

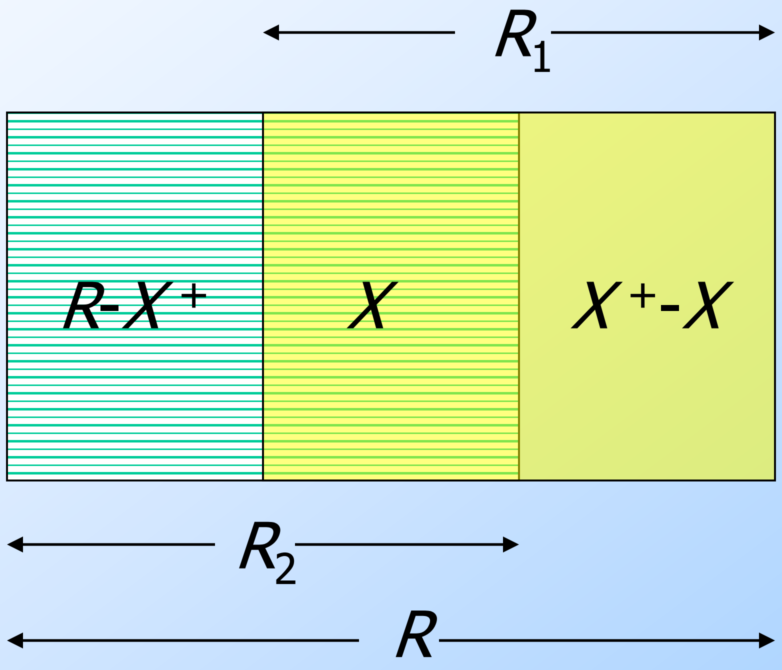
R dekomponálása X -> Y felhasználásával

- 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

Alkeszek(<u>név</u>, cím, <u>kedveltSörök</u>, gyártó, kedvencSör)

F = név->cím, név->kedvencSör, kedveltSörök->gyártó

- Vegyük név->cím FF-t:
- $\{$név$\}^{+}$ = {név, cím, kedvencSör}.
- A dekomponált relációsémák:

1. Alkeszek1(<u>név</u>, cím, kedvencSör)
2. Alkeszek2(<u>név</u>, <u>kedveltSörök</u>, gyártó)

Példa -- folytatás

- Meg kell néznünk, hogy az Alkeszek1 és Alkeszek2 táblák BCNF-ben vannak-e.
- Az FF-ek projektálása könnyű.
- A Alkeszek1(<u>név</u>, cím, kedvencSör), az FF-ek név->cím és név->kedvencSör.
  - Tehát az egyetlen kulcs: {név}, azaz az Alkeszek1 BCNF-ben van.

Példa -- folytatás

- Az Alkeszek2(<u>név</u>, <u>kedveltSörök</u>, gyártó) esetén az egyetlen FF

  kedveltSörök->gyártó, az egyetlen kulcs: {név, kedveltSörök}.
  - Sérül a BCNF.
- kedveltSörök^{+} = {kedveltSörök, gyártó}, az *Alkeszek2* felbontása:

1. Alkeszek3(<u>kedveltSörök</u>, gyártó)
2. Alkeszek4(<u>név</u>, <u>kedveltSörök</u>)

Példa -- befejezés

- Az *Alkeszek* dekompozíciója tehát:
  1. Alkeszek1(<u>név</u>, cím, kedvencSör)
  2. Alkeszek 3(<u>kedveltSörök</u>, gyártó)
  3. Alkeszek 4(<u>név</u>, <u>kedveltSörök</u>)
- Az *Alkeszek1* az alkeszekről, az *Alkeszek3* a sörökről, az *Alkeszek4* az alkeszek és kedvelt söreikről tartalmaz információt.

Miért működik a BCNF?

- (R, F) esetén ha $R_1, \ldots, R_k$ egy veszteségmentes felbontás, S_1, S_2 pedig R_1 veszteségmentes felbontása, akkor S_1, S_2, $R_2, \ldots, R_k$ is veszteségmentes felbontás.
- Könnyen ellenőrizhető, hogy a fenti R_1, R_2 veszteségmentes.

  Feladat: bizonyítsuk be, hogy ha az R(A, B, C) reláció esetén $B \rightarrow C$ teljesül, akkor az $R_1(A, B)$, $R_2(B, C)$ felbontás mindig veszteségmentes.
- Minden két attribútumú séma BCNF normálformában van.
- Az algoritmus tehát valóban veszteségmentes felbontást ad, ám sajnos exponenciális lépésszámú is lehet a függőségek vetítése miatt.

Feladat

1. Adott $R = ABCD$ reláció és $F = \{ AB \rightarrow C, A \rightarrow D, BD \rightarrow C\}$ függőségi halmaz mellett adjuk meg R egy BCNF dekompozícióját.
2. Legyen most $R = BOISQD$, $F = \{ S \rightarrow D, I \rightarrow B, IS \rightarrow Q, B \rightarrow O \}$. Ugyanez a feladat.

Veszteségmentes szétvágás I.

- Ha $r = \Pi_{R1}(r) \ |X| \ \ldots \ |X| \ \Pi_{Rk}(r)$ teljesül, akkor az előbbi összekapcsolásra azt mondjuk, hogy veszteségmentes. Itt r egy R sémájú relációt jelöl.
- $\Pi_{Ri}(r)$ jelentése: r sorai az Ri attribútumaira projektálva.
- Megjegyzés: könnyen látható, hogy $r \subseteq \Pi_{R1}(r) \ |X| \ \ldots \ |X| \ \Pi_{Rk}(r)$ mindig teljesül. (Miért?)

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

Példa

- A szétvágás után keletkező relációk összekapcsolása nem veszteségmentes:

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

Chase-teszt veszteségmentességhez I.

- 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 |X| R_2 |X| R_3 egy $t = (a, b, c, d)$ sorát. Bizonyítani kell, hogy t R egy sora. A következő tablót készítjük el:

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 veszteségmentességhez II.

- 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 veszteségmentességhez III.

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 veszteségmentességhez IV.

- 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 → 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 → 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.

Chase-teszt veszteségmentességhez IV.

- $\{A, B\}$, $\{(a, b)\}$, $(a_3, b_3)\}$.
- $\{B, C\}$, $\{(b, c_1), (b, c), (b_3, c)\}$,
- $\{C, D\}$, $(c_1, d_1), (c, d_1), (c, d)\}$
- $\bowtie$, $\{(a, b, c_1), (a, b, c), (a_3, b_3, c)\}$.
- $\bowtie$, , $\{(a, b, c_1, d_1), (a, b, c, d_1), (a, b, c, d), (a_3, b_3, c, d_1), (a_3, b_3, c, d)\}.\}$
- 2 extra sor (rekord), és (a, b, c, d), amit kell is tartalmaznia.

A harmadik normálforma -- motiváció

- Bizonyos FF halmazok esetén a felbontáskor elveszíthetünk függőségeket.
- *AB ->C* és *C ->B*.
  - Példa: *A* = f_cím, *B* = város, *C* = mozi.
- Két kulcs van: {*A,B* } és {*A,C* }.
- {f_cím, város}, {f_cím, mozi}
- *C ->B* megsérti a BCNF-t, tehát *AC*, *BC*-re dekomponálunk. [*mozi->város,* nem szuperkulcs *C*]

FF-ek kikényszerítése

- A probléma az, hogy *AC* és *BC* sémákkal nem tudjuk kikényszeríteni

  AB -> *C* függőséget.
- Példa *A* = f_cím, *B* = város, *C* = mozi, a következő dián.

Egy kikényszeríthetetlen FF

F_cím	mozi
Antz	Guild
Antz	Park

város	mozi
Cambridge	Guild
Cambridge	Park

Kapcsoljuk össze a sorokat (mozi).

F_cím	város	mozi
Antz	Cambridge	Guild
Antz	Cambridge	Park

A szétbontott relációkban egyik FF sem sérül, az eredményben az F_cím város -> mozi nem teljesül.

A probléma megoldása: 3NF

- 3. normálformában (3NF) úgy módosul a BCNF feltétel, hogy az előbbi esetben nem kell dekomponálnunk.
- Egy attribútum *prím*, ha legalább egy kulcsnak eleme.
- *X* ->*A* megsérti 3NF-t akkor és csak akkor, ha *X* nem szuperkulcs és *A* nem prím (elsődleges attribútum).
- minden nem triviális függőségre igaz, hogy bal oldala szuperkulcs, vagy jobb oldala csak elsődleges attribútumokat tartalmaz
- 3NF feltétel és a BCNF feltétel közötti különbség a „vagy jobb oldala csak elsődleges attribútumokat tartalmaz”

Példa: 3NF

- A problematikus esetben az *AB->C* és *C->B* FF-ek esetén a kulcsok *AB* és *AC*.
- Ezért *A*, *B* és *C* mindegyike prím.
- Habár *C->B* megsérti a BCNF feltételét, 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 kikényszeríthetőek az előre megadott függőségek.

3NF és BCNF -- folytatás

- Az (1) tulajdonság teljesül a BCNF esetében.
- A 3NF (1) és (2)-t is teljesíti.
- A BCNF esetén (2) sérülhet.
  - Az F_cím - város - mozi erre volt egy példa.

Minimális bázis létrehozása

1. Jobboldalak szétvágása.
2. Próbáljuk törölni az FF-eket egymás után. Ha a megmaradó FF-halmaz nem ekvivalens az eredetivel, akkor nem törölhető az épp aktuális FF.
3. Egymás után próbáljuk csökkenteni a baloldalakat, és megnézzük, hogy az eredetivel ekvivalens FF-halmazt kapunk-e.

3NF-re bontás – (2)

- A minimális bázis minden FF-re megad egy sémát a felbontásban.
  - A séma a jobb- és baloldalak uniója lesz (*X*-> *A* FF, *XA* séma) .
- Ha a minimális bázis FF-jei által meghatározott sémák (*X*-> *A* FF, *XA* séma) között nincs *szuperkulcs*, akkor hozzáadunk a felbontáshoz egy olyan *sémát*, amely maga egy *kulcs* az R relációra.

Példa: 3NF felbontás

- A reláció: R = ABCD.
- FF-ek: *A->B* és *A->C*.
- Felbontás: AB és AC az FF-ekből és AD-t is hozzá kell venni, mert AB, AC egyike sem kulcs.

Miért működik?

- 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.

Minimális bázist kiszámító algoritmus

Jelölje F^+ az F függőségi halmazból következő függőségek halmazát.

1. Kezdetben G az üreshalmaz.
2. Minden $X \rightarrow Y \in F$ helyett vegyük az $X \rightarrow A$ függőségeket, ahol $A \in Y - X$).

   Megjegyzés: Ekkor minden G-beli függőség $X \rightarrow A$ alakú, ahol A attribútum.
3. 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.
4. Minden $X \rightarrow A \in G$-re, amíg van olyan $B \in X$-re, hogy $A \in (X - B)^+$ a G-szerint, vagyis $(X - B) \rightarrow A$ teljesül, akkor $X := X - B$.

   Megjegyzés: E lépés után minden baloldal minimális lesz.

Feladat

- Példa: $F = \{ H \to T, U \to HS, HD \to KU \}$. Mi a minimális bázis?
- Feladat: $R = BOISQD$,

  $F = \{ S \to D, I \to BS, IS \to Q, B \to OQ, SD \to O\}$. Adjunk meg egy függőségőrző, veszteségmentes 3NF dekompozíciót.