Bölüm4: Bütünlük Kısıtlamaları ve İlişkisel Tasarım

4.1. Bütünlük Kısıtlamaları

➢ Veri tabanının "tutarlı bir bütün" oluşturması; yanlış, eksik, birbiriyle çelişen, tutarsız veri içermemesi istenir. Bunu sağlamak için tanımlanan her türlü kısıtlamaya bütünlük kısıtlaması adı verilir.

- ➢ Her veri tabanı ile ilgili çok sayıda bütünlük kısıtlaması vardır. Bu kısıtlamaların:
- bir kesimi veri tabanının yapısında yer alır,
- bir kesimi kural olarak tanımlanır ve denetimleri işletim aşamasında yapılır,
- bir kesimi uygulama programları içinde yer alır,
- bir kesiminin ise kullanıcılar tarafından bilindiği varsayılır ve kullanıcıların bu kurallara uyması beklenir.

➢ Uygulama programlarında yer alan ya da kullanıcıların uyması gereken bütünlük kısıtlamalarına uyumun yüzde yüz olması beklenemez

➢ Veri tabanı bütünlüğünün sağlanması hem tasarımı hem de uygulamaları ilgilendiren bir konudur.

4.2. Alan Kısıtlamaları

➢ Alan kısıtlamaları her niteliğe bir alan eşlenmesi ve niteliğin alabileceği değerlerin bu alandaki değerlerle sınırlanması ile ilgilidir.

➢ Ancak Veri Tabanı Yönetim Sistemlerinde genellikle alanların tam olarak tanımlanması yerine tür tanımları yapılır ve değerlerle ilgili kimi kısıtlamalar tanımlanır.

- Alan türleri : tamsayı, kesirli sayı, karakter, değişken uzunluklu karakter, tarih, parasal değer, ..vb.
- Değer sınırları : alt sınır değeri, üst sınır değeri, ..vb.
- Boş (null) değer : nitelik değerinin eksik olup olamıyacağı.

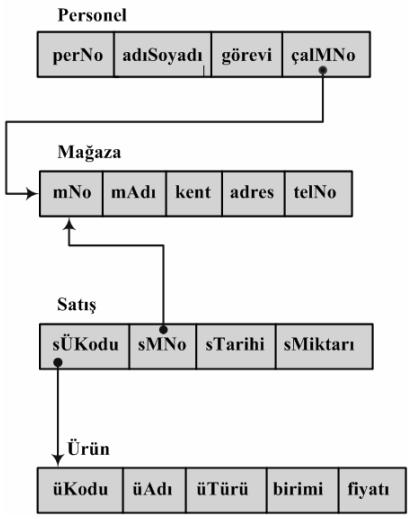
4.3. Referans Kısıtlaması

➢ Referans kısıtlaması bir ilişkideki (A) kimi niteliklerin alabileceği değerlerinin, bir başka ilişkideki (B) kimi niteliklerin varolan değerleri ile sınırlanmasıdır.

➢ Örnek: Ürün (<u>üKodu</u>, üAdı, üTürü, birimi, fiyatı)
Mağaza (<u>mNo</u>, mAdı, kent, adres, telNo)
Satış (<u>sÜKodu, sMNo</u>, sTarihi, sMiktarı)
Personel (<u>perNo</u>, adıSoyadı, görevi, çalMNo)

1. Satış ilişkisindeki sÜKodu niteliği, Ürün ilişkisinin birincil anahtarını (üKodu) referans göstermektedir. sÜKodu niteliği Satış ilişkisinde yabancı anahtardır.
2. Satış ilişkisindeki sMNo niteliği, Mağaza ilişkisinin birincil anahtarını (mNo) referans göstermektedir. Satış ilişkisinde sMNo niteliği de yabancı anahtardır.
3. Personel ilişkisindeki çalMNo niteliği, Mağaza ilişkisinin birincil anahtarını (mNo) referans göstermektedir. Personel ilişkisinde çalMNo yabancı anahtardır.

Örnek Veri Tabanı İçin Referans Çizeneği



4.4. Nitelikler Arası Bağımlılıklar

➢ Nitelikler arası bağımlılık, veri tabanındaki bazı niteliklerin birbirinden bağımsız olmaması; bu niteliklerin değerlerinin birbirinden bağımsız olarak belirlenememesi anlamına gelmektedir.

➢ Eğer ilişkiler oluşturulurken nitelikler arası bağımlılıklar dikkate alınmazsa, veri tabanında bir dizi aykırılık oluşabilir. Bu durumda veri tabanının bütünlük ve tutarlılığı korunamaz.

➢ Nitelikler arası bağımlılıklar dikkate alınmadan (kötü) tasarlanmış örnek bir çizelge:

Satıcı (üKodu, fNo, fAdı, fAdresi, sFiyatı)

Bu ilişki şemasının yol açabileceği sorunlar:

a. Veri yinelemesi.
b. Günleme aykırılığı.
c. Ekleme Aykırılığı.
d. Silme Aykırılığı.

➢ Yukarıdaki tek ilişki yerine aşağıdaki iki ilişki oluşturulursa, sayılan aykırılıkların ortadan kalktığı görülür.

Firma (fNo, fAdı, fAdresi)
SÜrün (fNo, üKodu, sFiyatı)

4.4.1. İşlevsel Bağımlılık : Temel Kavramlar

➢ $R = \{A1, A2, A3, \ldots, An)$ $X \subseteq R$, $Y \subseteq R$

Eğer X nitelik değerleri aynı olan tüm çoklularda, Y nitelik değerlerinin de aynı olması gerekiyorsa: $t1[X] = t2[X] \Rightarrow t1[Y] = t2[Y]$

X Y'yi işlevsel belirler, ya da Y X'e işlevsel bağımlıdır denir ve $X \rightarrow Y$ gösterimi kullanılır.

➢ İşlevsel bağımlılık kavramı iki düzeyde düşünülebilir:

- Kavramsal düzeyde.
- Olgu (örnek) düzeyinde.

➢ Kısaca işlevsel bağımlılık deyince kavramsal düzeydeki işlevsel bağımlılık anlaşılacaktır.

➢ Örnek: R(A, B, C, D) ilişki şemasına göre oluşturulmuş aşağıdaki r ilişki olgusu veriliyor.

r :

A	B	C	D
1	a	x	e
2	a	y	b
2	b	x	c
3	c	x	c
4	a	x	e

Bu ilişki olgusunun sağladığı işlevsel bağımlılıklar şunlardır.

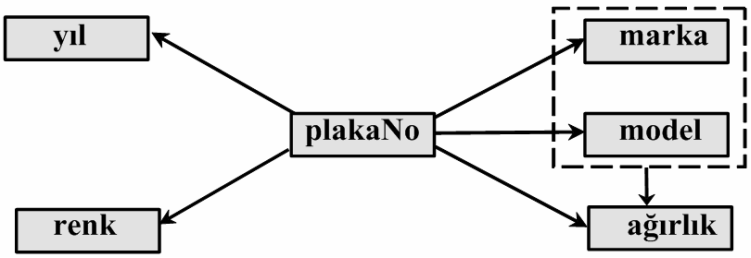
$D \rightarrow C$
$A, B \rightarrow C, D$
$A, C \rightarrow B, D$
$A, D \rightarrow B, C$
$B, C \rightarrow D$

➢ Örnek: Taşıt (plakaNo, marka, model, yıl, ağırlık, renk)

İşlevsel bağımlılıklar:

plakaNo $\rightarrow$ marka, model, yıl, ağırlık, renk
marka, model $\rightarrow$ ağırlık

➢ İşlevsel bağımlılık çizeneği:

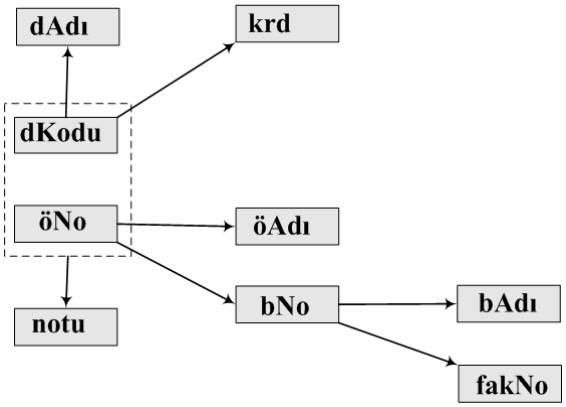


4.4.2. İşlevsel Bağımlılık Türleri

- Kısmi İşlevsel Bağımlılık. Eğer X A'yı belirliyorsa ve X'in en az bir öz altkümesi de A'yı belirliyorsa ($X \rightarrow A$ ve $\exists Z \subset X : Z \rightarrow A$) $X \rightarrow A$ işlevsel bağımlılığına kısmi işlevsel bağımlılık denir.
- Tam işlevsel bağımlılık. Eğer X A'yı belirliyorsa ve X'in hiçbir öz altkümesi A'yı belirlemiyorsa ($X \rightarrow A$ ve $\exists \neg Z \subset X : Z \rightarrow A$) $X \rightarrow A$ işlevsel bağımlılığına tam işlevsel bağımlılık denir.
- Önemsiz İşlevsel Bağımlılık. Eğer X A'yı belirliyorsa ve A X'in bir altkümesi ise ($X \rightarrow A$ ve $A \subseteq X$) $X \rightarrow A$ işlevsel bağımlılığına önemsiz işlevsel bağımlılık denir.
- Önemli İşlevsel Bağımlılık. Eğer X A'yı belirliyorsa ve A X'in bir altkümesi değilse ($X \rightarrow A$ ve $A \subseteq X$) $X \rightarrow A$ işlevsel bağımlılığına önemli işlevsel bağımlılık denir.

- Geçişli işlevsel bağımlılık. Eğer X Y'yi, Y de Z'i belirliyorsa ($X \rightarrow Y$ ve $Y \rightarrow Z$) $X \rightarrow Z$ işlevsel bağımlılığına geçişli işlevsel bağımlılık denir.

Örnek: R (öNo, öAdı, bNo, bAdı, fakNo, dKodu, dAdı, krd, notu)



Önemsiz işlevsel bağımlılık örnekleri:
öNo, öAdı → öAdı
bNo, bAdı, fakNo → bNo, fakNo

Önemli işlevsel bağımlılık örnekleri:
öNo → öAdı
öNo, öAdı, dKodu → bNo, krd
öNo, dKodu → notu

Kısmi işlevsel bağımlılık örnekleri:
öNo, dKodu → krd
öNo, dKodu, dAdı → notu

Tam işlevsel bağımlılık örnekleri:
öNo → öAdı, bNo
bNo → fakNo
öNo, dKodu → noto

Geçişli işlevsel bağımlılık örnekleri:
öNo → fakNo
öNo → bA

- Buna göre en değerli, en çok bilgi taşıyan işlevsel bağımlılıklar önemli, tam ve geçişli olmayan (bu üç özelliği birlikte taşıyan) işlevsel bağımlılıklardır. Örnek R ilişkisi için niteliler arasındaki bu tür işlevsel bağımlılıklar aşağıdakilerdir.

öNo → öAdı, bNo
bNo → bAdı, fakNo
dKodu → dAdı, krd
dKodu, öNo → notu

4.4.3. İşlevsel Bağımlılıklarla İlgili Kimi Tanım, Önerme ve Algoritmalar

A. Bir İşlevsel Bağımlılık Kümesinin Kapanışı

Geniş Tanım: Bir R nitelik kümesi ve bu nitelikler üzerinde tanımlı bir F işlevsel bağımlılık kümesi verildiğinde, F'deki işlevsel bağımlılıklara ek olarak:R'deki tüm önemsiz işlevsel bağımlılıklar ile F'deki işlevsel bağımlılıklardan türetilebilecek tüm kısmi ve geçişli

bağımlılıkları içeren işlevsel bağımlılık kümesine F'nin kapanışı (*closure*) denir ve bu küme F+ olarak gösterilir.

Sınırlı Tanım: Bir R nitelik kümesi ve bu nitelikler üzerinde tanımlı bir F işlevsel bağımlılık kümesi verildiğinde, F'deki işlevsel bağımlılıklara ek olarak:
Bu bağımlılıklardan türetilebilecek önemli ve tam bağımlılıkları içeren işlevsel bağımlılık kümesine F'nin kapanışı (*closure*) denir ve bu küme F+ olarak gösterilir.

Örnek: R(A, B, C, D, E)

$F = \{ A \rightarrow B , \; B \rightarrow C , \; CD \rightarrow E \}$

$F+ = \{ A \rightarrow BC , \; B \rightarrow C , \; AD \rightarrow E , \; BD \rightarrow E , \; CD \rightarrow E \}$

B. İşlevsel Bağımlılıkları Türetme Kuralları

Temel Kurallar (Armstrong aksiyomları):

1. Dönüşlülük (*reflexitivity*) kuralı:
   $Y \subseteq X \Rightarrow X \rightarrow Y$
2. Arttırma (*augmentation*) kuralı:
   $X \rightarrow Y \Rightarrow XZ \rightarrow Y$
3. Geçişlilik (*transitivity*) kuralı:
   $X \rightarrow Y$ ve $Y \rightarrow Z \Rightarrow X \rightarrow Z$

Diğer Kuralar:

4. Birleşim (*union*) kuralı:
   $X \rightarrow Y$ ve $X \rightarrow Z \Rightarrow X \rightarrow YZ$
5. Ayrıştırma (*decomposision*) kuralı:
   $X \rightarrow YZ \Rightarrow X \rightarrow Y$ ve $X \rightarrow Z$
6. Sözde geçişlilik (*pseudo transitivity*) kuralı:
   $X \rightarrow Y$ ve $YZ \rightarrow W \Rightarrow XZ \rightarrow W$

➢ **Örnek 4.8. R (A, B, C, D, E, G) F : $A \rightarrow BD$, $BC \rightarrow EG$, $D \rightarrow E$**

$DG \rightarrow E$ (artırma kuralına göre)

$A \rightarrow E$ (ayrıştırma ve geçişlilik kurallarına göre)

$A \rightarrow ABD$ (dönüşlülük ve birleşim kurallarına göre)

$A \rightarrow B$ ve $A \rightarrow D$ (ayrıştırma kuralına göre)

$AC \rightarrow EG$ (ayrıştırma ve sözde geçişlilik kurallarına göre)

C. Kanonik Örtü

➢ Bir F işlevsel bağımlılık kümesinin kanonik örtüsü (Fc), F'ye eşdeğer olan ve en az sayıda işlevsel bağımlılık içeren bir kümedir. F ve Fc kümelerinin eşdeğerliği F deki her işlevsel bağımlılığın Fc deki işlevsel bağımlılıklardan türetilebilmesi anlamındadır.

➢ Bir işlevsel bağımlılık kümesinin kanonik örtüsünün sağlaması gereken koşullar şunlardır:
   K1. Fc deki işlevsel bağımlılıkların sol tarafları birbirinden farklı olmalıdır.
   K2. Fc deki işlevsel bağımlılıklardan hiçbirinin sol tarafında artık nitelik bulunmamalıdır
   K3. Fc deki işlevsel bağımlılıklardan hiçbirinin sağ tarafında artık nitelik bulunmamalıdır

➢ Yukarıdaki koşullardan anlaşılabileceği gibi Fc F'ye eşdeğer olan ve önemsiz, kısmi ya da geçişli hiçbir işlevsel bağımlılık içermeyen bir kümedir (K2 ve K3). Ayrıca, Fc de sol tarafı aynı olan işlevsel bağımlılıkların birleştirilmiş olması gerekmektedir (K1).

➢ R (A, B, C, D) $F = \{B \rightarrow C , \; C \rightarrow A , \; BC \rightarrow A , \; B \rightarrow A\}$

$Fc = \{B \rightarrow C , \; C \rightarrow A\}$

D. Artıklık Algoritması

➢ Bir F işlevsel bağımlılık kümesi verildiğinde, eğer kümedeki işlevsel bağımlılıklardan biri (f), kümedeki diğer işlevsel bağımlılıklardan türetilebiliyorsa, bu işlevsel bağımlılık (f) kümede artıktır.

Artıklık algoritması

Başlangıçta T = { X } yap

1. "F – { f }"teki her $W \rightarrow Z$ işlevsel bağımlılığı için:
   eğer $\{W\} \subseteq T$ ise $\Rightarrow$ $T = T \cup \{Z\}$ yap
2. T değiştiği sürece 2. adımı tekrarla
3. Sonuçta eğer $Y \subseteq T$ ise $(f : X \rightarrow Y)$ işlevsel bağımlılığı F'de artıktır.

➢ Bir işlevsel bağımlılık kümesindeki işlevsel bağımlılıklardan artık olanlar aranırken iki noktaya dikkat etmek gerekir.

1. Eğer $X \rightarrow Y$ işlevsel bağımlılığında Y birden çok nitelikten oluşuyorsa, artıklık bu niteliklerden herbiri için ayrı ayrı araştırılmalıdır.
2. F = {f1, f2, f3,…., fn} işlevsel bağımlılık kümesinde artık işlevsel bağımlılıklar için birden çok seçenek bulunabilir.

Örnek: R (A, B, C, D, E, G) nitelik kümesi üzerinde tanımlı aşağıdaki işlevsel bağımlılık kümesi verilmiş olsun.

$F : A \rightarrow BCDE$
$G \rightarrow BD$
$BC \rightarrow E$
$CG \rightarrow A$
$BDE \rightarrow ACG$

➢ F'de $A \rightarrow C$ ve $CG \rightarrow A$ işlevsel bağımlılıkları artıktır. Buna göre F'nin kanonik örtüsü aşağıdaki gibi bulunur.

$Fc : A \rightarrow BDE$
$G \rightarrow BD$
$BC \rightarrow E$
$BDE \rightarrow ACG$

E. Türetilebilirlik Algoritması

➢ Bir F işlevsel bağımlılık kümesi verildiğinde, bu kümedeki işlevsel bağımlılıklardan $f : X \rightarrow Y$ işlevsel bağımlılığının türetilip türetilemeyeceğini bulan bir algoritmadır.

➢ **Türetilebilirlik algoritması**

1. Başlangıçta $T = \{ X \}$ yap
2. F'deki her $W \rightarrow Z$ işlevsel bağımlılığı için:
   Eğer $\{W\} \subseteq T$ ise $\Rightarrow$ $T = T \cup \{Z\}$ yap
3. T değiştiği sürece 2. adımı tekrarla
4. Sonuçta eğer $Y \subseteq T$ ise $(f : X \rightarrow Y)$ işlevsel bağımlılığı F'den türetilebilir.

➢ Eğer $X \rightarrow Y$ işlevsel bağımlılığında Y birden çok nitelikten oluşuyorsa, bu niteliklerden her birinin türetilebilirliği ayrı ayrı araştırılmalıdır.

F. Bir Nitelik Kümesinin Kapanışı

➢ R(A1, A2, A3, …. , An) nitelik kümesi üzerinde tanımlı bir F işlevsel bağımlılık kümesi verilmiş olsun. Eğer X R'nin bir altkümesi ise $(X \subseteq R)$, X'in kapanışı (X+) X'e işlevsel bağımlı (ya da X'in belirlediği) niteliklerin tümüdür.

X+ hesaplama algoritması

1. Başlangıçta $T = \{ X \}$ yap
2. F'deki her $W \rightarrow Z$ işlevsel bağımlılığı için:
   eğer $\{ W\} \subseteq T$ ise $\Rightarrow$ $T = T \cup \{Z\}$ yap
3. T değiştiği sürece 2. adımı tekrarla

Sonuçta X+ = T olarak bulunur.

Örnek: R(A, B, C, D) F = {B → C , C → A}

A+ = A (AB)+ = ABC

B+ = BCA (BD)+ = ABCD

C+ = CA

4.5. İlişki Anahtarları

- Süper anahtar. Eğer bir nitelik altkümesi (K) ilişkideki tüm nitelikleri işlevsel belirliyorsa, başka bir deyişle K'nın kapanışı R'ye eşitse, bu nitelik altkümesi ilişkinin süper anahtarıdır.

  $K \subseteq R : K \rightarrow R \Rightarrow K+ = R$'dir ve K ilişkinin süper anahtarıdır.

- Anahtar ya da anahtar adayı. Eğer bir nitelik altkümesi (K) ilişkideki tüm nitelikleri işlevsel belirliyorsa ve de K'nın hiçbir altkümesi tüm nitelikleri belirlemiyorsa, K ilişkinin anahtar adayı, ya da kısaca anahtarıdır.

  $(K \subseteq R : K \rightarrow R)$ ve $(\exists \neg K1 \subseteq K : K1 \rightarrow R) \Rightarrow$ K ilişkinin anahtarıdır.

- Her anahtar bir bütünlük kısıtlamasıdır. İlişkinin tüm örneklerinde, her anahtarın değeri ilişkinin tüm çoklularında birbirinden farklı olmalıdır.

Örnek R(A, B, C, D, E) ilişki şeması üzerinde tanımlı

F = { A → C , B → D , D → AE, CE → B } işlevsel bağımlılık kümesi veriliyor.

İlişkinin anahtarları (ya da nahtar adayıları): B, D, AE ve CE

4.6. İlişkiler İçin Normal Biçimler

4.6.1. Birinci Normal Biçim (1NF)

- 1NF İlişki Tanımı. Eğer bir ilişkideki tüm niteliklerin değer alanları yalın değer alanları ise (hiçbir niteliğin hiçbir değeri bir dizi, bir matris ya da karmaşık bir değer değilse) ilişki Birinci Normal Biçimdedir.
- Birinci Normal Biçimde olmayan ilişkiye kısaca N1NF (*Non 1NF*) ilişki denir.
- Örnek: Öğrenci (öNo, öAdı, ders (dAdı, notu)) N1NF ilişki

  ÖğrDers (öNo, ÖAdı, dAdı, Notu) 1NF ilişki

ÖĞRENCİ

		DERS	
ÖNO	ÖADI	DADI	NOTU
123	Ali	Mat	87
		Fiz	69
		Kim	93
207	Kaya	Kim	64
186	Nur	Fiz	75
		Mat	45
316	Ayşe	Bio	59
		Kim	87
		Mat	60
		Fiz	77
		İng	98

ÖĞRDERS

ÖNO	ÖADI	DADI	NOTU
123	Ali	Mat	87
123	Ali	Fiz	69
123	Ali	Kim	93
207	Kaya	Kim	64
186	Nur	Fiz	75
186	Nur	Mat	45
316	Ayşe	Bio	59
316	Ayşe	Kim	87
316	Ayşe	Mat	60
316	Ayşe	Fiz	77
316	Ayşe	İng	98

4.6.2. İkinci Normal Biçim (2NF)

- 2NF İlişki Tanımı. Eğer bir ilişki Birinci Normal Biçimde (1NF) ise ve asal olmayan (hiçbir anahtarda yer almayan) niteliklerden hiçbiri anahtarlardan hiçbirine kısmi işlevsel bağımlı değilse bu ilişki İkinci Normal Biçimdedir.
- **Örnek**: Satıcı (<u>üKodu, fNo</u>, fAdı, fAdresi sFiyatı)

  İlişki 1NF bir ilişkidir (2NF değil çünkü asal olmayan fAdı ve fAdresi anahtara

kısmi işlevsel bağımlıdır).

4.6.3. Üçüncü Normal Biçim (3NF)

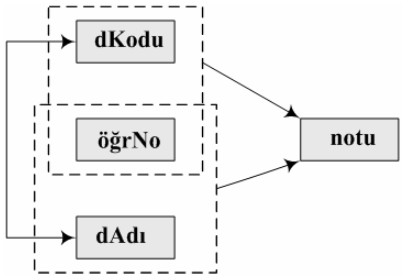
- 3NF İlişki Tanımı. Eğer bir ilişki 2. Normal Biçimde ise, ve asal olmayan hiçbir nitelik hiçbir anahtara geçişli bağımlı değilse, bu ilişki 3. Normal Biçimdedir.
- Örnek: Taşıt (plakaNo, marka, model, yıl, ağırlık, renk)

İlişki 2NF bir ilişkidir (3NF değil çünkü asal olmayan "ağırlık" anahtara kısmi işlevsel bağımlıdır).

4.6.4. Boyce Codd Normal Biçimi (BCNF)

- BCNF İlişki Tanımı. Eğer bir ilişki 1. Normal Biçimde ise ve tüm belirleyenler ilişkinin anahtarı ise bu ilişki Boyce Codd Normal Biçimindedir. Buna göre, eğer K ilişkinin bir anahtarını gösteriyorsa, BCNF ilişkideki tüm işlevsel bağımlılıklar $K \rightarrow X$ biçimindedir.
- Birden çok anahtarı bulunan 3NF bir ilişkide asal nitelikler birbirinden bağımsız olmayabilir. İlişkinin anahtarlarından en az ikisi birden çok nitelikten oluşuyorsa ilişki BCNF olmayabilir.
- Örnek: ÖğrDers (öğrNo, dKodu, dAdı, Notu)

F : dKodu → dAdı
dAdı → dKodu
dKodu, öğrNo → notu
dAdı, öğrNo → notu



- İlişkinin anahtarları: (dKodu, öğrNo) ve (dAdı, öğrNo)
  İlişkinin biçimi : 3NF (BCNF değil)

4.7. İlişkilerin Ayrıştırılması

- R ilişki şeması ve nitelikler arasında tanımlı F işlevsel bağımlılık kümesi verildiğinde R nin ayrıştırılması, R yerine aşağıdaki koşulları sağlayan {R1, R2, R3, … , Rn} ilişki kümesi konulmasıdır.

K1 : R deki niteliklerin her biri en az bir Ri'de bulunmalıdır : $R = R1 \cup R2 \cup R3 \cup \ldots Rn$

K2 : Ayrıştırma yitimsiz-birleştirme ayrıştırması (*lossless-join decomposition*) olmalıdır
($r = \pi R1\ (r)$ $\pi\ R2\ (r)$ $\pi R3\ (r)$ ….. $\pi Rn\ (r)$

K3 : Ayrıştırma F'deki işlevsel bağımlılıkları korumalıdır.

- Kısaca ayrıştırmanın geçerli bir ayrıştırma olması için:
  Yitimsiz-birleştirme ayrıştırması olması, ve
  İşlevsel bağımlılıkları koruması
  gerekli ve yeterlidir (K1 artık bir koşul).

4.7.1. Yitimsiz-Birleştirme Ayrıştırması

A	B	C
a_1	b_1	c_1
a_2	b_2	c_3
a_3	b_1	c_2
a_4	b_2	c_4

a) **r**

B	C
b_1	c_1
b_2	c_3
b_1	c_2
b_2	c_4

c) $r_2 = \pi_{BC}(r)$

A	B
a_1	b_1
a_2	b_2
a_3	b_1
a_4	b_2

b) $r_1 = \pi_{AB}(r)$

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2
a_2	b_2	c_3
a_2	b_2	c_4
a_3	b_1	c_2
a_3	b_1	c_1
a_4	b_2	c_3
a_4	b_2	c_4

d) $r_3 = r_1 \bowtie r_2$

İkili Bir Ayrıştırma İçin Yitimsizlik Koşulu

- Tanım. Bir R ilişki şeması ve nitelikler arası F işlevsel bağımlılık kümesi verildiğinde, R'nin {R1, R2} ikili ayrıştırması, eğer aşağıdaki koşullar sağlanıyorsa yitimsizdir.

  K1 : R1 ve R2'de ortak nitelik ya da nitelikler bulunmalıdır.
  (Eğer R(X, Y, Z) ise R1 (X, Y) ve R2 (X, Z) olmalıdır).

  K2 : R1 ve R2'deki ortak nitelik ya nitelikler (X) R1 ve R2'den en az birinin anahtarı olmalıdır. ($X \rightarrow$ R1 ya da $X \rightarrow$ R2 işlevsel bağımlılıklarından en az biri F+ da bulunmalıdır).

- **Örnek**: R(A, B, C, D, E)

  F : $A \rightarrow BC$
  $D \rightarrow B$
  $E \rightarrow A$
  $CD \rightarrow E$

  R1 (A, B, C) R2 (A, D, E) ayrıştırması yitimsiz midir?
  R1 ve R2 ilişkilerindeki ortak nitelik A'dır.
  A R1 ilişkisinin anahtarı olduğu için bu ayrıştırma yitimsizdir.
  Ancak bu ayrıştırma işlevsel bağımlılıkları koruyamamaktadır.

Ayrıştırmaların Yitimsizlik Sınaması İçin Algoritma (Genel)

R (A1 , A2 , A3 , … , An) bir ilişki şeması, ve F işlevsel bağımlılık kümesi için {R1 , R2 , R3 , … , Rk} ayrıştırmasının yitimsiz olup olmadığının bulunması.

1. Her Ri ilişkisi için bir satırı; her Aj için de bir kolonu bulunan bir çizelge oluştur. Satırlara ve kolonlara 1'den başlayarak sıra numarası ver.
2. Çizelgenin i. satır j. kolonundaki elemanına:
   - Eğer Ri ilişkisinde Aj niteliği varsa : aj yaz.
   - Eğer Ri ilişkisinde Aj niteliği yoksa : bij yaz.
3. F'deki her (f : $X \rightarrow Y$) işlevsel bağımlılığı için:
   eğer 2 ya da daha çok satırda, X'i oluşturan kolonlardaki değerler aynı ise:
   bu satırlarda Y'yi oluşturan tüm kolonlardaki değerleri eşitle (eğer eşitlenecek değerlerden en az biri aj ise hepsini aj yap; hiçbiri aj değil hepsi b'lerden oluşuyorsa, aralarından rasgele birini seç ve diğerlerini buna eşitle).
4. Çizelgede değişiklik olduğu sürece, satırlardan biri tüm a'lardan oluşuncaya kadar 3. adımı tekrarla.
5. Sonuçta eğer satırlardan biri tüm a'lardan (a1 , a2 , a3 , .. , ak) oluşuyorsa ayrıştırma yitimsizdir; değilse ayrıştırma yitimsiz değildir.

Örnek R (A, B, C, D, E)

F : $A \rightarrow BC$
$D \rightarrow B$
$E \rightarrow A$
$CD \rightarrow E$

R1 (A, B, C) R2 (A, E) R3 (C, D, E) ayrıştırması yitimsiz midir?

	1 A	2 B	3 C	4 D	5 E
R_1	a_1	a_2	a_3	b_{14}	b_{15}
R_2	a_1	a_2 ~~b_{22}~~	a_3 ~~b_{23}~~	b_{24}	a_5
R_3	a_1 ~~b_{31}~~	a_2 ~~b_{32}~~	a_3	a_4	a_5

4.7.2. Ayrıştırmanın İşlevsel Bağımlılıkları Koruması

- Her işlevsel bağımlılık bir bütünlük kısıtlamasıdır. Veri tabanının bütünlüğünün ve tutarlığının korunması için, yapılan her günleme işlemininde işlevsel bağımlılıkların sağlandığının denetlenmesi gerekir.
- Bu denetimlerin her birinin, birden çok ilişkinin birleştirilmesini gerektirmeden, tek bir ilişki üzerinde yapılabilmesi gerekir.
- Bunun için de R ilişkisi R1 , R2 , R3 , … , Rk ilişkilerine ayrıştırıldığında, F'deki işlevsel bağımlılıklardan her birinin bir Ri'de bulunması, ya da Ri'lerdeki işlevsel bağımlılıklardan türetilebilmesi gerekir.
- İşlevsel Bağımlılıkların Korunması Algoritması

1. F+ nın her Ri üzerindeki izdüşümü bulunur.
   $Fi = \pi Ri\ (F+) = \{ f\ (X \rightarrow Y) :\ f \in F+ \text{ ve } X, Y \subseteq Ri \}$
2. Fi'lerin küme birleşimi bulunur.
   $G = F1 \cup F2 \cup \ldots\ldots \cup Fk$
3. Sonuçta eğer G F'ye eşdeğer ise (F+ = G+) ayrıştırma işlevsel bağımlılıkları korumaktadır. Değilse ayrıştırmada bazı işlevsel bağımlılıklar yitirilmiştir.

Örnek R (A, B, C, D, E)

$F : A \rightarrow BC$
$D \rightarrow B$
$E \rightarrow A$
$CD \rightarrow E$

R1 (A, B, C) R2 (A, E) R3 (C, D, E)
Yitimsiz olduğunu bulduğumuz bu ayrıştırma işlevsel bağımlılıkları koruyor mu?

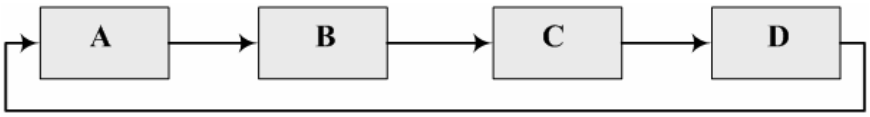
$F+ : A \rightarrow BC$
$D \rightarrow B$
$E \rightarrow ABC$
$CD \rightarrow AE$
$AD \rightarrow E$

$F1 = \{A \rightarrow BC\}$
$F2 = \{E \rightarrow A\}$
$F3 = \{CD \rightarrow E\}$
$G = F1 \cup F2 \cup F3 = \{A \rightarrow BC , E \rightarrow A , CD \rightarrow E\}$

İşlevsel bağımlılıklardan ($D \rightarrow B$) G'de yoktur ve G'deki işlevsel bağımlılıklardan türetilemez. Bu nedenle daha önce yitimsiz olduğunu gördüğümüz bu ayrıştırma işlevsel bağımlılıkları korumamaktadır.

- **Örnek**: R (A, B, C, D)

$F : A \rightarrow B$
$B \rightarrow C$
$C \rightarrow D$
$D \rightarrow A$



R ilişkisinin {R1 (A, B) , R2 (B, C) , R3 (C, D)} ayrıştırması yitimsiz bir ayrıştırmadır. Bu ayrıştırma acaba işlevsel bağımlılıkları koruyor mu?

$F+ : A \rightarrow BCD$
$B \rightarrow CDA$
$C \rightarrow DAB$
$D \rightarrow A\ BC$

$F1 = \{A \rightarrow B , B \rightarrow A\}$
$F2 = \{B \rightarrow C , C \rightarrow B\}$
$F3 = \{C \rightarrow D , D \rightarrow C\}$

$G : A \rightarrow B$
$B \rightarrow AC$
$C \rightarrow BD$
$D \rightarrow C$

($D \rightarrow A$) işlevsel bağımlılığı G'de yoktur. Ancak türetme kuralları ile G'deki işlevsel bağımlılıklardan bu işlevsel bağımlılık türetilebilir ($D \rightarrow C \rightarrow A \Rightarrow D \rightarrow A$). Bu nedenle de bu ayrıştırma işlevsel bağımlılıkları korumaktadır.

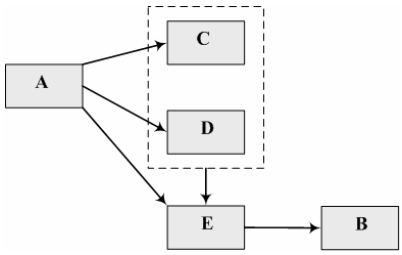
4.7.3. BCNF Ayrıştırma Algoritması

- Bir R ilişki şeması (nitelik kümesi) ve nitelikler arası F işlevsel bağımlılık kümesi verildiğinde, eğer R BCNF bir ilişki değilse, ilişkisinin BCNF ilişkilere ayrıştırılması için aşağıdaki algoritma kullanılabilir.
- **BCNF ayrıştırma algoritması:**

1. R1 = R , k = 1 , T = { R1 }
2. F+ yı hesapla
3. T'deki ilişkilerden BCNF olmayan her Ri için:
   F+ daki işlevsel bağımlılıklardan, Ri'de tanımlı
   önemli her $X \rightarrow Y$ işlevsel bağımlılığı için:
   eğer X Ri'nin anahtarı değilse ($X \rightarrow Ri$ F+ da yoksa)
   Ri'den Y'deki nitelikleri çıkar,
   k'yı 1 arttır,
   T'ye Rk (X, Y) ilişki şemasını ekle.
4. T'deki tüm ilişkiler BCNF oluncaya dek 3. adımı tekrarla.

- BCNF ayrıştırma algoritması ile elde edilen ayrıştırma yitimsiz bir ayrıştırma olur. Ancak ayrıştırmanın işlevsel bağımlılıkları koruma güvencesi yoktur.
- BCNF ayrıştırma algoritması ile elde edilen ayrıştırma yitimsiz bir ayrıştırma olur. Ancak ayrıştırmanın işlevsel bağımlılıkları koruma güvencesi yoktur.
- **Örnek:** R (A, B, C, D, E)

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



- İlişkisinin anahtarı: A
  İlişkinin biçimi : 2NF (BCNFdeğil).

F+ : $A \rightarrow BCDE$
$E \rightarrow B$
$CD \rightarrow BE$

BCNF ayrıştırma:
R1 (<u>A</u>, C, D)
R2 (<u>C, D</u>, E)
R3 (<u>E</u>, B)
Bu ayrıştırma yitimsizdir. Ayrıca ayrıştırma işlevsel bağımlılıkları da korumaktadır.

4.7.4. 3NF Ayrıştırma Algoritması

- Bir R ilişki şeması (nitelik kümesi) ve nitelikler arası F işlevsel bağımlılık kümesi verildiğinde, eğer R 3NF bir ilişki değilse, ilişkisinin 3NF ilişkilere ayrıştırılması için aşağıdaki algoritma kullanılabilir.

➢ 3NF ayrıştırma algoritması:

1. k = 1 , T = { }
2. Fc yi hesapla
3. Fc deki her $X \rightarrow Y$ işlevsel bağımlılığı için:
   eğer T'deki Ri ilişki şemalarından hiçbiri XY niteliklerini içermiyorsa:
   k'yı 1 arttır, T'ye Rk (X, Y) ilişki şemasını ekle.
4. Sonuçta eğer T'deki ilişki şemalarından hiçbiri R'nin anahtarlarından hiçbirini içermiyorsa:
   k'yı 1 arttır, T'ye Rk (K) ilişki şemasını ekle.
   (K R'nin anahtarlarından herhangi biri olabilir)

➢ 3NF ayrıştırma algoritması ile hem yitimsiz olan, hem de işlevsel bağımlılıkları koruyan ayrıştırmalar elde edilir.

➢ **Örnek:** R (A, B, C, D, E, G)

F : $AB \rightarrow CD$
$AC \rightarrow E$
$DE \rightarrow G$

➢ F+ : $AB \rightarrow CDEG$
$AC \rightarrow E$
$DE \rightarrow G$
$ACD \rightarrow G$

➢ Fc = F

➢ 3NF Ayrıştırma:
R1 (<u>A, B</u>, C, D)
R2 (<u>A, C</u>, E)
R3 (<u>D, E</u>, G)

➢ BCNF Ayrıştırmalar:

<u>1. Ayrıştırma</u>
R1 (<u>A, B</u>, C, D)
R2 (<u>A, C</u>, E)
R3 (<u>D, E</u>, G)

<u>2. Ayrıştırma</u>
R1 (<u>A, B</u>, C, D)
R2 (<u>A, C</u>, E)
R3 (<u>A, C, D</u>, G)

➢ Bu ayrıştırmalardan ilki şlevsel bağımlılıkları korumaktadır. İkinci ayrıştırmada ise $DE \rightarrow G$ işlevsel bağımlılığı yitirilmektedir.

4.7.5. BCNF ve 3NF Normal Biçimlerinin Karşılaştırılması

➢ BCNF ilişkiler 3NF ilişkilere göre daha sorunsuz, aykırılıklara yol açmayan ve bakımı daha kolay ilişkilerdir. Ancak BCNF ayrıştırma algoritması, elde edilecek ayrıştırmanın yitimsizliğini güvencelemekle birlikte işlevsel bağımlılıkların korumasını güvencelememektedir. İşlevsel bağımlılıkların korunmaması ise sakıncalıdır. Bu nedenle verilen bir ilişki şeması için, BCNF ayrıştırma algoritması ile, işlevsel bağımlılıkları koruyan bir ayrıştırma elde edilebiliyorsa, bu ayrıştırma öncelikle tercih edilmelidir. Böyle bir ayrıştırma elde edilemiyorsa, 3NF ilişkileri kullanmak daha uygun olabilir.

➢ 3NF ayrıştırma algoritması ile elde edilen ayrıştırmanın hem yitimsiz olması hem de işlevsel bağımlılıkları koruması bir üstünlüktür. Ancak BCNF koşulunu sağlamayan 3NF ilişkilerin kimi sakıncaları vardır. Bu sakıncaların bir bölümü veri tekrarı ve günleme aykırılıklarıdır. Diğer bir bölümü ise "null" değeri kullanılarak aşılabilecek ekleme ve silme aykırılıklarıdır.

4.8. Çok Değerli Bağımlılık ve Dördüncü Normal Biçim

4.8.1. Çok Değerli Bağımlılığın Tanımı

➢ Nitelikler arasındaki tüm bağımlılıklar işlevsel değildir. Nitelikler arasında çok değerli bağımlılık olarak adlandırılan bağımlılıklar da bulunabilir.

➢ Örnek: Personel (çalışan, ücreti, çocuğu)

F : çalışan $\rightarrow$ ücreti

Bu ilişkide "çalılşan" ve çocuğu nitelikleri arasında çok değerli bağımlılık vardır. Bu bağımlılık "her çalışanın belirli sayıda çocuğu vardır ve bunlar çalışanın ücretinden bağımsızdır" gerçeğine karşı gelmektedir. Bu çok değerli bağımlılık

çalışan çocuğu
biçiminde gösterilir.

Çok Değerli Bağımlılığın Biçimsel Tanımı

- R (A1, A2, A3, …. , An) ilişki şemasındaki nitelikleri 3 ayrık altkümeye (X, Y ve Z) ayıralım: $X \cup Y \cup Z = R$, $X \cap Y \cap Z = \Phi$
- x ve z birer değişmez olmak üzere Yxz ile X niteliklerinin değeri x, Z niteliklerinin değeri ise z olan çoklulardaki Y niteliklerinin değerlerinin kümesini gösterelim.

  $Yxz = \{ y : < x, y, z > \in r \}$
- Tanım: Eğer Yx z yalnız x'e bağımlı ise (z'den bağımsız ise), başka bir deyişle:

  Yx z1 = Yx z2 = …. = Yx zn

  ise, R ilişkisinde X Y çok değerli bağımlılığı vardır denir.
- **Örnek**: R (ders, kitap, öğretmen)

Her ders kullanılan kaynak kitaptan bağımsız olarak, belirli öğretmenler tarafından verilebilmektedir. Örneğin

öğretmen Mat, K1 = öğretmen Mat, K2 = {Ali, Mehmet}

Ders öğretmen'i çok değerli belirlemektedir: ders öğretmen

- **Örnek: R (ders, kitap, öğretmen)**

ders	kitap	öğretmen
Mat	K1	Ali
Mat	K2	Ali
Mat	K1	Mehmet
Mat	K2	Mehmet
Fizik	K3	Ali
Fizik	K4	Ali
Fizik	K1	Ali
Fizik	K3	Kemal
Fizik	K4	Kemal
Fizik	K1	Kemal

Her ders kullanılan kaynak kitaptan bağımsız olarak, belirli öğretmenler tarafından verilebilmektedir. Örneğin
öğretmen Mat, K1 =
öğretmen Mat, K2 = {Ali, Mehmet}
Ders öğretmen'i çok değerli belirlemektedir:
ders öğretmen

- Diğer taraftan her ders için kullanılabilecek belirli kaynak kitaplar vardır ve bunlar dersi veren öğretmenden bağımsızdır. Örneğin:

  kitap Mat, Ali = kitap Mat, Mehmet = {K1, K2}

  kitap Fizik, Ali = kitapFizik, Kemal = {K1, K3, K4}

ders ve kitap nitelikleri arasında çok değerli bağımlılık vardır;
ders kitap'ı çok değerli belirlemektedir.

ders kitap

4.8.2. İlişkiler İçin 4. Normal Biçim (4NF)

4NF İlişki Tanımı

Eğer bir R ilişkisinde, K bir anahtar olmak üzere:

tüm işlevsel bağımlılıklar $K \rightarrow X$

tüm çok değerli bağımlılıklar da K Y

biçiminde ise R ilişkisi 4. Normal Biçimdedir (4NF) denir.

Bölüm5: İlişkisel Cebir

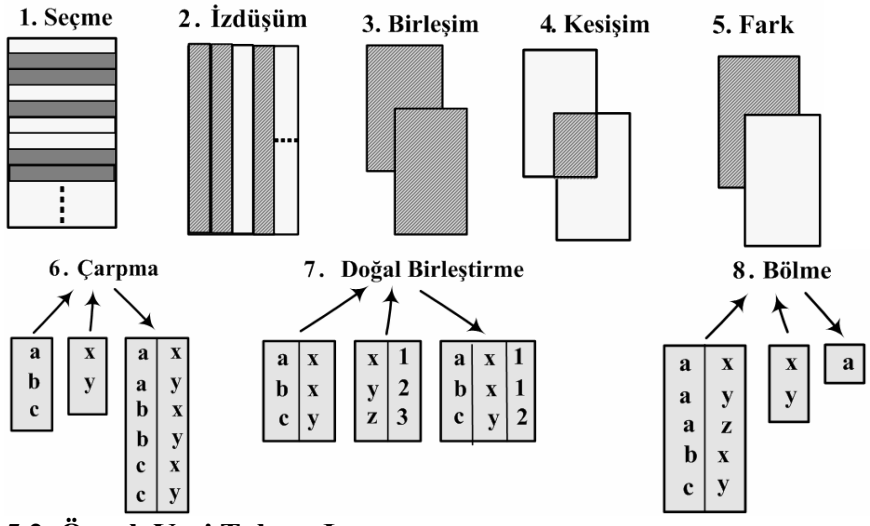
5.1. İlişkisel Diller

➢ İlişkisel Veri Tabanı Yönetim Sistemlerinde kullanılan diller (başka bir deyişle ilişkisel diller) kuramsal temelleri açısından ikiye ayrılır.

- ➢ İlişkisel Cebire (*Relational Algebra*) Dayalı Diller ya da kısaca Cebirsel Diller. Bir sorgunun yanıtını elde etmek için, belirli ilişkisel cebir işlemlerinin belirli sırada uygulanması gerekir. İlişkisel cebirde 8 temel işlem yer almaktadır.
- ➢ İlişkisel Hesaba (*Relational Calculus*) Dayalı Diller. Bu dillerde sorgu, istenilen çokluların/niteliklerin sağlaması gereken bir önerme oluşturularak tanımlanır. Bu dillerin kuramsal temelini niceleme mantığı (*predicate calculus*) oluşturur. İlişkisel hesaba dayalı diller de kendi içinde ikiye ayrılır.

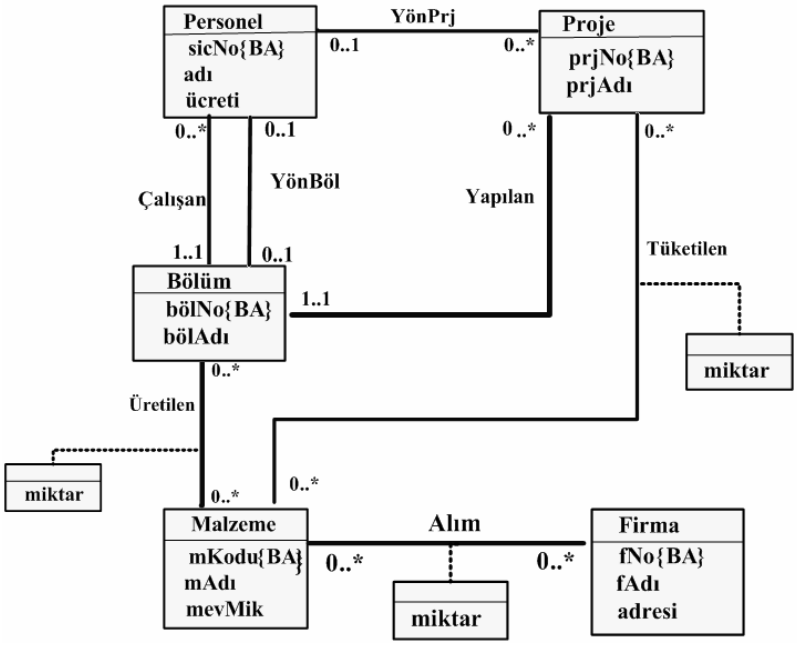
1. Çoklulara Yönelik İlişkisel Hesap (*Tuple Oriented Relational Calculus*) Tabanlı Diller. Bu dillerde işlenen birim çokludur. Önermede çoklu değişkenleri kullanılır.
2. Alanlara Yönelik İlişkisel Hesap (*Domain Oriented Relational Calculus*) Tabanlı Diller. Bu dillerde işlenen birim çokluların bileşenleri olan alanlar, ya da niteliklerdir. Önermede alan değişkenleri kullanılır.

➢ Cebirsel diller ilişkisel hesaba dayalı dillere oranla daha yordamsal dillerdir.



5.2. Örnek Veri Tabanı I

Sanayi Veri Tabanı Varlık-Bağıntı Çizeneği



➢ **Sanayi Veri Tabanının İlişkisel Şeması**

Bölüm (bölNo, bölAdı, yönSicNo)
Peronel (sicNo, adı, ücreti, bölNo)
Proje (prjNo, prjAdı, bölNo, yönSicNo)
Malzeme (mKodu, mAdı, mevMik)
Üretim (bölNo mKodu, miktar)
Tüketim (prjNo, mKodu, miktar)
Firma (fNo, fAdı, adresi)
Alım (fNo, mKodu, miktar)

5.3. İlişkisel İşlemler

➢ İlişkisel cebir, ilişkilere uygulanan bir dizi yüksek düzeyli işlemden oluşur.

➢ İlişkisel cebir işlemleri ya da kısaca ilişkisel işlemler olarak adlandırılan bu işlemler "ilişkisel olarak yetkin (*relationally complete*)" bir küme oluşturur. İlişkisel yetkinlik kavramı, ilişkisel dillerin seçme gücünün bir ölçüsü olup, dilin eksiksiz olduğunun bir göstergesidir. Bir ilişkisel dilin yetkin olabilmesi için, ilişkisel hesap deyimleri ile elde edilebilen her sonucun bu dil ile de elde edilebilmesi gerekir.

➢ İlişkisel cebir işlemlerinden ikisi (seçme ve izdüşüm) tekli, diğerleri ise ikili işlemlerdir.

➢ İkili işlemlerden dördü (birleşim, kesişim, küme fark ve Karteziyen çarpım) alışılmış küme işlemleridir. Diğer ikili işlemler (θ-birleştirme, doğal birleştirme ve bölme) ise özel işlemlerdir.

➢ İlişkisel cebir işlemlerinin tümü birbirinden bağımsız değildir. İşlemlerden birkaçının diğer işlemler cinsinden ifade edilmesi mümkündür.

5.3.1. Seçme İşlemi

➢ Seçme (*selection*) ya da sınırlama (*restriction*) işlemi bir ilişkinin belirli bir koşulu sağlayan çoklularını seçmeyi sağlar. İlişki bir çizelge olarak düşünülürse, seçme işlemi ile çizelgenin belirli satırları seçilir.

➢ Genel yazılış: $\sigma F (R)$
σ: seçme işleci (işlem işareti)
F : seçme koşulu
R : seçme işlemi uygulanacak ilişkinin adı

➢ Örnekler: σücreti > 5000 (Personel)
σ(bölNo = 50) $\wedge$ (miktar > 100) (Üretim)
σ(fNo= 17) $\wedge$ ((mKodu = 'EN567') $\vee$ (miktar > 1000)) (Alım)

5.3.2. İzdüşüm İşlemi

➢ Tekli bir işlem olan izdüşüm (*projection*) işlemi ile bir ilişkinin belirli nitelikleri seçilerek diğerleri atılır (dikey seçme); elde edilen yeni çizelgede, varsa, tekrarlı satırlar da atılır.

➢ Genel yazılış: πnitelik-listesi (R)
π: izdüşüm işleci (işlem işareti)
nitelik-listesi : izdüşüm alınacak niteliklerin listesi (istenilen sırada)
R : izdüşümü alınacak ilişkinin adı

➢ Örnekler: 1. πbölNo, sicNo, adı (Personel)
2. πmKodu (Üretim)
3. πprjNo (σmKodu = 'GD1872' (Tüketim))

5.3.3. Birleşim İşlemi

➢ İkili bir işlem olan birleşim (*union*) işlemi iki ilişkinin küme birleşimini alır. Rasgele

iki ilişkinin birleşimi alınamaz. İki ilişkiye birleşim işleminin uygulanabilmesi için iki ilişkinin dereceleri (nitelik sayıları) aynı olmalı; aynı sıradaki nitelikleri de aynı değer alanı üzerinde tanımlanmış olmalıdır (birleşim-uyar ilişkiler).

- Genel yazılış: R1 ∪ R2
  ∪ : birleşim işleci (işlem işareti)
  R1 ve R2 : birleşimi alınacak ilişkiler

- Örnekler: 1. πyönSicNo (Proje) ∪ πyönSicNo(Bölüm)
  2. πmKodu (σbölNo = 3 (Üretim)) ∪ πmKodu (σfNo = 7 (Alım))

5.3.4. Kesişim İşlemi

- İkili bir işlem olan kesişim (*intersection*) işlemi ile iki ilişkinin küme kesişimi elde edilir. Birleşim işleminde olduğu gibi, iki ilişkiye kesişim işleminin uygulanabilmesi için bu ilişkilerin birleşim-uyar olması gerekir.
- Genel yazılış: R1 ∩ R2
  ∩ : kesişim işleci (işlem işareti)
  R1 ve R2 : kesişimi alınacak ilişkiler

- Örnekler: 1. πyönSicNo (Proje) ∩ πyönSicNo (Bölüm)
  2. πmKodu (σbölNo = 3 (Üretim)) ∩ πmKodu (σfNo = 7 (Alım))

5.3.5. Küme Fark İşlemi

- İkili bir işlem olan küme fark (*set difference*), ya da kısaca fark işlemi ile iki ilişkinin küme farkı elde edilir. Birleşim ve kesişim işlemleri gibi fark işlemi de rasgele iki ilişkiye uygulanamaz. İki ilişkiye fark işlemi uygulanabilmesi için bu ilişkilerin birleşim-uyar olması gerekir.
- Genel yazılış: R1 – R2
  – : küme fark işleci (işlem işareti)
  R1 ve R2 : farkı alınacak ilişkiler
- Örnekler: 1. πyönSicNo(Proje) – πyönSicNo (Bölüm)
  2. πmKodu (σbölNo = 3 (Üretim)) – πmKodu (σfNo = 7 (Alım))
- İlişkisel cebir işlemlerinin tümü birbirinden bağımsız değildir. Örneğin:
  R1 ∩ R2 = R1 (R1 – R2)

5.3.6. Karteziyen Çarpım İşlemi

- İkili bir işlem olan Karteziyen çarpım (*Cartesian product*) ya da kısaca çarpma işlemi ile iki ilişkinin küme çarpımı bulunur.
- Genel yazılış: R1 x R2= { a1a2 : a1∈R1 , a2∈R2}
  x : çarpma işleci (işlem işareti)
  R1 ve R2 : çarpılacak ilişkiler
- Örnekler: 1. Proje x Malzeme
  2. Bölüm x Üretim
- Karteziyen çarpım işlemi ile elde edilecek ilişkiler çok büyük (nitelik sayısı çok olan, çoklu sayısı ise çok büyük olan) ilişkilerdir. Bu ilişkiler genellikle ara ilişkiler olup, üzerlerine birkaç işlem daha uygulandıktan sonra amaç ilişkiler elde edilir.

5.3.7. Birleştirme İşlemi

- İkili bir işlem olan birleştirme (*join*) işlemi ile, iki ilişkinin Karteziyen çarpımının bir altkümesi elde edilir. Birleştirme işleminde işlenen ilişkilerden her birinin belirli bir niteliği (birleştirme niteliği) ile bir aritmetik karşılaştırma işleci (θ) kullanılır. Bu nedenle de işlem genellikle θ-birleştirme (*θ-join*) işlemi olarak anılır.

➢ Genel yazılış: $R1 \underset{a1\ \theta\ a2}{\bowtie} R1 = \sigma_{a1\ \theta\ a2}(R1 \times R2)$

R1 ve R2 : birleştirilen ilişkiler

$\underset{a1\ \theta\ a2}{\bowtie}$: birleştirme işleci (işlem işareti)

a1 ve a2 : sırasıyla R1 ve R2'nin birer niteliği

θ : bir aritmetik karşılaştırma işleci

➢ **Örnekler:**

1. Üretim $\underset{bölNo = bölNo}{\bowtie}$ Bölüm

2. $\pi_{bölNo,\ miktar}(\sigma_{mKodu= 'K32'}(Üretim)) \bowtie \pi_{miktar}(\sigma_{(mKodu = 'K32') \wedge (prjNo = 7)}(Tüketim))$

miktar > miktar

5.3.8. Doğal Birleştirme İşlemi

➢ Doğal birleştirme işlemi ile, iki ilişki aynı adı taşıyan niteliklerinin eşitliği üzerinden birleştirilir ve sonuç ilişkide tekrarlı niteliklerden birer tanesi atılır.

➢ Genel yazılış: $R1 \bowtie R2$

R1 ve R2 : birleştirilen ilişkiler

$\bowtie$: birleştirme işleci (işlem işareti)

➢ **Örnekler:**

1. Alım $\bowtie$ Firma

2. $\pi_{sicNo,\ ücreti,\ bölNo}$ (Personel) $\bowtie$ Bölüm

3. $\pi_{bölNo,\ mKodu}$ (Üretim) $\bowtie$ $\pi_{prjNo,\ mKodu}$ (Tüketim)

5.3.9. Bölme İşlemi

➢ İkili bir işlem olan bölme (*division/quotient*) işlemi, ilişkisel işlemler arasında en karmaşık olanıdır. Doğal birleştirme işleminde olduğu gibi, iki ilişkiye bölme işleminin uygulanabilmesi için, ilişkilerde en az bir ortak (aynı adlı) nitelik bulunması gerekir. Bölme işlemi bu ortak nitelik ya da nitelikler üzerinden uygulanır.

➢ Genel yazılış: $R1 \div R2$

R1 : bölünen ilişki

R2 : bölen ilişki

÷ : bölme işleci (işlem işareti)

➢ İlişkisel bölme işleminin nasıl çalıştığını anlatabilmek için bölünen ve bölen ilişkilerin ikişer nitelikli olduğunu düşünelim: $R_1(A, B)$ ve $R_2(B, C)$

➢ $R_1(A, B) \div R_2(B, C) = \pi_A(R_1) - \pi_A((\pi_A(R_1) \times \pi_B(R_2)) - R_1)$

➢ Bölme işleminin, daha anlaşılır sözel bir tanımı da şöyle yapılabilir. Bölme işlemi ile elde edilecek ilişki tek nitelikli (A) bir ilişkidir. Sonuç ilişkide bir $< a >$ çoklusunun yer alabilmesi için, R_2 ilişkisindeki her farklı bi değeri için, R_1 ilişkisinde $< a, bi >$ çoklusunun yer alması gerekir.

➢ Örnekler: 1. $(\pi_{prjNo,\ mKodu}(Tüketim)) \div$ Alım

2. $(\pi_{bölNo,\ MKodu}(Üretim)) \div$ Malzeme

5.4. Aktarma İşlemi ve Ara İlişkilerin Adlandırılması

➢ İlişkisel cebir işlemleri çok alışık olduğumuz işlemler değildir. İlişkisel işlemler cinsinden düşünmek ve birçok ilişkisel işlem içeren uzun bir ilişkisel cebir deyimini doğru olarak

yazmak ya da yazılmış bir deyimi anlamak kolay değildir. Bu nedenle birçok işlem içeren uzun deyimlerin parçalanarak yazılabilmesinde büyük yarar vardır.

➢ Bir ilişkisel işlemin uygulanması ile elde edilecek ilişkinin niteliklerinin adları, işlenen ilişki ya da ilişkilerdeki nitelik adları ile aynı olur. Ancak bu şekilde elde edilen bir ilişkide aynı adlı birden çok nitelik bulunursa, bunları birbirinden ayırdetmek için ilişki adları ile niteleme yapılabilir. Ancak bazı nitelikleri niteleme yöntemiyle de ayırdetmek mümkün olmayabilir.

➢ Aktarma (ya da ara ilişkileri adlandırma) işlemi kullanılarak hem uzun deyimler küçük parçalar halinde yazılarak düşünme ve anlama kolaylaştırılacak hem de ara ilişkilerin ve niteliklerinin adlandırılması ile aynı adı taşıyan niteliklerin ayırdedilmesi sağlanacaktır.

➢ Genel yazılış: P [(yeni-nitelik-adları)] ← ilişkisel-cebir-deyimi
P : ara ilişkinin adı
yeni-nitelik-adları : ara ilişkinin niteliklerine verilen yeni adlar
← : aktarma işleci (işlem işareti)

➢ Örnekler: 1. $\pi_{prjNo,\ prjAdı}$ ($\sigma_{mKodu= 'K32'}$ (Proje ⋈ Tüketim))

Aktarma işlemini kullanarak, yukarıdaki deyim yerine aşağıdaki deyimler yazılabilir.

X1 ← Proje ⋈ Tüketim

X2 ← $\sigma_{mKodu = 'K32'}$ (X1)

X3 ← $\pi_{prjNo,\ prjAdı}$ (X2)

2. En az iki farklı malzeme üreten bölümlerin numaralarını bulmak için:

X1 ← Üretim

X2 ← $\sigma_{(Üretim.mKodu \neq X1.mKodu) \wedge (Üretim.bölNo = X1.bölNo)}$ (Üretim x X1)

X3 ← $\pi_{Üretim.bölNo}$ (X2)

➢ 3. En az iki projede kullanılan malzemeleri bulmak için:

X1 (prjNo1, mKodu) ← $\pi_{prjNo,\ mKodu}$ (Tüketim)

X2 ← Tüketim ⋈ X1

X3 ← π_{mKodu} ($\sigma_{prjNo \neq prjNo1}$ (X2))

5.5. İlişkisel İşlemlerle Sorgu Örnekleri

➢ SÖ.5.1. "Çalıştığı bölüm dışındaki bir bölümün yöneticisi olan kişilerin sicil numarasını ve adını bul".

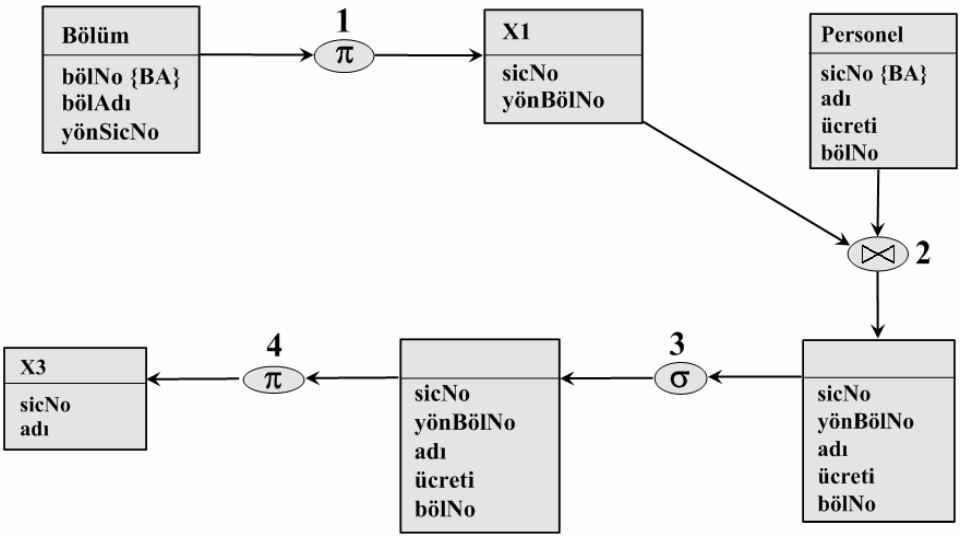
Bu sorgunun ilişkisel işlemlerle anlatımı için örnek bir çözüm:

X1(sicNo,yönBölNo) ← $\pi_{yönSicNo,\ bölNo}$ (Bölüm)

X2 ← $\sigma_{yönBölNo \neq bölNo}$ (X1 ⋈ Personel)

X3 ← $\pi_{sicNo,\ adı}$ (X2)

➢ **SÖ.5.1 Numaralı Sorgu İçin İlişkisel İşlemler Çizeneği**



➢ **SÖ.5.2.** "Tükettiği malzemelerin tümü yapıldığı bölümde üretilen malzemeler olan (yapıldığı bölümde üretilmeyen hiçbir malzemeyi tüketmeyen) projelerin numaralarını ve adlarını bul"

$X1 \leftarrow \pi_{prjNo,\ mKodu}$ (Tüketim)

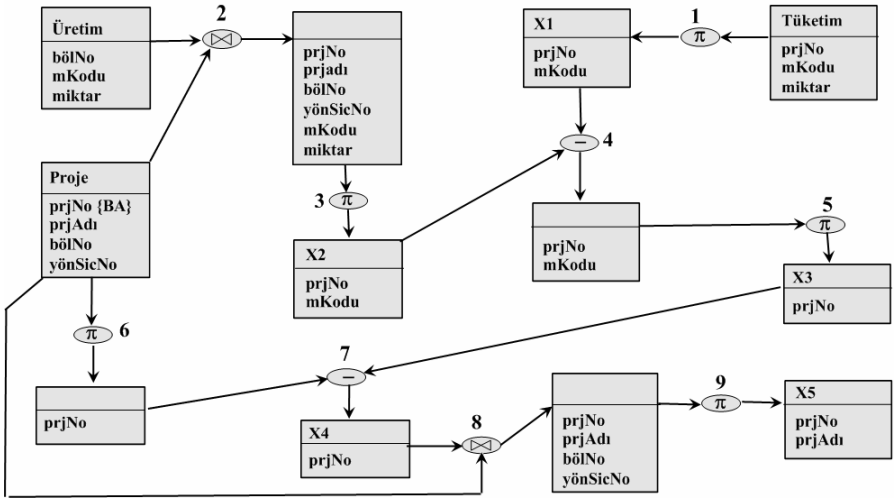
$X2 \leftarrow \pi_{prjNo,\ mKodu}$ (Proje $\bowtie$ Üretim)

$X3 \leftarrow \pi_{prjNo}$ (X1 – X2)

$X4 \leftarrow \pi_{prjNo}$ (Proje) – X3

$X5 \leftarrow \pi_{prjNo,\ prjAdı}$ (Proje $\bowtie$ X4)

➢ **SÖ.5.2 Numaralı Sorgu İçin İlişkisel İşlemler Çizeneği**



➢ **SÖ.5.3.** "Çalıştığı bölümün yöneticisinden daha yüksek ücret alan personelin sicil numarasını ve adını bul".

$X1 \leftarrow \pi_{bölNo,\ yönSicNo}$ (Bölüm)

$X2 \leftarrow$ Personel $\bowtie$ X1

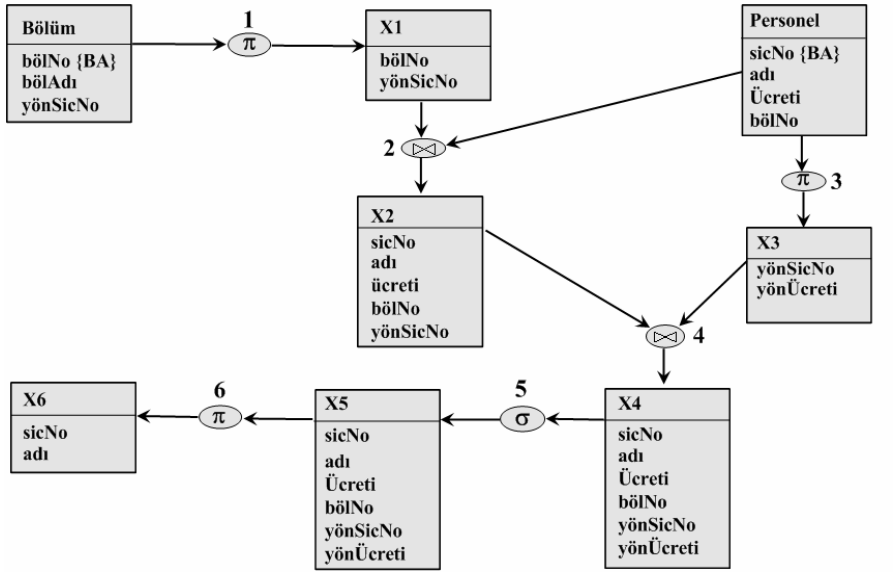
$X3$ (yönSicNo,yönÜcreti) $\leftarrow \pi_{sicNo,\ ücreti}$ (Personel)

$X4 \leftarrow X2 \bowtie X3$

$X5 \leftarrow \sigma_{ücreti\ >\ yönÜcreti}$ (X4)

$X6 \leftarrow \pi_{sicNo,\ adı}$ (X5)

➢ **SÖ.5.3 Numaralı Sorgu İçin İlişkisel İşlemler Çizeneği**



➢ **SÖ.5.4.** "6 numaralı projede tüketilen tüm malzemeleri tüketen (tükettiği malzemeler arasında, 6 numaralı projede tüketilen malzemelerin tümü bulunan) projelerin no ve adlarını bul".

$X1 \leftarrow \pi_{prjNo,\ mKodu}(Tüketim) \div \pi_{mKodu}(\sigma_{prjNo = 6}(Tüketim))$

$X2 \leftarrow \pi_{prjNo,\ prjAdı}(Proje \bowtie X1)$

➢ **SÖ.5.4 Numaralı Sorgu İçin İlişkisel İşlemler Çizeneği**