A) İlişkisel bağımlılık kanonik kapanışı:

1-) Bütün kombinasyonları bul

2-) Bu kombinasyonlar F^+ yani kapanıştır.

B-) Artıklık Algoritması

1-) Sağda birden fazla nitelik varsa ~~g~~ ayrılır.

örn

$a \to bd$

$a \to b$

$a \to d$

2-) Daha sonra elde ki ilişkisel bağımlılıkların F^+ hesaplanır. Eğer F^+ için sonuç diğer ilişkisel bağımlılıkla aynıysa o zaman listeden atılır.

örn

$\cancel{a \to b}$ $\quad a^+ = T\{a, d, \underline{c}, \underline{b}\}$

$a \to d\cancel{c}$

$\cancel{c} \to b$

$a \to b$ atılır. !!

C-) Kanonik Örtü:

ilk önce sağ taraf parçaları sonra bu parçaların sonra sol taraf parçaları. Sol tarafı parçaların doğrunun aynısı aynı F^+'da alınır. Eğer sol taraf nitelikleri hangi biri yine herhangi bir değer tek başına ulaşılıyorsa listeden diğer nitelik atılır.

örn

$b \to c$ $\quad b \to a$

$c \to a$

$bc \to a$ $\quad c \to a$ kalır

$b \to ca$

$c \to a$

$b^+ = T\{b, c, \underline{a}\}$

C-) Kanonik Devam;

2. adım: Ayrıştırılan bağımlılık ilişkileri artıklık algoritması uygular.

3. Adım: En son artıklıklardan temizlenen F ilişkisel bağımlılık kümesi tekrar birleştirilir ve bu Fc olur (Kanonik ö.t.).

D-) İlişki Anahtarları:

R(A,B,C)

Süper anahtarlar:

F+ nın hesaplanan ilişkisel bağımlılıklar yine R kümesinin hepsini varsayarsa süper anahtardır.

Aday Anahtar (Anahtar):

Bulunan süper anahtarın sol tarafta herhangi bir alt kümesi yine ~~bul~~ süper anahtar değilse Aday Anahtardır.

Not: Her anahtar süper anahtardır. Her süper anahtar anahtar değildir.

E-) 2NF

=> Bütün değer alanları yalın ise 2NF'dir

örnek R(A,B,C)

F-) 2NF:

1-) 1NF dacak

2-) Kısmi ilişkisel bağımlı olmayacak.

Yani: Asal olmayan tüm nitelikler tam anahtarlara tam ilişkisel bağlı olmasın istiyor.

AB → C
A → C

} oldukça Kısmidir ve 2NF değildir.

NOT: 2NF'de F+ hesaplanır süper Anahtarlar bulunur.

G-) 3NF

1-) 2NF karşılanacak

2-) Geçişsiz olacak

* Asal olmayan hiçbir nitelik hiçbir anahtara geçişli bağımlı değilse ~~yine~~ 3NF'tir.

Örnek

$ab \to cde$
$c \to e$

$c \to e$ ile geçiş olduğundan 3NF değildir.

H-) BCNF

1-) 3NF sağlanacak

2-) Önemsiz ilişkisel bağımlılık olmayacak yani sol taraftakilerin hepsi anahtar ile asal nitelik olacak $R(A,B,C,D)$

Örnek

$AB \to C$
$BC \to D$
$CD \to A$
$AD \to B$

Anahtarlar = $\{AB, BC, CD, AD\}$

Asal Nitelikler = $\{A, B, C, D\}$

Tüm anahtarlar aynı zamanda asal nitelik olduğundan BCNF'tir.

NOT: Anahtarlar F^+ içerisinde süper anahtar olmasından bulunur. Tüm ilişkisel bağımlıdır. O yüzden anahtarda Asal nitelikler ile Anahtarların harfleridir.

Sol taraf anahtarlardan oluşmalı ve bu anahtarlar asal nitelikler kümesi olmalı.

NOT: Bütün Sol Taraflar ANAHTAR OLACAK

I-) İlişkisel Bağımlılıkların Korunması ve Yıtımsızlık

Yıtımsızlık:

1. Kural:

R'deki nitelliklerin her biri en az bir R'de bulunmalıdır.

2. Kural:

Ayrıştırma yıtımsız birleştirme ayrıştırması olmalıdır. yani her R ilişkisinin R'lere göre izdüşümlerin doğal birleştirmesi R'ye eşit olmalıdır.

Örnek R(a, b, c) F: a → b

	A	B	C
R_1	a_1	a_2	b_{13}
R_2	b_{21}	a_2	a_3

bu satırda full a olması gerekirdi

Bu satırda da full a olması gerekirdi

2 sütunda da a full olmadığından Yıtımsız Değildir.

İlişkisel Bağımlılıkların Korunması

→ İlişkisel bağımlılık kısıtının F^+'ı hesaplar. Şimdiki bilgimize göre Adaylar Anahtarlar bulunur. F ilişkisel bağımlılık kısıtına ekler sonrasında örneğin $R_1(A, B, C)$, $R_2(A, E)$, $R_3(C, D, E)$ sırayla F_1 düşüp R_1'de yine F'e göre türetmek yani ilişkisel bağımlılıklar oluşturur. Bu işlem bütün R'ler için yapılır. Bulunan tüm ilişkisel bağımlılıklar tekrar G adı altında birleştirilir ve eğer G'de olup F'de yoksa ~~yada F'de olup G'de~~ hiçbir şekilde ulaşamıyorsak Bu ilişkisel bağımlılık korunmuyordur.

j-) 3NF'ye AYRIŞTIRMA:

R(A, B, C, D)

F:

$A \to BC$

$C \to A$

$B \to D$

$CD \to A$

1 NF Mi:

Evet Yalın

2 NF Mi:

F^+: $A^+ = T\{A, B, C, D\}$ S.A ve Anahtar

$C^+ = T\{A, B, C, D\}$ S.A ve Anahtar

$B^+ = T\{B, D\}$ X

$CD^+ = T\{C, D, A, B\}$ S.A ama anahtar değil X

$BD^+ = T\{B, D\}$ X

Anahtarlar: $\{A, C\}$ asal olmayanlar $= \{B, D\}$

asal Nitelikler: $\{A, C\}$

asal nitelikler: aynı zamanda anahtar. Anahtarlar tekli olduğu için kısmi olması söz konusu olamaz.

2 NF'dir ✓

3 NF Mi:

$A \to C$
$C \to A$ } Geçişli olduğundan 3NF değildir.

j-) 3NF Ayrıştırma (Devam):

1-) T={} isimli boş bir home oluştur. Sonra sırayla dın bağlılıkları R1 R2 diye isimlendirip eğer T'de yoksa ekle:

F: $A \to BC$
$C \to A$
$B \to D$
$CD \to A$

R1 = (A, B, C) not in T then put in T

T = {R1
R2 (C, A) in T ×

T = {R1
R3 (B, D) ✓

$A \to BC$
$B \to D$

T = {R1, R3}

R4 (C, D, A)

Not: F değil Fc hesaplanıp Fc algoritmaya verilir.

Fc: $\cancel{A \to B}$ $A \to C$ $C \to A$ $B \to D$ $\cancel{CD \to A}$

C^+ = T{C, A, $C \to A$

Artıklık:

$A \to D$? $A^+ = T\{A, C\}$ × $A \to C$? $A^+ = T\{A, B, D\}$

$C \to A$ = $A^+ = T\{A \ldots C\}$ $\cancel{a \to C}$ atılır

$B \to D$? $B^+ = T\{B\}$

Fc: $A \to BC$ $B \to D$
$C \to A$

R1 (A, B, C) T = {R1, R2
R2 (B, D)
R3 (C, A)

R1: $A \to BC$
R: $B \to D$

NOT D Anahtar olmadığı için 3NF'de

K-) BCNF Ayrıştırma:

1-) R(A, B, C, D, E) gibi herhangi bir ilişkisel bağımlılık olsun.

ilk önce $R_1 = R$ şeklir $T = \{R_1\}$ yapılır.

2.) Adım

F^+ hesaplanır.

Burada anahtar bulunan anahtarlar F'ye eklenir.

3.) F^+ kümesinde Anahtar olmayan sırasıyla K 1 arttırılır R ile birlikte: R_1 R_2 ... diye ekleniz eğer eklenen R'nin sağ tarafı R_1'de ~~o değer~~ varsa o değer R_1'den silinir.

NOT: F^+ hesaplamamın sebebi: Sol taraftaki anahtarları tespit etmek

Örn/ R(A, B, C, D, E)

F:
$A \to CDE$
$E \to B$
$CD \to E$

F^+:
$A^+ = T\{A, C, D, E, B\}$ SP ve ✓
$E^+ = T\{E, B\}$ ×
$CD^+ = T\{C, D, E, B\}$

Anahtar: {A}
asal Nitelikler {A}

E ve CD anahtar olmadığından BCNF değildir.

Şimdi: Algoritmaya anahtar olmayan ekleniz.

K=1 R_1 $R_1\{R\}$ $R_1(A, \cancel{B}, C, D, \cancel{E})$

K=2 R_2 anahtar değil $R_2(E, B)$

NOT: Sade $E \to B$ için sağ tarafta R_1'de varsa R_1'den silinir. Yoksa R_2'ye B eklenmezdi.

K=3 $R_3(C, D, E)$

$R_1(A, C, D)$
$R_2(E, B)$
$R_3(C, D, E)$

F1: $A \to CD$
F2: $E \to B$
F3: $CD \to E$

Hareket (Transaction)

Veri tabanındaki verilere erişen ve veriler üzerinde çoğunlukla değişiklik yapan bir program kesimidir.

Begin transaction - End transaction

- Bir banka hesabından bir diğerine para aktarma, bir rezervasyon işlemi, bir stok işlemi v.b

Hareketin taşıması gereken özellikler:

- Bölünmezlik (Atomicity)
- Tutarlılık (Consistency)
- Ayrılma (Isolation)
- Kalıcılık (Durability)

1-) Bölünmezlik (Atomicity):

Hareketi oluşturan işlemlerin, uygulama açısından bölünmemesi gerektiğini gösterir.

örnek

```
Begin transaction;
    Read(A);
    A ← A-100;
    Read (B);
    B ← B+100
    Write (B);
end transaction
```

Bankadaki A hesabından B hesabına 100 TL para aktarın bu hareketin bölünmemesi gerektiği açıktır.

2-) Tutarlılık :

Hareket, veri tabanını tutarlı bir durumdan, tutarlı bir başka duruma taşımalıdır.
Bir önceki örnek için A+B değerinin hep sabit kalmasıdır.

3-) Ayrılma : (isolation)

Birden çok hareket birlikte uygulandığında, hareketler birbirinden bağımsız olarak uygulanabilmelidir.
Ht1 ve Ht2 hareketleri birlikte uygulanıyorsa, Ht1 için her şey sanki Ht2 hiç başlamamış ya da tamamlanmış gibi görünmelidir.

4-) Kalıcılık (durability) :

Hareket tamamlandıktan sonra, hareket tarafından veri tabanında yapılan değişikliklerin kalıcılığı sağlanmalıdır. Bir arıza durumunda değişiklikler yok olmamalıdır.

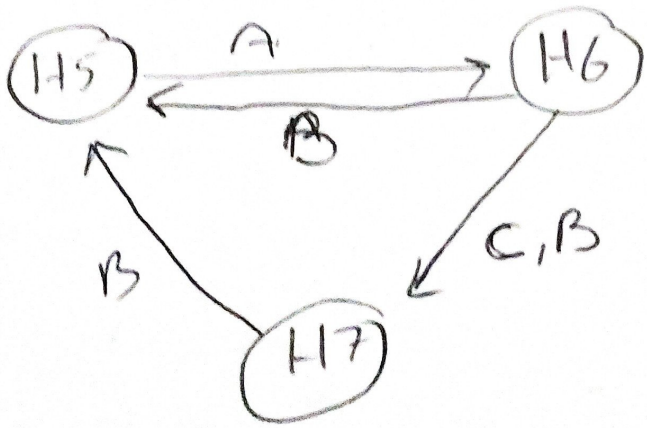
= Hareket Durumları =

Bütün hareketler başarıyla uygulanamayabilir.
İşletildi (Commit edildi) : Uygulanması başarıyla sonuçlanan hareketler için.
Durdurulduu (abort edildi) : Uygulanması başarıyla sonuçlanmayan hareket için kullanılan ifade.

Roll Back : Abort edilen hareketler için bütünlüğünün sağlanması için hareketin geri alınmasıdır.

Serileştirilebilir Hedef Ağaç

	H5	H6	H7
		Read(C) Read(B) Write(B)	
			Read(B); Read(C);
	Read(A); Write(A);		
			Write(B); Write(C);
		Read(A)	
	Read(B) Write(B);		
		Write(A)	



Burada döngü var o yüzden serileştirilemez.

Bir $R = (ABCDE)$ ilişkisi için işlevsel bağımlılıklar

$F_c = (A \to C, B \to C, C \to D, DE \to C, CE \to A)$

a) Ayrıştırma yitimsiz midir?

$R_1(AD)$

$R_2(AB)$

$R_3(BE)$

$R_4(CDE)$

$R_5(AE)$

j \ i	1 A	2 B	3 C	4 D	5 E
R_1	a_1		a_3	a_4	a_5
R_2	a_1	a_2	a_3	a_4	
R_3	a_1	a_2	a_3	a_4	a_5
R_4	a_1		a_3	a_4	a_5
R_5	a_1		a_3	a_4	a_5

R_3 komple a olduğundan yitimsizdir.

b) İşlevsel bağımlılıklar korunuyor mu?

F^+:

$A^+ = T\{A, C, D\}$ x

$B^+ = T\{B, C, D\}$ x

$C^+ = T\{C, D\}$ x

$DE^+ = T\{D, E, C, A\}$ x

$CE^+ = T\{C, E, A, D\}$

$BE^+ = T\{B, E, C, D, A\}$ SP ve AA

$A \to CD$

$B \to CD$

$C \to D$

$DE \to CA$

$CE \to AD$

$BE \to CDA$

F_1: (R_1)

$A \to D$

F_2: (R_2)

$\emptyset$

F_3: (R_3)

$\emptyset$

F_4: (R_4)

$DE \to C$

$CE \to D$

F_5: (R_5)

$\emptyset$

R_2, R_3, R_5 F_c'de tanımlı olmadığından ilişki korunmuyor.