# Adatbázisok vizsgabeugró mintakérdések

A vizsgabeugrón hasonló nehézségű feladatokra lehet számítani. A kidolgozására 10-15 perc áll rendelkezésre. Súlyos hiba esetén a vizsga elégtelen, kis hiba esetén jegylevonás a vizsgajegyből, esetleg pótbeugró írása. A gyűjtemény végén kb. a felének a megoldásását is megtalálod.

[Adatbázisok vizsgabeugró mintakérdések 1](#_Toc300262721)

[1. Szigorú 2PL 2](#_Toc300262722)

[2. REDO protokoll 2](#_Toc300262723)

[3. Sorosítási és várakozási gráf 2](#_Toc300262724)

[4. Fa protokoll 2](#_Toc300262725)

[5. Figyelmeztető protokoll 2](#_Toc300262726)

[6. Entitás relációs diagramból hálós séma 2](#_Toc300262727)

[7. Relációs sémák felbontása 2](#_Toc300262728)

[8. Példák különböző normálformákra 2](#_Toc300262729)

[9. Hálós adatmodell set típusa 2](#_Toc300262730)

[10. Kulcskeresés, normálformák 2](#_Toc300262731)

[11. Reláció algebra 2](#_Toc300262732)

[12. Sorkalkulus, oszlopkalkulus és relációalgebra 2](#_Toc300262733)

[13. Dekompozíció BCNF sémákba 2](#_Toc300262734)

[14. Objektum-orientált adatbáziskezelők 2](#_Toc300262735)

[15. Armstrong axiómák 2](#_Toc300262736)

[16. Fizika társzervezés vödrös hashing segítségével 2](#_Toc300262737)

[17. Biztonságos sorkalkulus 2](#_Toc300262738)

[18. Időbélyeges tranzakciókezelés 2](#_Toc300262739)

[19. Verziókezelés időbélyegek mellett 2](#_Toc300262740)

[20. Veszteségmentes és függőségőrző felbontás 2](#_Toc300262741)

[21. Vetített relációban levő sorok száma 2](#_Toc300262742)

[22. Függőséghalmaz elemszáma 2](#_Toc300262743)

[23. Függéshalmaz minimális fedése 2](#_Toc300262744)

[Megoldások 2](#_Toc300262745)

[1. Szigorú 2PL 2](#_Toc300262746)

[3. Sorosítási és várakozási gráf 2](#_Toc300262747)

[4. Fa protokoll 2](#_Toc300262748)

[5. Figyelmeztető protokoll 2](#_Toc300262749)

[6. Entitás relációs diagramból hálós séma 2](#_Toc300262750)

[7. Relációs sémák felbontása 2](#_Toc300262751)

[13. Dekompozíció BCNF sémákba 2](#_Toc300262752)

[15. Armstrong axiómák 2](#_Toc300262753)

[16. Fizika társzervezés vödrös hashing segítségével 2](#_Toc300262754)

[17. Biztonságos sorkalkulus 2](#_Toc300262755)

[18. Időbélyeges tranzakciókezelés 2](#_Toc300262756)

[19. Verziókezelés időbélyegek mellett 2](#_Toc300262757)

[20. Veszteségmentes és függőségőrző felbontás 2](#_Toc300262758)

[21. Vetített relációban levő sorok száma 2](#_Toc300262759)

[22. Függőséghalmaz elemszáma 2](#_Toc300262760)

[23. Függéshalmaz minimális fedése 2](#_Toc300262761)

## 1. Szigorú 2PL

Adott egy tranzakció. Szigorú 2PL-e? Ha nem, módosítsa, hogy az legyen! Mit garantál a szigorú 2PL? Legyen a tranzakció szekvenciája a következő:

LOCK A

LOCK B

READ A

UNLOCK A

LOCK C

C=A\*2

WRITE C

COMMIT

UNLOCK B

UNLOCK C

## 2. REDO protokoll

Mi történik az „újra” protokoll szerint, ha a tranzakció a bejelölt (1-6) pontokban abortál? Hogyan működik az „újra” protokoll?

|  |  |
| --- | --- |
|  | naplóba(T, BEGIN) |
| 1 |  |
|  | LOCK(A); |
|  | LOCK(B); |
| 2 |  |
|  | naplóba(T, <A régi értéke>, <A új értéke>); |
|  | naplóba(T, <B régi értéke>, <B új értéke>); |
| 3 |  |
|  | naplóba(T, COMMIT); |
| 4 |  |
|  | write(A); |
|  | write(B); |
| 5 |  |
|  | UNLOCK(A); |
|  | UNLOCK(B); |
| 6 |  |

## 3. Sorosítási és várakozási gráf

1) Rajzold fel a sorosítási és várakozási gráfot nem kétfázisú és kétfázisú esetben! (Az ütemezés legyen legális).

 T1 T2

 READ A

 READ B

 READ B

 READ A

2) Rajzold fel a várakozási és a precedencia (sorosítási) gráfot. Használj zárakat. Hogyan változnak a gráfok, ha kétfázisú protokollt használunk?

 T1 T2

 WRITE A

 WRITE B

 WRITE B

 WRITE A

## 4. Fa protokoll

Fa protokoll felismerése. Adott egy tranzakció, és ebből kell felrajzolni a fát.

LOCK A

LOCK B

UNLOCK A

LOCK C

LOCK D

UNLOCK D

LOCK E

LOCK F

UNLOCK C

(zárak feloldása innentől számunkra érdektelen)

## 5. Figyelmeztető protokoll

A következő, figyelmeztető protokollt használó ütemezéshez tartozó lehetséges hierarchiákat keressük.

 T1 T2 T3

 WARN A

 WARN A

 WARN A

 WARN B

 LOCK C

 LOCK D

 UNLOCK C

 LOCK E

 UNLOCK D

 UNLOCK A

 UNLOCK E

 UNLOCK B

 LOCK B

 WARN C

 LOCK F

 UNLOCK A

 LOCK G

 UNLOCK B

 UNLOCK F

 UNLOCK C

 UNLOCK A

## 6. Entitás relációs diagramból hálós séma

ER-t kell rajzolni szöveges leírásból, majd abból hálós sémát készíteni.

## 7. Relációs sémák felbontása

Bizonyítsd be, hogy egy relációs séma egy tetszőleges felbontásánál a rész-sémákra vetített relációk természetes illesztésével sorok nem tűnhetnek el, csak újak jelenhetnek meg.

## 8. Példák különböző normálformákra

Adj olyan konkrét példákat a különböző normálformákat (1NF, 2NF, 3NF, BCNF) kielégítő relációs sémákra, ami adott normálformában benne van, de magasabb szintűben nem.

A válasz egy relációs séma, a rajta értelmezett függőségek és annak bizonyítása, hogy a séma az adott normálformában benne van, de az eggyel magasabbiban nem.

## 9. Hálós adatmodell set típusa

Bizonyítsd be, hogy a hálós modell esetén az egy set-típuson belül az owner rekordokhoz kapcsolódó memberek halmazai diszjunktak.

## 10. Kulcskeresés, normálformák

Adott relációs sémában kulcsok keresése, ill. annak bizonyítása, hogy csak azok a kulcsok vannak. Milyen normál formájú a séma?

## 11. Reláció algebra

Van egy színházas reláció sok attribútummal, valamint ezek függőségei.

1. Keressünk kulcsot! Milyen normálformában van a séma?

2. Adott egy másik reláció, és ebből a kettőből kellett a szokásos reláció algebrai lekérdezést csinálni adott című és szerzőjű színdarabra. Utána ugyanezt meg kellett fogalmazni sorkalkulussal is.

## 12. Sorkalkulus, oszlopkalkulus és relációalgebra

Sorkalkulus és relációalgebra felírása egy szövegesen megadott lekérdezésre.

## 13. Dekompozíció BCNF sémákba

Egy adott relációs sémát kell megadott függéshalmaz mellett veszteségmentes BCNF sémákra felbontani.

## 14. Objektum-orientált adatbáziskezelők

Az objektum-orientált adatbáziskezelésnél van egy fás-madaras-fészkes példa a könyvben. Azt kell lerajzolni és megmagyarázni.

## 15. Armstrong axiómák

Mutasd meg, hogy igaz a tranzitivitási axióma! X→Y és Y→Z esetén X→Z

Mutasd meg, hogy igaz a bővítési axióma! X→Y esetén XZ→YZ

**Mutasd meg, hogy igaz** a triviális függőségi axióma! Y⊆X esetén X→Y

## 16. Fizika társzervezés vödrös hashing segítségével

Vödrös hash szervezést használunk. A vödörkatalógusról tudjuk, hogy befér a memóriába. Cél: a rekordok elérése legfeljebb 3 blokkelérést jelentsen. A rekordok eloszlása egyenletes, a hash függvényé is. A paraméterek:

– 1 millió rekord

– 350 byte egy rekord

– 9 byte egy mutató

– 30 byte egy kulcs

– 2000 byte egy blokk header nélkül

Hány byte a vödörkatalógus mérete?

## 17. Biztonságos sorkalkulus

DOM(Ψ) definíció, alkalmazása, példák biztonságos és nem biztonságos kifejezésekre.

## 18. Időbélyeges tranzakciókezelés

Elemezze lépésről lépésre a következő időbélyeges tranzakciókat (hol abortál, mikor mi lesz a bélyegek értéke). Miben módosul az egész, ha egyszer írható végtelen tárolóegységet használunk?

## 19. Verziókezelés időbélyegek mellett

Verziókezeléses rendszerben futó tranzakciók futását kell elemezni.

## 20. Veszteségmentes és függőségőrző felbontás

Van egy relációnk, és annak egy nem függőségőrző felbontása. Ha a felbontásában az egyik relációhoz hozzáadunk egy új elemet, melyen nem érvényesül(nek) az „elveszett” függőség(ek), akkor a relációba egy helytelen elem kerülhet. Ezután ha vesszük a felbontásában szereplő relációk természetes illesztését, akkor az eredeti relációnál bővebb relációt kapunk, tehát a nem függőségőrző felbontás nem veszteségmentes.

Hol a hiba a gondolatmenetben?

## 21. Vetített relációban levő sorok száma

Legyen r egy R sémára illeszkedő r reláció, X pedig R attribútumainak egy részhalmaza.

Bizonyítsd be, hogy ha ∀ r-re ΠX(r) és r sorainak száma megegyezik, akkor bármely Y⊆R-re fennáll az X→Y funkcionális függés!

## 22. Függőséghalmaz elemszáma

Adott egy R(A,B,C) reláció és a rajta értelmezett függőségek F halmaza. R(A, B, C), F = {AB→C, C→A}. Hány nemtriviális függőség van a függőséghalmaz lezártjában?

## 23. Függéshalmaz minimális fedése

Adott egy relációs séma: R(A, B, C, D, E) és a rajta értelmezett F függőségek halmaza: F = {AB→CE, AD→C, CD→B, DE→BC, C→A, A→C}. Határozzuk meg ennek egy minimális fedését.

# Megoldások

## 1. Szigorú 2PL

A szigorú 2PL definícióját ez a tranzakció két dolog miatt sérti.

* A zár feloldása után van még újabb zárkérés (kétfázisú tulajdonság megsértése).
* A COMMIT, adatbázisba írás, zárak elengedése sorrendet sérti az, hogy előbb írjuk C-t, mielőtt commitálunk. Természetesen a tranzakció a kétfázisú tulajdonság sértése miatt a zárak elengedése sorrendet is sérti egyúttal, hiszen pl. UNLOCK A a COMMIT, adatbázisba írás szekvencia előtt van.

Készítsünk tehát a fenti tranzakció hatásaival ekvivalens, de szigorú 2PL protokoll szerinti tranzakció szekvenciát.

LOCK A

READ A

LOCK C

C=A\*2

COMMIT

WRITE C

UNLOCK A

UNLOCK C

Jól látszik, hogy a zárak elkérése, műveletek elvégzése, COMMIT, adatbázisba írás, zárak elengedése pontosan ebben a sorrendben történt, tehát valóban a szigorú kétfázisú protokoll szerint működik a tranzakció. Az pedig, hogy a fenti tranzakció hatásával ekvivalens ezen tranzakció hatása, következik abból, hogy csak a LOCK és UNLOCK műveletek rendjén változtattunk, mégpedig úgy, hogy az ütemezés legális.

## 3. Sorosítási és várakozási gráf

a) Egy lehetséges nem kétfázisú protokollt követő ütemezés:

 T1 T2

 LOCK A

 READ A

UNLOCK A

 LOCK B

 READ B

 UNLOCK B

 LOCK A

 LOCK B

 READ B

 READ A

UNLOCK A

UNLOCK B

Az ütemezés legális, T1 zárolhatja A-t kétszer, mert nem 2PL esetén semmi sem tiltja, hogy egy adategységet egy tranzakción belül többször is zároljunk.

A várakozási gráf: T1 és T2 a gráf két pontja, de közöttük nincs él, hiszen se T1 nem váratja T2-t, sem fordítva (a várakozási gráf tetszőleges időben felrajzolvanem tartalmaz élet).

A sorosítási gráf: T2→T1, hiszen miután T2 feloldja B-n a zárat, T1 zárolja B-t.

b) Egy lehetséges kétfázisú protokollt követő ütemezés:

 T1 T2

 LOCK A

 READ A

 LOCK B

 READ B

 UNLOCK B

 LOCK B

 READ B

 READ A

UNLOCK A

UNLOCK B

A várakozási gráf: T1 és T2 a gráf két pontja, de közöttük nincs él, hiszen se T1 nem váratja T2-t, sem fordítva (a várakozási gráf tetszőleges időben felrajzolvanem tartalmaz élet).

A sorosítási gráf: T2→T1, hiszen miután T2 feloldja B-n a zárat, T1 zárolja B-t.

2) Rajzold fel a várakozási és a precedencia (sorosítási) gráfot. Használj zárakat. Hogyan változnak a gráfok, ha kétfázisú protokollt használunk?

 T1 T2

 WRITE A

 WRITE B

 WRITE B

 WRITE A

## 4. Fa protokoll

LOCK A után LOCK B miatt B nyilván A gyereke. UNLOCK A miatt a továbbiak csak B gyerekei lehetnek, különben sérülne a fa protokoll. LOCK C miatt C a B adatelem gyereke lesz. Idáig egyértelmű a fa, viszont LOCK D-nél nem tudjuk eldönteni, hogy D a C alá kerüljön vagy B alá, és így C testvére legyen. Tehát minden olyan esetben, amikor több adategység is zárolva van egyidőben, a következő zár elhelyezésekor bárhogy dönthetünk, melyik alá rajzoljuk be az adategységet a fában. Emiatt nem lesz egyértelmű a megoldás.

## 5. Figyelmeztető protokoll

Az adategységek hierarchiáját leíró fára egy lehetséges megoldást a könyvben lehet találni, a 9.5.2. Figyelmeztető protokoll c. fejezetében „Az ütemezéshez tartozó hierarchia” címmel.

A fa protokollhoz hasonlóan itt sem egyértelmű a megoldás, hiszen ha egy tranzakcióban egyidejűleg több adategységen is van WARN, akkor a következő LOCK X-nél az X adategység akármelyik WARN-nal megjelölt adategység gyereke lehet a fában.

Amennyiben ugyanazt az adategységet több tranzakció is zárolja (természetesen nem egyidőben), és valamelyik zároláskor csak egy érvényes WARN létezik, akkor az alapján egyértelmű melyik adategység gyereke lesz X. Érdemes tehát nem egyértelmű esetekben megnézni a többi tranzakció szekvenciáit is, hátha az alapján tudunk dönteni. Ha nem, akkor tetszőlegesen felvehetjük az adategységet a WARN-nal rendelkező csomópontok valamelyike alá, s így nyilván többféle fa is elképzelhető megoldásként.

Esetünkben pl. a D adategység kerülhet közvetlen A alá, vagy B alá is, hiszen D-t csak T1-ben zároljuk, és WARN A ill. WARN B is érvényben van ekkor.

## 6. Entitás relációs diagramból hálós séma

A szöveges leírásból fel kell rajzolni az ER diagramot. ER-ből hálós modellbe úgy konvertálunk, hogy az entitásoknak rekordokat feleltetünk meg, a bináris több-egy kapcsolatoknak pedig seteket.

## 7. Relációs sémák felbontása

A könyv 8.2.4. Veszteségmentes felbontás c. fejezetének következő tétele: „Adott egy R séma és egy ρ(R1, R2, …, Rn) felbontása. Minden r(R) relációra r⊆mρ(r).”

## 13. Dekompozíció BCNF sémákba

1NF sémák dekompozíciója BCNF sémákra tétel a könyvben (8.2.6 Sémadekompozíció adott normálformába c. fejezet), ennek a konstruktív bizonyítása módszert ad a BCNF sémákra bontásra. Van ott egy jó példa is egy NYELV relációval és a rajta értelmezett függőségekkel.

## 15. Armstrong axiómák

Mutasd meg, hogy igaz a tranzitivitási axióma! X→Y és Y→Z esetén X→Z

X→Y miatt a reláció minden t, t’∈r(R) sorára t[X] = t’[X] esetén t[Y] = t’[Y] is fennáll.

Y→Z miatt a reláció minden t, t’∈r(R) sorára t[Y] = t’[Y] esetén t[Z] = t’[Z] is fennáll.

Tehát ha a baloldal fennáll, akkor a reláció minden t, t’∈r(R) sorára t[X] = t’[X] esetén t[Y] = t’[Y] is fennáll, sőt ekkor t[Z] = t’[Z] is fennáll, így tehát X→Z is igaz lesz.

Mutasd meg, hogy igaz a bővítési axióma! X→Y esetén XZ→YZ

Lásd a tankönyvben.

**Mutasd meg, hogy igaz** a triviális függőségi axióma! Y⊆X esetén X→Y

Indirekt tfh. Y⊆X, de X→Y nem áll fent. A funkcionális függőség definíciója alapján ez azt jelenti, hogy ∃t, t’∈r(R) sor a relációban, amelyre t[X] = t’[X] esetén t[Y] ≠ t’[Y]. Mivel a feltétel szerint Y⊆X, így Y az X attribútumok egy részhalmaza, ahol ha X attribútumokban egyeznek a sorok, meg kell egyezniük az Y attribútumokban is, ám ez ellent mond az indirekt feltevésnek. Tehát az eredeti feltételezés igaz volt, azaz Y⊆X esetén X→Y teljesül.

## 16. Fizika társzervezés vödrös hashing segítségével

Amire nagyon figyeljetek, hogyha a vödörkatalógusban a kulcsnak is foglaltok helyet, akkor egyből mehettek haza!

3 blokk elérés maximum ⇒ egy vödör 3 blokkot tartalmaz.

1 blokkba alsó egészrész[2000 / 350] = 5 rekord fér el ⇒ a vödör 3 blokkjába így 15 rekord

1 millió rekord ⇒ 66.667 vödör ⇒ 66.667 × 9 byte = 600.003 byte a vödörkatalógus mérete.

## 17. Biztonságos sorkalkulus

**Biztonságos kifejezések**

Legyen R egy egyelemű reláció, értéke pl. 4. A {t | t = 2 /\ t > R} kifejezés esetén DOM(Ψ) = {2, 4}. Az igazságérték mindig hamis lesz, mivel 2 > 4 sosem teljesül, tehát a kifejezés biztonságos.

Minden hamis értékű kifejezés mindig biztonságos, mivel nem ad semmi eredményt, így nyilván nem ad DOM(Ψ)-n kívülit se.

**Nem biztonságos kifejezések**

Legyen az alaphalmaz A = {1, 2, 3, 4, 5, 6, 7, 8}. {t | t > 5} nem biztonságos, mert a Ψ-hez tartozó konstans 5, alapreláció nincs a formulában, tehát DOM(Ψ) = {5}. Viszont a kifejezés eredményként visszadja a 6, 7, 8 elemeket is, ami nincs benne DOM(Ψ)-ben.

Legyen az alaphalmaz A = {1, 2, 3, 4, 5, 6, 7, 8}, R pedig egy egyelemű reláció, értéke legyen pl. 4. {t | t = 2 \/ t > R} nem biztonságos. DOM(Ψ) = {2, 4}. A formula első fele, t = 2 nyilván akkor teljesül, ha t értéke 2. A formula másik fele pedig akkor, ha t nagyobb, mint 4, azaz t = 5, 6, 7, 8 esetén. A megoldáshalmaz így t = {2, 5, 6, 7, 8} lesz, ami nem eleme a DOM(Ψ) = {2, 4} halmaznak.

**Összetettebb kifejezés biztonságosságának vizsgálata**

Előfordulhat, hogy egy olyan kifejezésről kell megállapítani, hogy biztonságos-e, mely nem csak atomokból és formulákból, hanem részformulákból is áll (vagyis valamely formulába további formulák vannak ágyazva). Erre példa lehet a következő:

{t | R1(t) \/ R2(t) /\ Ψ(t), ahol Ψ(t) legyen R3(t) /\ (∃u)R4(u)}

Itt az a helyzet áll fenn, hogy az x \/ y /\ z alakú formulában z egy formula, ezért ennek a kifejezésnek a biztonságához teljesülnie kell a formulába ágyazott részformula biztonságosságának is.

A Ψ(t) formula biztonságos, ha az őt kielégítő t-k értékei DOM(R3(t))-beliek, valamint Ψ-nek a (∃u)R4(u) alakú részformulájára teljesül, hogy ha u kielégíti R4(u)-t, akkor az n elemű u sorváltozó minden komponense, vagyis u[1], u[2], ..., u[n]∈DOM(R4(u)). Emellett még természetesen annak is teljesülni kell, hogy t∈DOM(R1(t) \/ R2(t)). Összességében tehát akkor lesz biztonságos a kifejezés, ha teljesül minden a kifejezést kielégítő t-re, hogy t∈DOM(R1(t) \/ R2(t) /\ Ψ(t)), továbbá a részformulát kielégítő n elemű u sorváltozó minden komponense, vagyis u[1], u[2], ..., u[n]∈DOM(R4(u)).

Tetszőleges mélységig lehet ilyen részformulákat beágyazni, így először mindig a legmélyebbi formulára kell belátni, hogy az általa visszadott értékek biztonságosak-e, mert ha nem, akkor az egész kifejezés értéke nem lesz az, és már felesleges is tovább dolgozni.

## 18. Időbélyeges tranzakciókezelés

Egyszer írható tárolóegység esetén verziókezelésest fogunk használni, hiszen a régi értékeket csak olvasni tudjuk, felülírni nem.

## 19. Verziókezelés időbélyegek mellett

A példa a könyvben a 9.8.4 Verziókezelés időbélyegek mellett c. fejezetében található.

## 20. Veszteségmentes és függőségőrző felbontás

A nem függőségőrzés nem vonja maga után a nem veszteségmentességet. Ha egy új elemet hozzáadunk az egyik felbontásban szereplő relációhoz, akkor az mindenképpen megjelenik valamilyen formában az illesztésükben is, tehát a veszteségmentességet csak akkor vizsgálhatjuk, ha nem adunk hozzá új elemet a relációkhoz.

## 21. Vetített relációban levő sorok száma

A funkcionális függés definíciója szerint X→Y akkor áll fenn, ha minden t, t’∈r(R)-re t[X] = t’[X] esetén t[Y] = t’[Y] is fennáll. Amikor az X részhalmazra vetítünk, akkor minden olyan sor kiesik, amely X-ben megegyezik. Ha nem esett ki egyetlen sor sem, akkor az azt jelenti, hogy X értéke minden sorban különböző. Ebből következik, hogy X-től bármely Y⊆R-re fennáll az X→Y funkcionális függés, hiszen nem létezik t, t’∈r(R), amelyben t[X] = t’[X], így az implikáció szabálya szerint az X-től való funkcionális függőség ezután fennáll bármely attribútumhalmazra.

## 22. Függőséghalmaz elemszáma

Felhasználjuk, hogy egy n elemű halmaz részhalmazainak száma 2n.

Meghatározzuk az attribútumhalmazok lezártjait: AB+(F) = ABC, C+(F) = AC.

* AB→ABC részhalmazai: 23 = 8 db (AB→ABC, AB, AC, BC, A, B, C, 0), ám ez tartalmaz triviális függőségeket is, így azokat kivéve (23 – 4) = 4 db (AB→ABC, AC, BC, C).
* C→AC részhalmazai: 22 = 4 db (C→AC, A, C, 0), ám ez tartalmaz triviális függőségeket is, így azokat kivéve (22 – 2) = 2 db (C→AC, A).

A nemtriviális függőségek száma 12.

## 23. Függéshalmaz minimális fedése

A tanult algoritmus alapján egy lehetséges megoldás:

1. F' = {AB→C, AB→E, AD→C, CD→B, DE→B, DE→C, C→A, A→C}.
2. AB→C, AD→C, DE→B elhagyásával F'' = {AB→E, CD→B, DE→C, C→A, A→C}.
3. Nem hagyhatunk el semmit az F''-beli függőségek baloldaláról, ezért F''' = F''. A minimális fedés az F\* = F''' függéshalmaz.