

İKİNCİ BÖLÜM

2

2. İLİŞKİSEL VERİTABANI TASARIMI ve NORMALİZASYON

- 2.1. Veritabanı Tasarımı**
- 2.2. Kavramsal Tasarım**
- 2.3. Veritabanı Normalizasyon Kuralları**
- 2.4. Örnek Veritabanı Tasarımı**
- 2.5. Bölüm Özeti**
- 2.6. Çalışma Soruları**

2. İLİŞKİSEL VERİTABANI TASARIMI VE NORMALİZASYON

Veritabanı tasarımını üçayaklı bir sehpası gibi düşünürsek ayaklardan birisi olmaza sehpası ayakta durmaz veya yarınlı olursa sehpası sallanıp düşebilir. Bu üçayağımız yetenek, bilgi ve tecrübe dir. Öncelikle veritabanı sistemi ilişkileri ve normalizasyon kuralları bilinmelidir. Bu kurallar veritabanının hangi şartlara göre tasarlaması gerektiğini belirler ve veri tekrarını, veri kaybını veya veri yetersizliğini önler. Bu kurallar uygulanmak zorunda değildir ama depolanan veri miktarı artıkça bunlara ihtiyaç duyulmaktadır.

2.1. VERİTABANI TASARIMI

Projenin tasarım aşamasında veritabanı tasarımı çok iyi yapılmalıdır. Daha sonra yapılacak değişiklikler sorunlar çıkartabilir veya çok zahmetli olabilir.

Aşağıda veritabanı tasarımı yapılmırken izlenecek adımlar verilmiştir.

- **Depolanacak Veriler:** Veritabanı içerisinde tutmak istediğiniz bilgiler belirlenerek, bunlar gruplandırılmalıdır. Aşağıda örnek veritabanı ve tutulacak veri grupları verilmiştir. Bu gruplar tablolari, içerisindeki bilgiler ise sütunları oluşturmak için kullanılacak.
- **Okul Veritabanı:** Öğrenciler(no, adı, tcno, bölümü), Kurum Bilgileri(adi, mudur, adres, telefon), Notlar(öğrenci no, dersi, not1,not2), Dersler(ders kodu, ders adı), Bölümler(bölüm kodu, bölüm adı)
- **Tabloların Oluşturulması:** Belirlenen veri grupları ve sütunlar doğrultusunda tablolar oluşturulur. Tablo ve sütun isimlerinde Türkçe karakter, fonksiyon ismi kullanılmamalı ve isimler hatırlanacak şekilde belirlenmelidir. Sütunlar, tutulacak veri türüne göre tanımlanmalıdır.
- **Anahtar Sütunların Belirlenmesi:** Anahtar sütunların özelliklerinden daha önce bahsetmiştik. Bu bilgiler doğrultusunda kayıtların birbirinden ayırt edilebilmesi için anahtar sütun oluşturulur. Anahtar sütunun tanımlanma zorunluluğu yoktur ama verilere daha çabuk ulaşmak ve tekrar eden kayıtların önlenmesi için kullanılabilir.
- **Tabloları Bölme:** Tabloda tekrar eden kayıtlarla karşılaşılacaksa tekrar eden sütun için yeni tablo oluşturulur. Örneğin, öğrenci notlarının olduğu bir tabloda sütun olarak sadece öğrenci no, ders ve not sütunu var ise öğrencinin aynı dersten girmiş olduğu sınav sayısına kadar yeni kayıt oluşur. Ama notlar tablosunu tasarlarken öğrenci no, ders, not1 ve not2 şeklinde tasarlantıp dersler de ayrı bir tabloda tutulursa veritabanı daha sağlıklı sonuçlar verecektir ve gereksiz yere tablonun boyutu artmayacaktır.

- **İlişkilerin Kurulması:** Projelerin büyük kısmında tablolar arasında kesinlikle ilişki oluşturulmaktadır. Oluşturulan tablolar arasında ilişkiler sorgu yardımıyla oluşturulacaksa bu aşamada bir işleme gerek yoktur. MS Access gibi bir veritabanı sisteminde şematik olarak oluşturulacaksa tablolardaki sütunlar belirli kurallar çerçevesinde ilişkilendirilir. Örneğin, öğrenci bilgilerinin ve notların tutulduğu tablolarda, notlar tablosunda kesinlikle öğrenci no kullanılmıştır. Öğrenci tablosundaki öğrenci no ile notlar tablosundaki öğrenci no sütunu ilişkilendirilmelidir.

2.2. İLİŞKİSEL VERİTABANININ KAVRAMSAL TASARIMI

Kavramsal tasarım, veritabanında tutulacak verilerin daha üst seviyede gösterilmesi için kullanılır. Kavramsal tasarım için en çok kullanılan model ER(Entity Relationship-Varlık İlişki) modelidir.

Varlık-ilişki modeli bir tür veri modeli olmasına rağmen bugüne kadar hiçbir VTYS'de kullanılmamıştır. Buna karşılık varlık-ilişki modeli kavramsal tasarım için kullanılan en popüler modeldir. Bu model kullanılarak, VTYS'den bağımsız modelleme yapılarak ve ilişkiler tanımlanarak herhangi bir VTYS veritabanına dönüştürülebilir.

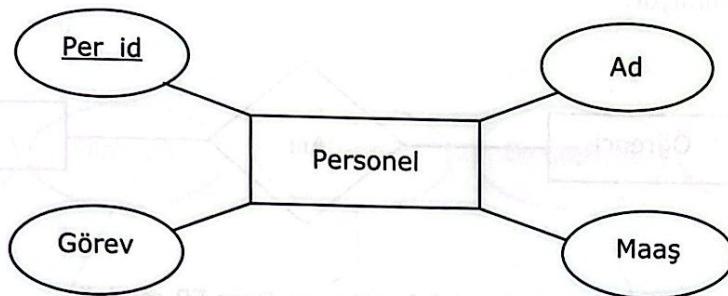
Varlık-ilişki modelinde kullanılan şekiller veritabanlarının şematik olarak tasarılanması için kullanılır.

Varlık-ilişki modelinde temel üç öğe vardır. Bunlar;

- **Varlık:** Modelin en temel öğesidir. Var olan ve benzerlerinden ayırt edilebilen her şey varlıktır. Örneğin, kitap, öğrenci, araba birer varlıktır. Birden fazla varlığın oluşturduğu kümeye **varlık kümesi** denilir. Model içerisinde varlık kümesi dikdörtgen ile gösterilir ve içerisine varlığın ismi yazılır. Veritabanı olarak düşünülürse her bir tablo bir varlık kümesidir.
- **Nitelik:** Varlıkların her bir özelliği bir nitelik olarak ifade edilir. Örneğin, öğrencinin numarası ve bölümü öğrenci varlığının nitelikleridir. Model içerisinde nitelikler oval ile gösterilir ve içerisine niteliğin ismi yazılır. Nitelik bağlı olduğu varlığa düz bir çizgi ile birleştirilir. Veritabanı olarak düşünülürse tablonun her bir sütunu bir varlığı gösterir. Bir niteliğin değeri her bir varlık için farklısa bu nitelik **anahtar nitelik** olarak belirlenir. Anahtar nitelik şema içerisinde niteliğin altı çizilerek gösterilir. Örneğin, her bir öğrenci varlığı farklı öğrenci numarası niteliklerine sahip olacağı için öğrenci no niteliği anahtar nitelik olarak belirlenebilir.

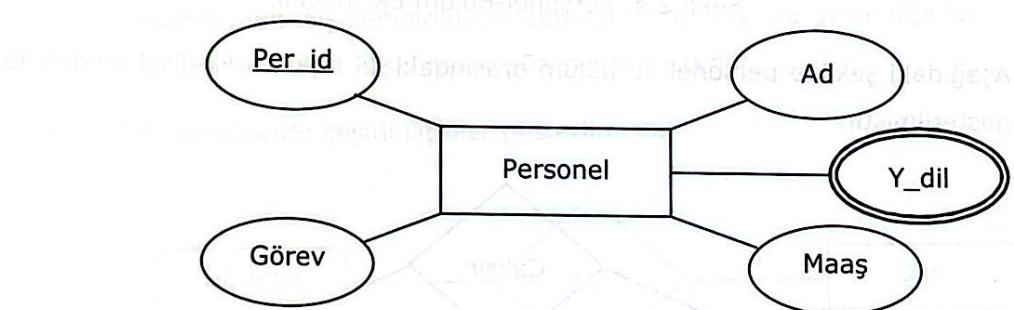
Bölüm 2: İlişkisel Veritabanı Tasarımı ve Normalizasyon

Aşağıdaki şekilde personel varlık kümeleri nitelikleri verilmiştir.



Şekil 2.1. Personel varlık kümeleri.

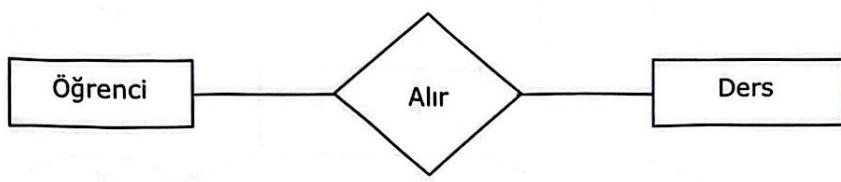
Bir varlık tek bir değere sahipse tek değerli nitelik olarak isimlendirilir ve yukarıda gösterildiği gibi kullanılır. Örneğin, bir öğrencinin numarası bir tanedir bu nedenle öğrenci numarası niteliği tek değerli bir niteliktir. Bazı durumlarda bir nitelik birden fazla değer içerebilir. Örneğin, bir personel birden fazla yabancı dil bileyebilir. Bu durumda personel varlık kümelerinin yabancı dil niteliği **çok değerli niteliktir**. Çok değerli nitelikler çift çizgili oval ile gösterilir.



Şekil 2.2. Personel varlık kümeleri ve çok değerli nitelik.

- **Domain(Etki Alanı):** Niteliklerin alabileceği değer aralığıdır. Örneğin, öğrenci notlarını içeren sınav niteliği için alacağı değerleri 0 ile 100 arasında belirlemek etki alanı oluşturmaktadır. Etki alanı ER şeması içerisinde gösterilmez.
- **İlişki:** Farklı varlıklar arasındaki ilişkileri ifade eder. Örneğin, öğrenci ve dersler ayrı varlık kümeleridir ama öğrenciler ders almak zorunda olduğu için iki varlık arasında ders alma ilişkisi vardır. Model içerisinde ilişkiler baklava dilimi ile gösterilir ve içerisine ilişkisinin ismi yazılır. Baklava dilimi ilişkili olduğu varlıklarla düz çizgi ile bağlanır. Tablolar arasında kurulan ilişkiler(1-n, 1-1, n-m) model içerisinde ilişki olarak geçmektedir. İki varlık kümeleri arasında birden fazla ilişki bulunabilir.

Aşağıdaki şekilde öğrenci ve ders arasındaki ilişki varlık-iliski modeli ile gösterilmiştir.



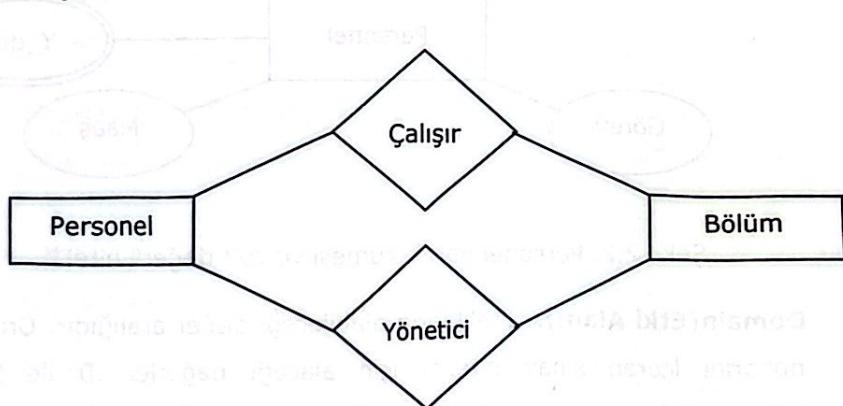
Şekil 2.3. Öğrenci-Ders ER modeli.

Aşağıdaki şekilde personel ve bölüm arasındaki ilişki varlık-iliski modeli ile gösterilmiştir.



Şekil 2.4. Personel-Bölüm ER modeli.

Aşağıdaki şekilde personel ve bölüm arasındaki iki ilişki varlık-iliski modeli ile gösterilmiştir.

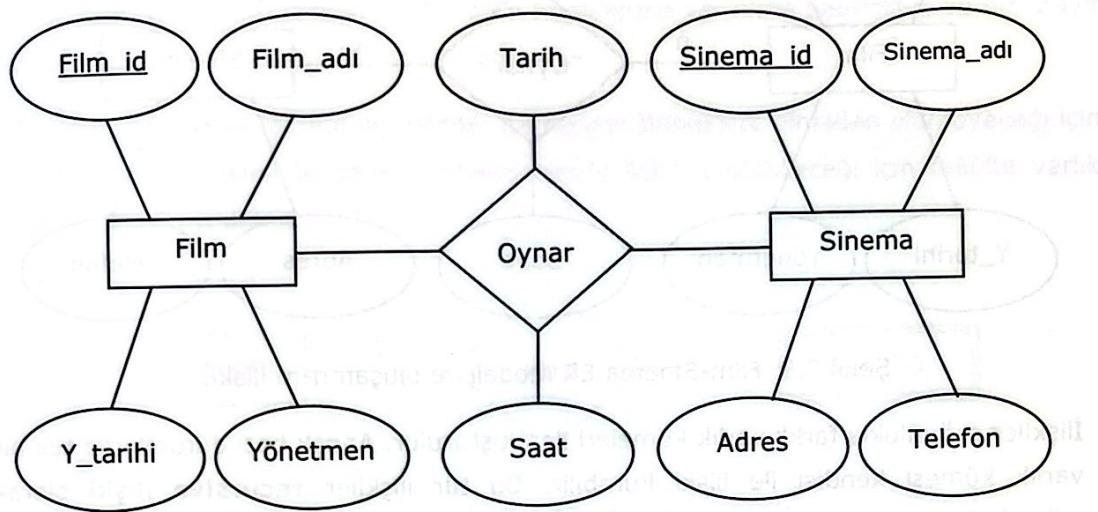


Şekil 2.5. Personel-Bölüm ER modeli.

Varlık kümeleri arasında oluşturulan ilişkilerde, ilişki sonucu nitelikler oluşabilir. Bu tür nitelikler **tanımlayıcı nitelik** olarak isimlendirilir ve nitelikte olduğu gibi oval ile gösterilip ilişkiye düz bir çizgi ile bağlanır. Oluşturulan bir ilişki yoksa tanımlayıcı nitelik hiçbir zaman oluşmayacaktır.

Bölüm 2: İlişkisel Veritabanı Tasarımı ve Normalizasyon

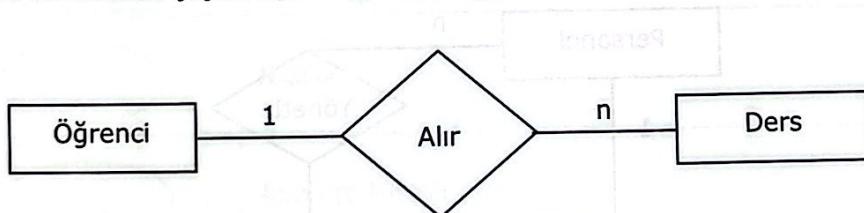
Örneğin, gösterime giren bir film çeşitli sinemalarda oynatılacaktır. Her sinemada gösterim tarih ve saatı farklı olacağı için oluşturulan ilişki saat ve tarih gibi tanımlayıcı niteliklere sahip olmalıdır.



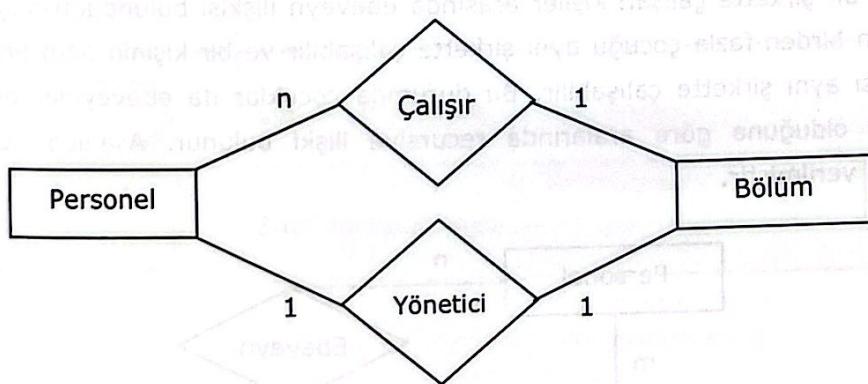
Şekil 2.6. Filem-Sinema ER modeli.

Varlık kümelerinin ilişkilendirilmesi sonucu 1-n, 1-1 ve n-m ilişkiler oluşacaktır. Oluşan ilişkiler de varlık-ilişki modeli üzerinde gösterilir.

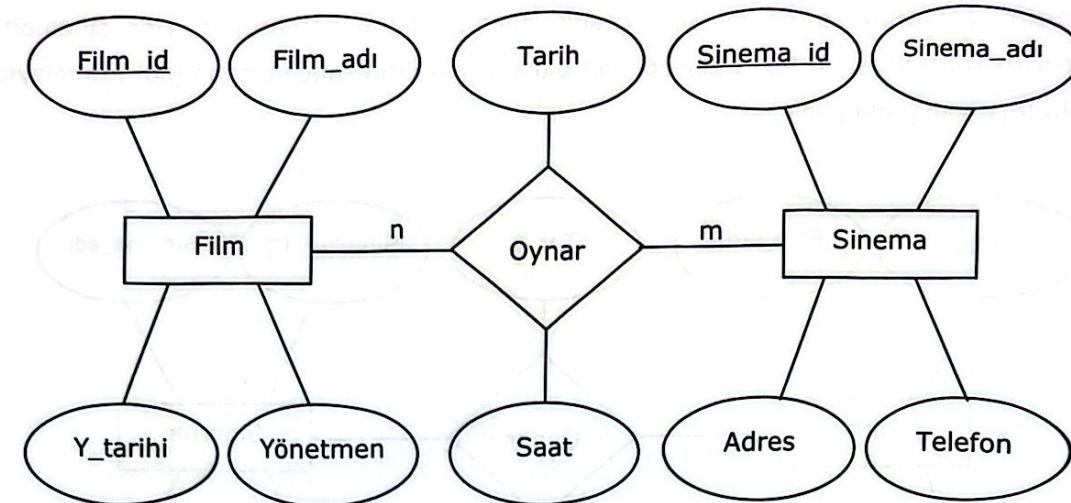
Aşağıdaki örneklerde çeşitli ilişkiler gösterilmiştir.



Şekil 2.7. Öğrenci-Ders ER modeli ve 1-n ilişki.



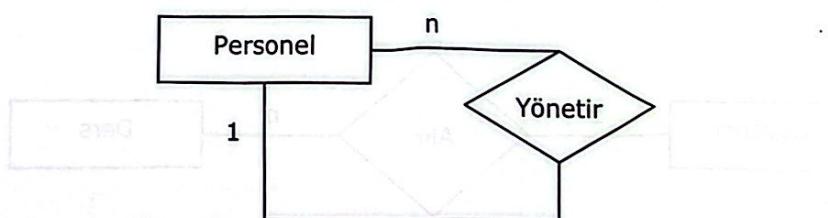
Şekil 2.8. Personel-Bölüm ER modeli ve oluşan ilişkiler.



Şekil 2.9. Film-Sinema ER modeli ve oluşan n-m ilişki.

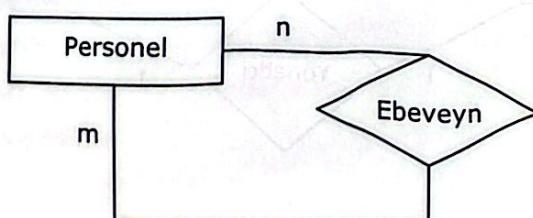
İlişkiler çoğunlukla farklı varlık kümeleri ile oluşturulur. Ancak bazı durumlarda tek bir varlık kümesi kendisi ile ilişki kurabilir. Bu tür ilişkiler **recursive ilişki** olarak adlandırılır.

Örneğin, bir kişi hem personel hem de yönetici olabilir. Yani birden fazla kişi bir kişinin yönetiminde olabilir ve bir kişi birden fazla kişiyi yönetir. Bu tür bir ilişkiyi personel farlılığının kendisi ile ilişkilendirilmesi sonucu elde edilir. Aşağıda bu ilişkinin gösterimi verilmiştir.



Şekil 2.10. Recursive ilişki örneği.

Örneğin, bir şirkette çalışan kişiler arasında ebeveyn ilişki bulduğu düşünülürse, bir kişinin birden fazla çocuğu aynı şirkette çalışabilir ve bir kişinin hem annesi hem de babası aynı şirkette çalışabilir. Bu durumda çocuklar da ebeveynler de şirketin personeli olduğuna göre aralarında recursive ilişki bulunur. Aşağıda bu ilişkinin gösterimi verilmiştir.

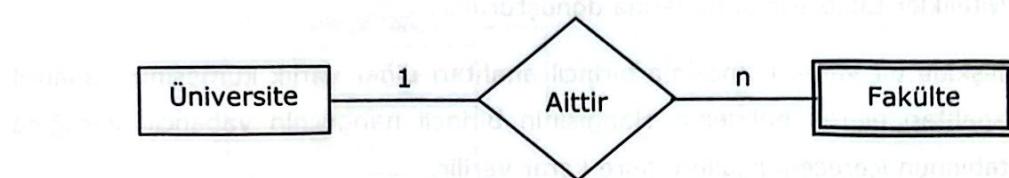


Şekil 2.11. Recursive ilişki örneği.

Zayıf Varlık Kümeleri

Bir varlık kümesi anahtar niteliğe sahip değilse **zayıf varlık kümesi** olarak adlandırılır. Zayıf varlık kümeleri güçlü varlık kümeleri ile ilişkilendirilerek kullanılırlar. Yani zayıf varlık kümelerinin güdü varlık kümelerine var olma bağımlılığı vardır. Zayıf varlık kümeleri çift çizgili dörtgen ile gösterilir.

Örneğin, üniversite fakülte ilişkisinde, bir fakülte üniversite olmadan olamayacağı için ve aynı fakülte isminde başka üniversitelerde fakülte olabileceği için fakülte varlık kümesi zayıf varlık kümesidir.



Şekil 2.12. Zayıf varlık kümesi.

Aşağıdaki tabloda varlık-iliski modelinde kullanılan semboller verilmiştir.

Sembol	Açıklama
[Empty rectangle]	Varlık Kümesi
[Empty oval]	Nitelik
[Oval with a horizontal line]	Anahtar Nitelik
[Diamond shape]	İlişki
[Empty oval with a line]	Çok değerli nitelik
[Empty rectangle with a line]	Zayıf Varlık Kümesi

2.2.1. Varlık-İlişki Modelinin Tablolara Dönüşürtlmesi

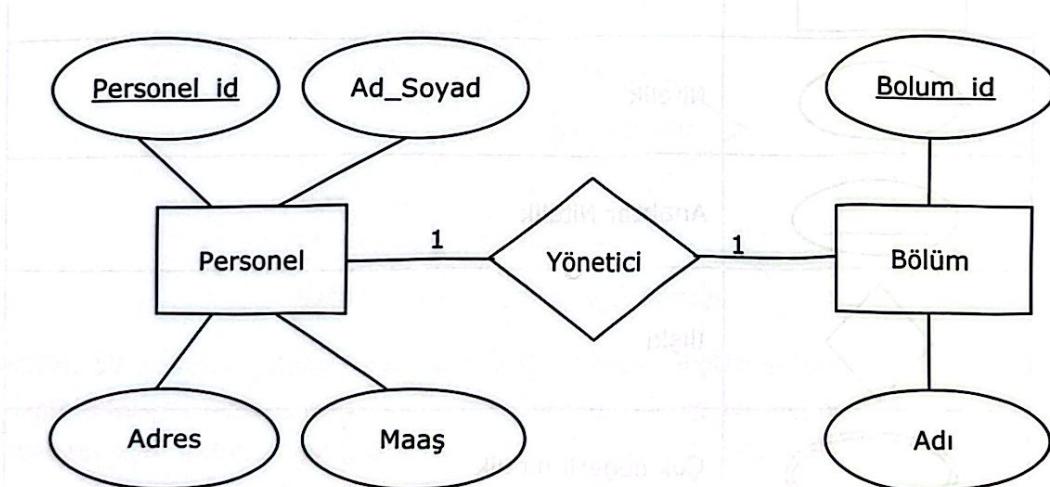
Oluşturulan model tabloya dönüştürülürken varlık kümeleri tabloyu, varlık kümelerinin nitelikleri de tablonun sütunlarını oluşturmak için kullanılacaktır.

Tabloya dönüştürme işleminde modelde oluşturulan ilişkilerin durumuna göre tabolların ilişkileri ve doğal olarak da anahtar sütunları belirlenir.

a. Bire-Bir(1-1) İlişkilerin Tabloya Dönüşürtlmesi

- Varlık kümeleri tablolara dönüştürülür.
- Nitelikler tabolların sütunlarına dönüştürülür.
- İlişkide bir varlık kümelerinin birincil anahtarı diğer varlık kümelerinin yabancı olanağı olarak belirlenir. Hangisinin birincil hangisinin yabancı olacağına tablonun içereceği bilgilere göre karar verilir.
- Model içerisinde oluşturulan ilişkilerde tanımlayıcı nitelik bulunuyorsa, tanımlayıcı nitelikler yabancı anahtarlar olarak kullanılan tabloya sütun olarak eklenir.

Örnek: Bu kurallar doğrultusunda aşağıda verilen varlık-ilişki modelini tabloya dönüştürelim.



Personel varlık kümelerini *personel* ve bölüm varlık kümelerini de *bolum* tablosu olarak isimlendirelim.

Personel tablosunun sütunları *personel_id*, *ad_soyad*, *adres* ve *maas*, bölüm tablosunun sütunları *bolum_id* ve *adi* olacaktır.

Bölüm 2: İlişkisel Veritabanı Tasarımı ve Normalizasyon

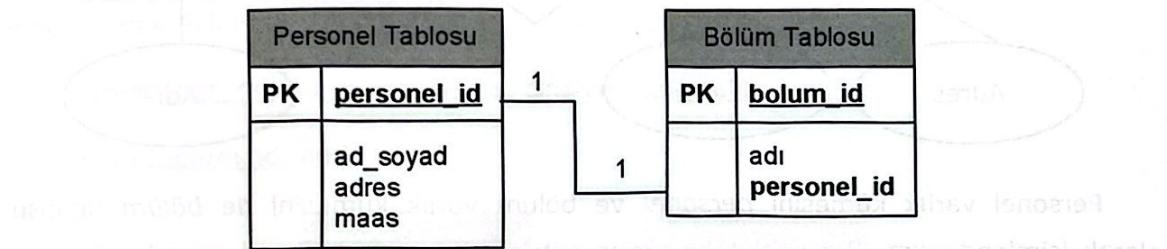
İlişki 1-1 olduğu için iki farklı yolla ilişkiyi ifade edebiliriz.

I.Yol

Personel tablosunun birincil anahtarı bölüm tablosunda yabancı anahtar olarak kullanılır. Tablolar aşağıdaki gibi elde edilir.

Personel(personel_id, ad_soyad, adres, maas)

Bölüm(bolum_id, adı, personel_id)

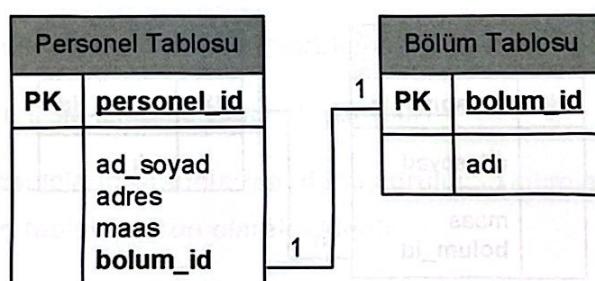


II.Yol

Bölüm tablosunun birincil anahtarı personel tablosunda yabancı anahtar olarak kullanılır. Tablolar aşağıdaki gibi elde edilir.

Personel(personel_id, ad_soyad, adres, maas, bolum_id)

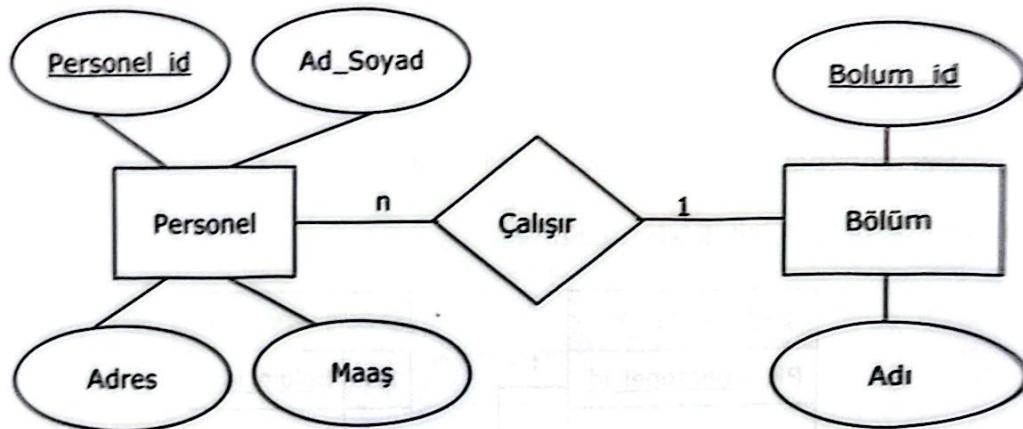
Bölüm(bolum_id, adı)



b. Bire-Çok veya Çoğa-Bir(1-n, n-1) İlişkilerin Tabloya Dönüşürtlmesi

- Varlık kümeleri tablolara dönüştürülür.
- Nitelikler tabloların sütunlarına dönüştürülür.
- İlişkinin n tarafındaki tabloya 1 tarafındaki tablonun birincil anahtar sütunu yabancı anahtar olarak eklenir.
- Model içerisinde oluşturulan ilişkilerde tanımlayıcı nitelik bulunuyorsa, tanımlayıcı nitelikler ilişkinin bulunduğu n tarafındaki tabloya sütun olarak eklenir.

Örnek: Bu kurallar doğrultusunda aşağıda verilen varlık-ilişki modelini tabloya dönüştürelim.



Personel varlık kümelerini *personel* ve bölüm varlık kümelerini de *bolum* tablosu olarak isimlendirelim. Personel tablosunun sütunları *personel_id*, *ad_soyad*, *adres* ve *maas*, bölüm tablosunun sütunları *bolum_id* ve *adi* olacaktır.

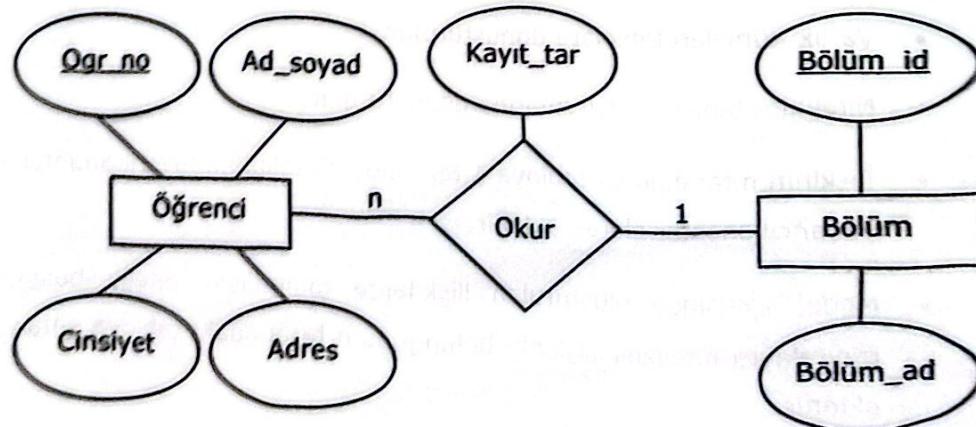
İlişkinin n olduğu taraf personel tablosu olduğu için bölüm tablosundaki birincil anahtar olan *bolum_id* sütununu personel tablosuna yabancı anahtar olarak eklenmesi gereklidir.

Personel(personel_id, ad_soyad, adres, maas, bolum_id)

Bölüm(bolum_id, adi)

Personel Tablosu		Bölüm Tablosu	
PK	personel_id	PK	bolum_id
	ad_soyad		
	adres		
	maas		
	bolum_id		adi
n		1	

Örnek: Aşağıda verilen varlık-ilişki modelini tabloya dönüştürelim.



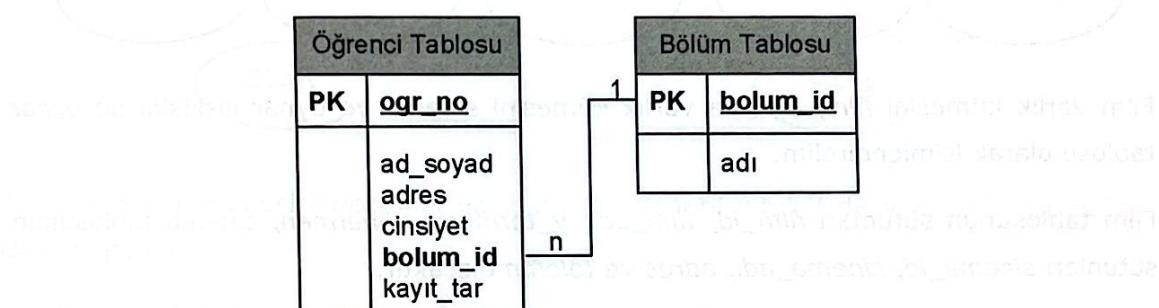
Öğrenci varlık kümesini *öğrenci* ve bölüm varlık kümesini de *bölüm* tablosu olarak isimlendirelim.

Öğrenci tablosunun sütunları *ogr_no*, *ad_soyad*, *adres* ve *cinsiyet*, bölüm tablosunun sütunları *bolum_id* ve *adı* olacaktır.

İlişkinin n olduğu taraf öğrenci tablosu olduğu için bolum tablosundaki birincil anahtar olan *bolum_id* sütununun öğrenci tablosuna yabancı anahtar olarak eklenmesi gerekir. Ayrıca tanımlayıcı nitelik olan *kayıt_tar* bilgisi de ilişkinin n olduğu öğrenci tablosuna sütun olarak eklenir.

Öğrenci(ogr_no, ad_soyad, adres, cinsiyet, bolum_id, kayıt_tar)

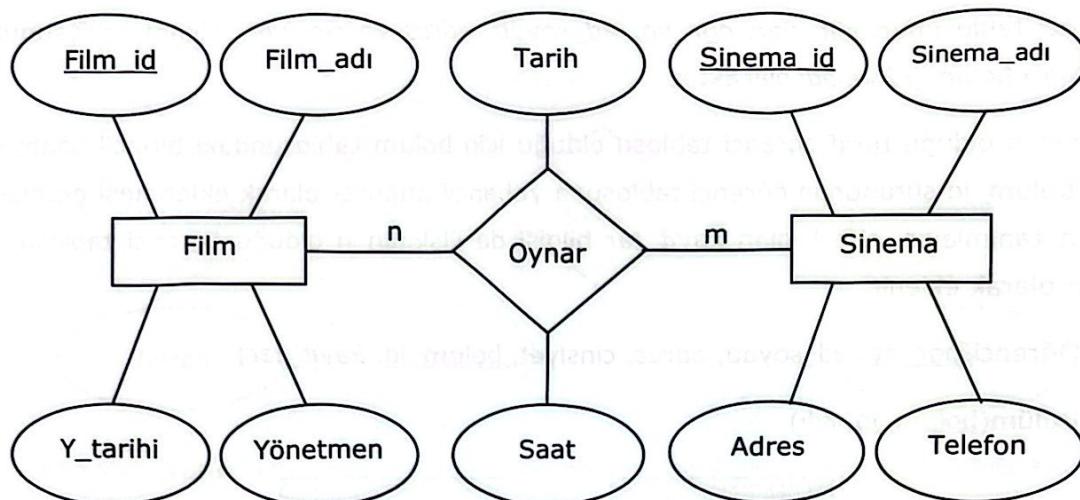
Bölüm(bolum_id, adı)



c. Çoğa-Çok(n-m) İlişkilerin Tabloya Dönüşürtülmesi

- Varlık kümeleri tablolara dönüştürülür.
- Oluşturulan ilişki isminde tablo oluşturulur.
- Nitelikler taboların sütunlarına dönüştürülür. Tanımlayıcı nitelikler ilişkiden oluşturulan tabloya sütun olarak eklenir.
- İlişkiyi oluşturan taboların birincil anahtarları ilişkiyi oluşturan taboya yabancı anahtar olarak eklenir.
- İlişkiden oluşturulan tablonun birincil anahtarı oluşturulan yabancı anahtarların birleşiminden oluşur. Eğer, bu şekilde oluşturulan birincil anahtar ihtiyaçlara cevap vermiyorsa yeni bir sütun eklenerek birincil anahtar yapılır.

Örnek: Bu kurallar doğrultusunda aşağıda verilen varlık-iliski modelini tabloya dönüştürelim.



Film varlık kümesini *film*, sinema varlık kümesini *sinema* ve oynar ilişkisini de *oynar* tablosu olarak isimlendirelim.

Film tablosunun sütunları *film_id*, *film_adi*, *y_tarihi* ve *yönetmen*, sinema tablosunun sütunları *sinema_id*, *sinema_adı*, *adres* ve *telefon* olacaktır.

Oynar tablosuna ise *tarih* ve *saat* sütunlarıyla birlikte film tablosunun birincil anahtar sütunu *film_id* ve sinema tablosunun birincil anahtar sütunu *sinema_id* yabancı anahtar sütun olarak eklenir. Oynar tablosunun birincil anahtar sütunu *film_id* ve *sinema_id* sütunlarının birleşimi olabilir ama bu örnekte *oynar_id* isminde yeni bir sütun eklenerek birincil anahtar olarak belirlenmiştir.

Film(film_id, film_adi, y_tarihi, yönetmen)
Sinema(sinema_id, sinema_adı, adres, telefon)
Oynar(oynar_id, film_id, sinema_id, tarih, saat)

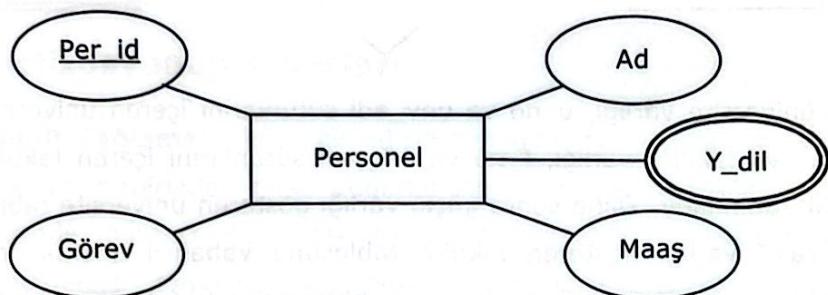


d. Çok Değerli Niteliklerin Tabloya Dönüştürülmesi

- Varlık kümeleri tablolara dönüştürülür.
- Nitelikler tabloların sütunlarına dönüştürülür.

- Çok değer içeren her bir nitelik için tablo oluşturulur. Oluşturulan tabloya çok değerli nitelik ve bağlı bulunduğu varlık kümesinin birincil anahtarları yabancı anahtar sütun olarak eklenir.
- Oluşturulan çok değerli nitelik tablosunun birincil anahtarı eklenen çok değerli nitelik sütunu ile yabancı anahtar sütununun birleşimidir.

Örnek: Bu kurallar doğrultusunda aşağıda verilen varlık-ilişki modelini tabloya dönüştürelim.



Personel varlık kümesini *personel* ve *y_dil* çok değerli niteliği de *y_dil* tablosu olarak isimlendirelim.

Personel tablosunun sütunları *per_id*, *ad*, *görev* ve *maaş* olacaktır.

y_dil tablosunun sütunları *y_dil* ve *per_id* olacaktır.

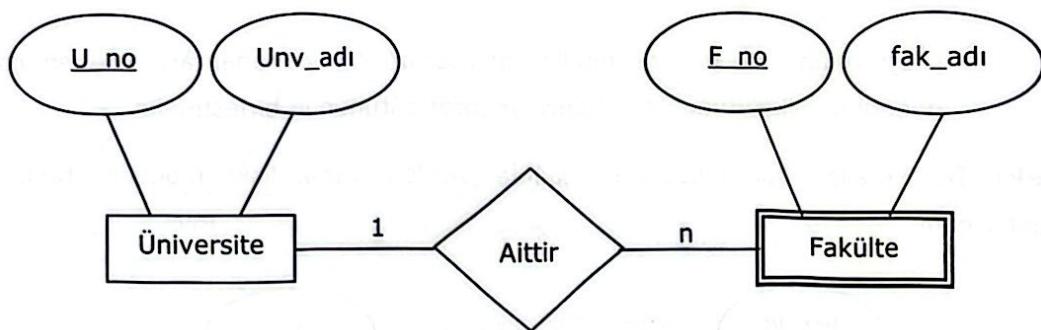
Personel Tablosu		y_dil Tablosu	
PK	<u>per_id</u>	PK	<u>y_dil</u>
	ad görev maaş	1	n

e. Zayıf Varlık Kümelerinin Tabloya Dönüştürülmesi

Zayıf varlık kümelerinin tablolara dönüştürülmesi 1-n ilişkisinin tabloya dönüştürülmesi ile aynıdır.

Zayıf varlık kümeleri ile oluşturulan ilişki tablolara dönüştürüldükten sonra güçlü varlığın sahip olduğu anahtar nitelik yani birincil anahtar zayıf varlık içerisinde yabancı anahtar olarak kullanılır.

Örnek: Aşağıda zayıf varlığa sahip bir ilişki verilmiştir. Verilen ilişkiyi tablolara dönüştürelim.



İlk olarak üniversite varlığı, u_no ve unv_adi sütunlarını içeren üniversite isminde bir tablo olarak ve fakülte varlığı, f_no ve fak_adi sütunlarını içeren fakulte isminde bir tablo olarak tanımlanır. Daha sonra güçlü varlığı gösteren üniversite tablosunun birincil anahtarı, zayıf varlığı gösteren fakulte tablosuna yabancı anahtarlar olarak eklenir. Fakulte tablosunda oluşan f_no ile u_no birleştirilerek fakulte tablosu için birincil anahtar olarak belirlenir.

Universite Tablosu

PK	<u>u_no</u>
	unv_adi

Fakulte Tablosu

PK	<u>u_no</u>
PK	<u>f_no</u>
	fak_adi

Relationship: 1 to n

2.3. VERİTABANI NORMALİZASYON KURALLARI

Normalizasyon, veritabanının tasarım aşamasında veri tekrarını, veri kaybını veya veri yetersizliğini önlemek için gerçekleştirilen işlemlerdir.

Tabloların oluşturulması, tablolar arasında artıklığı ve tutarsız bağımlılığı ortadan kaldırma normalizasyonun temel amacıdır.

Veritabanı normalleştirilmesi için birkaç kural bulunmaktadır. Her kurala "normal form" adı verilir. İlk kural kullanılıyorsa, veritabanının "ilk normal formda" olduğu söylenir. İlk üç kural kullanılıyorsa, veritabanı "Üçüncü normal formda" olarak nitelendirilir. Başka normalleştirme düzeyleri de kullanılabilmekle birlikte, çoğu uygulama için en yüksek düzey üçüncü normal formdur ama daha üst seviye olan dört ve beşinci normalizasyon kuralları da vardır.

Genel olarak, normalleştirme için ek tablolar gereklidir ve çoğunlukla bunun ek yük getirdiğini düşünülebilir. Normalleştirmenin ilk üç kuralından biri ihlal edildiği takdirde, uygulamalarda artık verilerin oluşturabileceği sorumlulara karşı hazır olunması gereklidir.

Normalizasyon, çok fazla satır ve sütunlardan oluşan bir tabloyu veri tekrarından korumak için daha az satır ve sütun içeren tablolara ayırma işlemidir.

Bu kurallar, depolanacak verilerin neler olacağını belirler ve kayıtlar arasında veri tekrarını, veri kaybını ve yetersizlikleri engeller. Normalizasyon kurallarına uygun tasarım yapılmadığında kayıt güncelleme, kayıt bulma ve yeni kayıt işlemlerinde sorunlarla karşılaşılabilir.

Üst normalizasyon kuralları alt normalizasyon kurallarını kapsamaktadır. Yani, 2NF kuralları 1NF kurallarını ve 3NF kuralları 1NF, 2NF kurallarını kapsar.

2.3.1. Normalizasyonun Amaçları

Veri Bütünlüğünü Sağlamak: Eğer bir sütun için gereksiz veri tekeri var ise, bu sütun bir süre sonra birbirinden farklı değerler içermeye başlayacaktır. Örneğin, bir veritabanı içerisinde öğrenci numarası birden fazla tabloda kullanıldığında veri bütünlüğü sağlanmazsa zamanla aynı öğrenci için birden fazla numara bilgisi oluşacaktır. Böyle bir durumla karşılaşmamak için normalizasyon kuralları ile veri bütünlüğü sağlanabilir.

Uygulamadan Bağımsızlık: Hazırlanacak ilişkisel model kullanılacak uygulamaya göre değil içereceği veriye göre hazırlanmalıdır. Böylece, kullanılan uygulama değişse bile veri modeli tutarlı olarak çalışmaya devam edecektir.

Performansı Artırmak: Tam olarak normalleştirilmiş veritabanı veri tekrarını en aza indireceği için kullanılan disk alanı ve veritabanı boyutu da azalacaktır. Böylece, veritabanı içerisinde gerçekleştirilen arama işlemleri daha kısa sürede gerçekleşecektir.

2.3.2. Fonksiyonel Bağımlılık

R ilişkisi X ve Y nitelik kümelerinden oluşmaktadır. Eğer X nitelik kümelerinin değerleri Y nitelik kümelerinin değerlerini belirliyorsa, Y niteliği X niteliğine fonksiyonel bağımlıdır ve $X \rightarrow Y$ şeklinde gösterilir. X 'in her bir değeri Y 'nin bir değerine karşılık geldiği durumlarda da **fonksiyonel bağımlılık** söz konusudur.

Eğer X nitelik kümelerinden bir nitelik çıkarıldığı halde bağımlılık devam ediyorsa **kısıtlı bağımlılık** söz konusudur.

Örnek: Örencilere ait bölüm ve sınav bilgilerini tutan bir tablo ogr_no, bolum_kodu, bolum_adi, ders_kodu ve sınav sütunlarından oluşmaktadır. Aşağıda bu tabloya ait fonksiyonel bağımlılıklar gösterilmiştir.

OGRENCI(ogr_no, bolum_kodu, bolum_adi, ders_kodu, sınav)

Bu tablo için her bir öğrenci aynı dersi bir kez alabileceği için ve veri tekrarını önlemek için ogr_no ve ders_kodu tablo için birleşik anahtar olarak tanımlanmıştır.

Bu tablo üzerindeki bağımlılıklar aşağıda verilmiştir.

Bolum ve bolum_kodu bilgilerinin oluşabilmesi için ogr_no tanımlanması gerekmektedir. Bu nedenle $ogr_no \rightarrow bolum_kodu$, $bolum$ bağımlılığı vardır.

Bolum_kodunun olması bolum bilgisinin var olmasını gerektirdiği için

$bolum \rightarrow bolum_kodu$ bağımlılığı vardır. Bu bağımlılık anahtar sütuna bağımlı olmadığı için **geçişli bağımlılıktır**.

Sınav bilgisinin var olması öğrenci ve derse bağımlı olduğu için

$(ogr_no, ders_kodu) \rightarrow sınav$ bağımlılığı vardır.

2.3.3. Normalizasyon Kuralları

1. Normal Form(1NF):

Veritabanın daha az yer kaplaması için geliştirilen ilk normalizasyon kuralıdır.

Kurallar:

- Veritabanında bulunan tablolar ilişkilendirilebilir bir şekilde tasarılanmalıdır.
- Birden fazla türde bilgi tek bir sütunda olamaz.
- Bir alan içerisindeki bilgi özel karakterlerle ayrılarak tutulmamalıdır.

Bir veritabanı yukarıda belirtilen kurallara uyum sağlıyorsa 1NF kuralına uymaktadır.

Örnek: Aşağıda ilk tasarımlı yapılan öğrenci tablosu verilmiştir.

Ogr_no	Bolum_kodu	Bolum	Ders_kodu	Sınav
009001	BTP	Bilgisayar	B1, B2, B3	75, 85, 45
009002	BTP	Bilgisayar	B2, B3, B4	25, 60, 55
009003	ELK	Elektrik	E1, E5, E4	45, 66, 74
009004	ELK	Elektrik	E5, E4, E8	66, 78, 75

Yukarıda verilen tablo incelendiğinde ders_kodu ve sınav sütunlarının atomik değerler içermemiği görülmektedir. Verilen bu tablo normal olmayan formdadır.

Bölüm 2: İlişkisel Veritabanı Tasarımı ve Normalizasyon

OGRENCI tablosuna ilk normalizasyon kuralı uygulandığında atomik olmayan değerler yeni satırlarda gösterilecektir. Tablonun 1NF kuralına uygun biçimini aşağıda verilmiştir.

Ogr_no	Bolum_kodu	Bolum	Ders_kodu	Sınav
009001	BTP	Bilgisayar	B1	75
009001	BTP	Bilgisayar	B2	85
009001	BTP	Bilgisayar	B3	45
009002	BTP	Bilgisayar	B2	25
009002	BTP	Bilgisayar	B3	60
009002	BTP	Bilgisayar	B4	55
009003	ELK	Elektrik	E1	45
009003	ELK	Elektrik	E5	66
009003	ELK	Elektrik	E4	74
009004	ELK	Elektrik	E5	66
009004	ELK	Elektrik	E4	78
009004	ELK	Elektrik	E8	75

1.Normal Formun Sorunları

1.Normal Form sonucu elde edilen tablo tekrarlı satırlara sahip olacağı için satır ekleme, satır silme ve satır güncelleme sorunları ile karşılaşabilir.

Satır Ekleme Sorunu

Örnekte 1.Normal Forma uygun şekli verilen öğrenci tablosu incelendiğinde yeni bir öğrenci veya bölüm tanımlaması yapılmamak istenirse ders kodu ve sınav bilgilerinin girilmesi zorunludur.

Satır Silme Sorunu

Öğrenci tablosu içerisinde bir bölüm için sadece bir öğrenci kayıtlı olursa öğrenci silindiğinde bölüm de silinecektir.

Satır Güncelleme Sorunu

Bir öğrencinin bölümü değiştiğinde birden fazla kaydın güncellenmesi gerekecektir. Örneğin, 009002 nolu öğrencinin bölümü BTP yerine ELK yapılmak istenirse 3 satırın ve 6 sütunun güncellenmesi gereklidir. Bu durum, çok fazla veri içeren veritabanları için ciddi performans sorunlarına neden olacaktır.

2.Normal Form(2NF):

1NF'de karşılaşılan sorunlardan özellikle güncelleme sorununu çözmek için tablo 2NF kuralına uygun olmalıdır. 2NF, nitelikler arasındaki fonksiyonel bağımlılıktan yararlanılarak tabloların birden fazla tabloya dönüştürülmesi ile sağlanır.

Kurallar:

- Bir tablo içinde tanımlı ve anahtar olmayan her sütun birincil anahtar olarak tanımlı anahtar sütunlara bağımlı olmalıdır. Anahtar sütunun **ihtiyaç** duyduğu bilgileri içermelidir. Örneğin, öğrenci bilgilerinin tutulduğu **bir** tabloda not ve ders bilgisinin olması gereksizdir. Çünkü notlarla **ilgili asıl** kılavuz öğrenci numarası olacaktır, bölüm veya ad gibi bilgiler ayrıntı olarak kalacaktır. Bunu çözmek için öğrenci bilgileri ve not bilgileri ayrı tablolarda tutulmalıdır.
- Anahtar sütun birden fazla sütunun birleşiminden oluşuyorsa tabloda **yer** alacak veriler iki sütuna da bağımlı olmalıdır. Tek sütuna bağımlı ise **ayrı bir** tabloda tutulmalıdır. Örneğin, bir firma farklı şehirlere satış yapmaktadır. Veri tekrarının önlenmesi için satış tablosunda müşteri no ve satış no alanları birleştirilmiş anahtar olarak tanımlanmıştır.

Satışlar Tablosu			
M.No	M.Adı	S.No	Tutar
002	Ahmet	2009001	5000,00
006	Murat	2009002	125000,00
005	Ayşe	2009003	500,00
002	Ahmet	2009004	12000,00

The diagram shows two arrows originating from the 'M.No' and 'M.Adı' columns of the table. One arrow points diagonally down and to the right, and the other points vertically down, both pointing towards a rectangular box at the bottom labeled 'Birleştirilmiş Anahtar Alanlar'.

Yukarıda vermiş olduğumuz tablo 1NF kuralına uymakta ama 2NF kuralına uymamaktadır. Birleştirilmiş anahtar sütunlar olduğunda diğer sütunların anahtar sütunların ikisine de bağımlı olması gerekir. Ancak "Tutar" sütununun tek başına "M.No" sütunu ile, "M.Adı" sütununun tek başına "S.No" sütunu ile bir bağı yoktur. Bu tablomuzu kuralımıza uyacak hale getirmek için aşağıdaki gibi iki ayrı tabloya bölmeliyiz.

Müşteri Tablosu	
M.No	M.Adı
002	Ahmet
006	Murat
005	Ayşe
002	Ahmet

Satışlar Tablosu		
M.No	S.No	Tutar
002	2009001	5000,00
006	2009002	125000,00
005	2009003	500,00
002	2009004	12000,00

Örnek: 1NF kuralına uygun olan ama 2NF kuralına uymayan OGRENCI tablosunu 2NF kuralına uygun hale getirmek için ders ve not bilgileri ayrı tablolarda tanımlanmalıdır. Buna göre OGRENCI tablosunun 2NF kuralına uygun durumu aşağıda verilmiştir.

OGRENCI		
Ogr_no	Bolum_kodu	Bolum
009001	BTP	Bilgisayar
009002	BTP	Bilgisayar
009003	ELK	Elektrik
009004	ELK	Elektrik

NOTLAR		
Ogr_no	Ders_kodu	Sınav
009001	B1	75
009001	B2	85
009001	B3	45
009002	B2	25
009002	B3	60
009002	B4	55
009003	E1	45
009003	E5	66
009003	E4	74
009004	E5	66
009004	E4	78
009004	E8	75

2.Normal Formun Sorunları

2.Normal Form sonucu güncelleme sorunu çözülür ama ekleme ve satır silme sorunu devam etmektedir.

Satır Ekleme Sorunu

Örnekte 2.Normal Forma uygun şekli verilen öğrenci tablosu incelendiğinde yeni bir bölüm tanımlaması yapılabilmesi için öğrenci tanımlaması zorunludur.

Satır Silme Sorunu

Öğrenci tablosu içerisinde bir bölüm için sadece bir öğrenci kayıtlı olursa öğrenci silindiğinde bölüm de silinecektir.

3.Normal Form(3NF):

2NF'de karşılaşılan sorunları çözmek için geçişli bağımlılıkları da ortadan kaldırmak gerekmektedir.

2NF'de sadece anahtar sütunlara göre bağımlılıklar kullanılmıştı, 3NF'de ise bir tablo içerisinde anahtar olmayan bir sütun, başka bir tablonun anahtar sütunu veya bulunduğu tablonun sütunlarıyla ilgili olmalıdır veya bir tablo için anahtar olmayan bir sütun anahtar olmayan başka hiçbir sütuna bağımlı olamaz. Ayrıca, veritabanındaki ilişkiler 2NF kuralına uymalıdır.

Örneğin, ürünler tablosunda ürün kodu, ürün adı ve birimi bilgileri tutuluyor. Girilecek birimler kg, cm ve litre değerleri ise bu sütunun karşılık geldiği bir tablo bulunmalıdır. Eğer birim sütunu da ürünler tablosunda bulunuyorsa kuralımıza aykırıdır.

Birim Tablosu	
No	Birim Adı
001	Kg
002	Cm
003	Litre

Ürün Tablosu		
Ü.Kodu	Adı	Birim
K001	Alçı	001
K002	Tel	002
K003	Boya	003

Örnek: OGRENCI tablosu için bolum→bolum_kodu geçişli bağımlılığı mevcuttur. OGRENCI tablosundaki sorunlardan kurtulmak için bu bağımlılığından kaldırılması gereklidir. Bağımlılığı kaldırmak için bölümler ayrı bir tablo olarak oluşturulmalıdır.

OGRENCI	
Ogr_no	Bolum_kodu
009001	BTP
009002	BTP
009003	ELK
009004	ELK

BOLUMLER	
Bolum_kodu	Bolum
BTP	Bilgisayar
BTP	Bilgisayar
ELK	Elektrik
ELK	Elektrik

NOTLAR		
Ogr_no	Ders_kodu	Sınav
009001	B1	75
009001	B2	85
009001	B3	45
009002	B2	25
009002	B3	60
009002	B4	55
009003	E1	45
009003	E5	66
009003	E4	74
009004	E5	66
009004	E4	78
009004	E8	75

Bölüm 2: İlişkisel Veritabanı Tasarımı ve Normalizasyon

Örnek: Aşağıda verilen sipariş tablosu 1NF kuralına uymaktadır.

Müşteri	Ürün_id	Şehir	Şehir_kodu	Miktar
M145	U1	Tokat	60	500
M458	U1	Amasya	05	450
M478	U2	Samsun	55	700
M458	U2	Amasya	05	350
M145	U2	Tokat	60	200
M478	U1	Samsun	55	145

Bu tabloda, anahtar sütun müşteri ve ürün_id sütunlarının birleşimidir. Burada miktar sütunu bu iki sütunun birleşiminden oluşan anahtara bağımlıdır. Şehir ve şehir_kodu sütunları ise sadece müşteri sütununa bağımlıdır, ürün_id sütunu ile bir bağlantısı yoktur. Tabloyu 2NF kuralına uydurmak için şehir ve şehir_kodu sütunları ayrı bir tabloya ayrılmalıdır. Sipariş tablosu, 2NF kuralına uyması için aşağıda verilen iki ayrı tabloya bölünmüştür.

Müşteri	Ürün_id	Miktar
M145	U1	500
M458	U1	450
M478	U2	700
M458	U2	350
M145	U2	200
M478	U1	145

Müşteri	Şehir	Şehir_kodu
M145	Tokat	60
M458	Amasya	05
M478	Samsun	55

Oluşturulan ikinci tabloda anahtar sütun müşteri sütunudur ve anahtara bağımlı olmayan sütun bulunmaktadır. Şehir ve şehir_kodu sütunları arasında bağımlılık vardır ama bunlardan herhangi biri anahtar olmadığı için 3NF kuralına uymaz. Oluşturulan yeni tablo da ikiye bölündüğünde 3NF kuralına da uyum sağlanmış olur. Tablonun aşağıda verilen son hali 3NF kuralına doğal olarak da 2NF ve 1NF kuralına da uyum sağlamıştır.

Müşteri	Ürün_id	Miktar
M145	U1	500
M458	U1	450
M478	U2	700
M458	U2	350
M145	U2	200
M478	U1	145

Müşteri	Şehir_kodu
M145	60
M458	05
M478	55

Şehir_kodu	Şehir
60	Tokat
05	Amasya
55	Samsun

Boyce-Codd Normal Form(BCNF):

Eğer bir tablo 3NF kuralına uygunsa ve her belirleyici aday anahtar ise tablo BCNF kuralına uygundur. Belirleyici, bağımlılık işaretinin solundaki özellik veya özellik grubudur. Örneğin, $A \rightarrow B$ bağımlılığında belirleyici A'dır.

BCNF kuralına uygun bir tablo 1NF, 2NF ve 3NF'ye de uygundur ama 3NF'ye uygun olduğunda BCNF'ye uygun olamaz.

Örnek: Aşağıdaki tablo hasta ve randevu bilgilerini içermektedir.

Hasta_no	Hasta_ismi	Randevu_durumu	Saat	Doktor
1	Ayşe	0	09:00	Ufuk
2	Murat	0	09:00	Aytaç
3	Mehmet	1	10:00	Ufuk
4	Burcu	0	13:00	Aytaç
5	Deniz	1	14:00	Ufuk

Yukarıda verilen tablo özel bir diyet uygulayan kliniğe aittir. Her hasta 4 aşamalı (0, 1, 2, 3) bir tedaviden geçer. 0 nolu randevu durumu saat 09:00 veya 13:00'de, 1 nolu randevu saat 10:00 veya 14:00'dedir. 2 ve 3 nolu randevular da bu şekilde devam etmektedir.

Verilen bilgiler doğrultusunda bu tabloya ait fonksiyonel bağımlılıklar aşağıdaki gibidir.

$\text{Hasta_no} \rightarrow \text{Hasta_ismi}$

$\text{Hasta_no}, \text{randevu_durumu} \rightarrow \text{saat}, \text{doktor}$

$\text{Saat} \rightarrow \text{randevu_durumu}$

Verilen bağımlılıklara göre tablo için iki farklı Primary Key belirlenebilir. Bunlar;

Sonuç1: DB(hasta_no, hasta_ismi, randevu_durumu, saat, doktor)

Veya

Sonuç2: DB(hasta_no, hasta_ismi, randevu_durumu, saat, doktor)

Elde edilen sonuçları normalizasyon kurallarına göre inceleyelim.

Sonuç1 - DB(hasta_no, hasta_ismi, randevu_durumu, saat, doktor)

1NF: Uygun çünkü tekrar eden gruplar yok.

2NF: Hasta isminin hasta_no ile bağımlılığı vardır ama randevu_durumu ile bağımlılığı yoktur. Bu nedenle hasta_ismi tablo ikiye bölünerek yeni bir tabloda tutulduğunda 2NF kuralı uygun hale getirilmiş olur.

DB(hasta_no, randevu_durumu, saat, doktor)

DB2(hasta_no, hasta_ismi)

3NF: Geçişli bağımlılıklar bulunmadığı için son durum 3NF kuralına uygundur.

BCNF: Her belirleyici bir anahtar olmalıdır.

Bağımlilik ifadelerinin sol tarafında bulunan nitelikler belirleyicidir.

Hasta_no → Hasta_ismi: Hasta_no belirleyicisi DB içerisinde anahtar durumunda olduğu için uygundur.

Hasta_no, randevu_durumu → saat, doktor: Hasta_no, randevu_durumu belirleyicisi DB içerisinde anahtar durumunda olduğu için uygundur.

Saat → randevu_durumu: saat belirleyicisi DB içerisinde anahtar durumunda olmadığı için BCNF'ye uygun değildir. BCNF'ye uygun duruma getirmek için son durumu aşağıdaki gibi olmalıdır.

DB(hasta_no, saat, doktor)

DB2(hasta_no, hasta_ismi)

DB3(saat, randevu_durumu)

Sonuç2: DB(hasta_no, hasta_ismi, randevu_durumu, saat, doktor)

1NF: Uygun çünkü tekrar eden gruplar yok.

2NF: Hasta isminin hasta_no ile bağımlılığı vardır ama saat ile bağımlılığı yoktur. Randevu durumunun saat ile bağımlılığı vardır ama hasta_no ile bağımlılığı yoktur. Bu nedenle hasta_ismi tablo yeni bir tabloda tutulduğunda ve randevu_durumu yeni bir tabloda tutulduğunda 2NF kuralına uygun hale getirilmiş olur.

DB(hasta_no, saat, doktor)

DB2(hasta_no, hasta_ismi)

DB3(saat, randevu_durumu)

3NF: Geçişli bağımlılıklar bulunmadığı için son durum 3NF kuralına uygundur.

BCNF: Tüm belirleyiciler anahtar konumunda olduğu için BCNF'ye uygundur.

4.Normal Form(4NF):

Birincil anahtar olan sütunlar ile anahtar olmayan sütunlar arasında birden fazla bağımsız 1-n ilişkiye izin verilmez. 4NF sağlamak için her bağımsız 1-n ilişki için ayrı tablo oluşturmak gereklidir.

Örnek: Aşağıda öğretim elemanları bilgilerinin tutulduğu bir tablo verilmiştir.

Ogrt_el	Ad	Soyad	Bolum_id	Ders
A123	Turgut	Özseven	BTP	Veritabanı
A124	Ahmet	Kaçar	ELK	Elektrik Mak.
A125	Mustafa	Çağlayan	MAK	Teknik Resim
A126	Murat	Kaçmaz	BIO	Enformatik

Tablo incelendiğinde ilk olarak herhangi bir sorun olmadığı görülür. Bu tablo için **birincil anahtar** ogrt_el sütunudur. Bu tabloya göre her öğretim elemanı sadece bir derse girebilir. Aynı öğretim elemanı için iki veya daha fazla ders girilmek istendiğinde birincil anahtardan dolayı aynı ogrt_el kodu kullanılamayacak dolayısıyla da yeni ders girilemeyecektir ve 1-n ilişki söz konusu olmayacağındır. 4NF kuralını ve 1-n ilişkiliyi sağlamak için mevcut tablo iki ayrı tabloya bölünür. Aşağıda 4NF kuralına uygun tablolar verilmiştir.

Ogrt_el	Ad	Soyad	Bolum_id
A123	Turgut	Özseven	BTP
A124	Ahmet	Kaçar	ELK
A125	Mustafa	Çağlayan	MAK
A126	Murat	Kaçmaz	BIO

Ogrt_el	Ders
A123	Veritabanı
A124	Elektrik Mak.
A125	Teknik Resim
A126	Enformatik
A123	Internet Prog.I
A123	Donanım

5.Normal Form(5NF):

Tekrarları önlemek için her tabloyu mümkün olduğunda küçük parçalara bölmek gereklidir. Aslında ilk 4 kural bu işlemi gerçekleştirir ama bu kurallar kapsamında olmayan tekrarlamlar da beşinci normalizasyon kuralı ile giderilir.

Örnek: Aşağıda ürün, marka ve satış danışmanı bilgisini içeren bir tablo verilmiştir.

Satış Danışmanı	Marka	Ürün
D001	ABC	U001
D001	ABC	U002
D001	CDE	U001
D002	CDE	U003
D002	EFG	U001
D003	ABC	U003
D003	CDE	U004

Tablo incelendiğinde marka ve ürün bilgisi satış danışmanına göre değişmektedir ve rastgele değerler değildir. Marka ve ürün bilgilerini başka tabloda tutarak satış danışmanı için belirlenen marka ve ürün dışında veri girişi engellenebilir.

Satış Dan.	Ürün
D001	U001
D001	U002
D001	U001
D002	U003
D002	U001
D003	U003
D003	U004

Satış Dan.	Marka
D001	ABC
D001	CDE
D002	CDE
D002	EFG
D003	ABC
D003	CDE

Marka	Ürün
ABC	U001
ABC	U002
ABC	U003
CDE	U001
CDE	U003
CDE	U004
EFG	U001

Bir Tabloyu Normalleştirme Örnek 1

Aşağıda verilen tablo normalleştirilmemiş bir tablodur. Bu tablo üzerinde normalizasyon kurallarını uygulayalım(<http://support.microsoft.com>).

1. Normalleştirilmemiş tablo.

Ogr_no	Danisman	Oda_no	Ders1	Ders2	Ders3
1022	Güneş	412	101-07	143-01	159-02
4023	Etikan	216	201-01	211-02	214-01

2. İlk Normal Form(1NF): Yinelenen grup yok.

Tablolar yalnızca iki boyutlu olmalıdır. Aynı öğrenci birden çok ders alacağı için, bu dersler ayrı bir tabloda listelenmelidir. Yukarıdaki kayıtlardaki Ders1, Ders2 ve Ders3 alanları bir tasarım sorununun göstergesidir. İlk normal formda yinelenen grubu (Ders) aşağıdaki gösterildiği gibi kaldırarak başka bir tablo oluşturun:

Ogr_no	Danisman	Oda_no	Ders
1022	Güneş	412	101-07
1022	Güneş	412	143-01
1022	Güneş	412	159-02
4023	Etikan	216	201-01
4023	Etikan	216	211-02
4023	Etikan	216	214-01

3. İkinci Normal Form(2NF): Artık verileri kaldırma.

Yukarıdaki tabloda her ogr_no değeri için birden çok Ders değeri vardır. Ders ögesi işlevsel olarak ogr_no ögesine (birincil anahtar) bağımlı olmadığı için, bu ilişki ikinci normal formda değildir.

Aşağıdaki iki tabloda ikinci normal form gösterilmektedir:

Öğrenciler:

Ogr_no	Danisman	Oda_no
1022	Güneş	412
4023	Etikan	216

Kayıt:

Ogr_no	Ders
1022	101-07
1022	143-01
1022	159-02
4023	201-01
4023	211-02
4023	214-01

4. Üçüncü Normal Form(3NF): Anahtara bağımlı olmayan verileri kaldırma.

Sonörnekte, Oda No ögesi (danışmanın oda numarası) Danışman öz niteliğine işlevsel olarak bağımlıdır. Çözüm, bu öz niteliği aşağıda gösterildiği gibi Öğrenciler tablosundan Fakülte tablosuna taşımaktır:

Öğrenciler:

Ogr_no	Danisman
1022	Güneş
4023	Etikan

Fakülte:

Ad	Oda	Bolum
Güneş	412	42
Etikan	216	42

Bir Tabloyu Normalleştirme Örnek 2

Aşağıda verilen tablo normalleştirilmemiş bir tablodur. Bu tablo üzerinde normalizasyon kurallarını uygulayalım.

1. Normalleştirilmemiş tablo.

Ogr_no	Ogr_adi	Alan	bolum	danisman
0900001	Ayhan Ayhan	Matematik	MatFen	Danışman1
0900002	Ayşe Bilmez	Türkçe	TürkceMat	Danışman2
0900003	Murat Ermez	Matematik	MatFen	Danışman1
0900004	Fuat Han	Biyoloji	MatFen	Danışman3

2. İlk Normal Form(1NF): Yinelenen grup yok

Verilen tabloda tekrar eden kayıtlar ve sütunlar bulunmadığı için 1NF kuralına uygundur. Ama bir öğrenci birden fazla bölüme kayıt olabiliyorsa 1NF kuralına uymayacaktır.

3. İkinci Normal Form(2NF): Artık verileri kaldırma

Yukarıdaki tabloda birincil anahtar ogr_no sütunudur. 2NF kuralına göre birincil anahtar haricindeki sütunların birincil anahtara yani ogr_no sütununa bağlı olması gereklidir.

Yukarıdaki tablo için ogr_no sütununa ogr_adi sütunu bağlıdır ama öncelik diğer sütunların bağımlılığı olduğu için öğrenci isimleri ayrı bir tabloda tutulmalıdır.

Aşağıdaki iki tabloda ikinci normal form gösterilmektedir:

Ogr_no	Alan	bolum	danisman
0900001	Matematik	MatFen	Danışman1
0900002	Türkçe	TürkceMat	Danışman2
0900003	Matematik	MatFen	Danışman1
0900004	Biyoloji	MatFen	Danışman3

Ogr_no	Ogr_adi
0900001	Ayhan Ayhan
0900002	Ayşe Bilmez
0900003	Murat Ermez
0900004	Fuat Han

4. Üçüncü Normal Form(3NF): Anahtara Bağımlı Olmayan Verileri Kaldırma

Elde edilen son duruma göre bolum sütunu ile alan sütunun bağımlılığı vardır ama bolum sütunu ile ogr_no sütunu arasında bağımlılık yoktur. Aynı şekilde danışman sütunu ile de ogr_no sütunu arasında da bağımlılık yoktur.

Alan ve bölüm bilgilerini tutmak için iki tabloya ek olarak üçüncü bir tablo oluşturulursa 3NF kuralına uyum sağlanmış olacak.

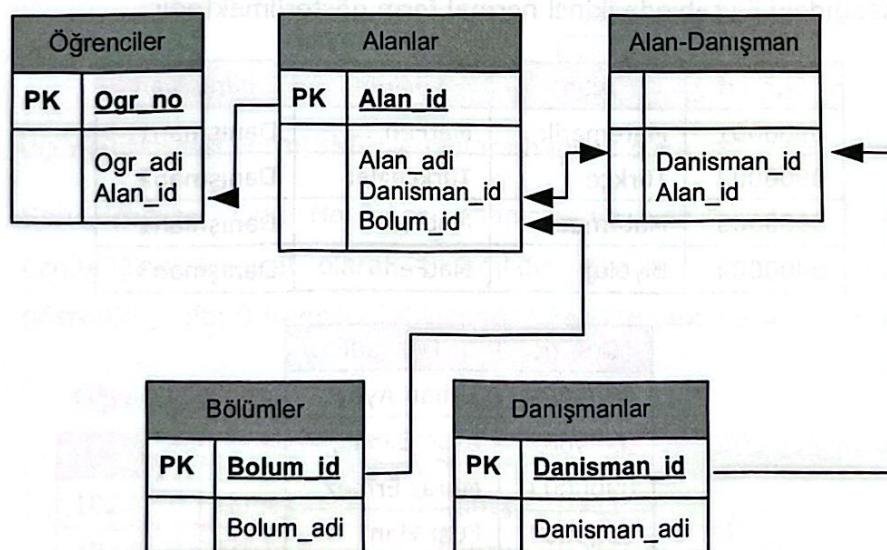
Ogr_no	Alan
0900001	Matematik
0900002	Türkçe
0900003	Matematik
0900004	Biyoloji

Ogr_no	Ogr_adi
0900001	Ayhan Ayhan
0900002	Ayşe Bilmez
0900003	Murat Ermez
0900004	Fuat Han

Alan	bolum	danisman
Matematik	MatFen	Danışman1
Türkçe	TürkceMat	Danışman2
Biyoloji	MatFen	Danışman3

Elde ettiğimiz son tablolar 3NF kuralı ile tam olarak normalleştirilmiştir. 4NF ve 5NF kurallarına gerek yoktur. Ancak, bir alanın birden fazla danışmanı olabiliyorsa 4NF kuralı uygulanmalıdır. Bunun için de danışman bilgileri ayrı bir tabloda tutulmalıdır.

Olabilecek tüm ihtimaller göz önüne alınarak aşağıda görüldüğü gibi tablolar oluşturulursa herhangi bir sorunla karşılaşılmayacaktır.

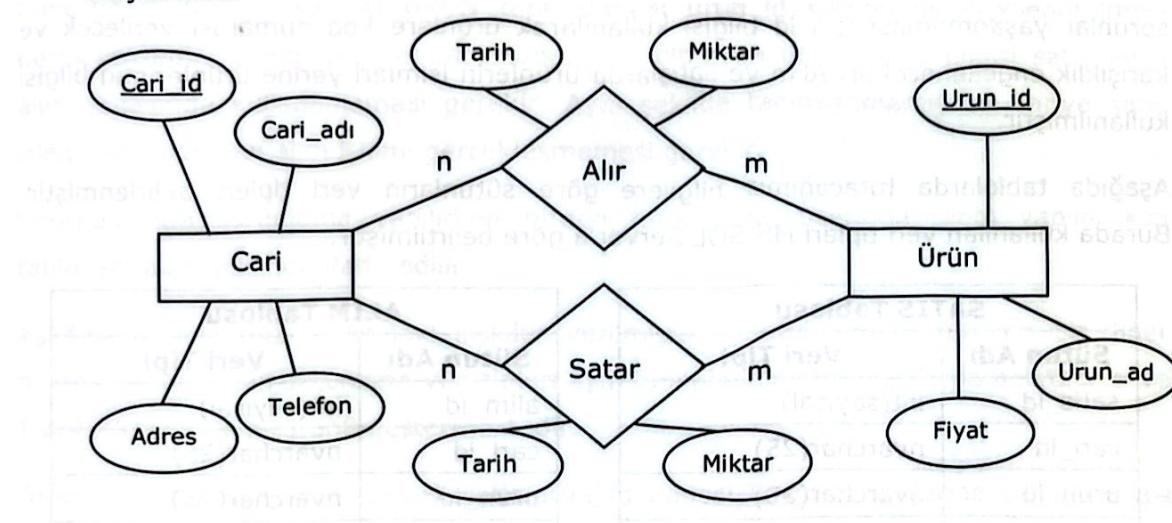


2.4. ÖRNEK VERİTABANI TASARIMI

Örnek 1:

Veritabanı kullanılan projeler, projeyi oluşturmaya başlamadan önce veritabanının tasarımının yapılması gerekmektedir. Hazırlayacağımız veritabanı bir firmanın ürünlerini, ürün alımlarını, satışlarını ve cari bilgilerini tutmaktadır.

- İlk olarak ürünler ve carilerin durumunu belirten varlık-ilişki modelini oluşturalım.



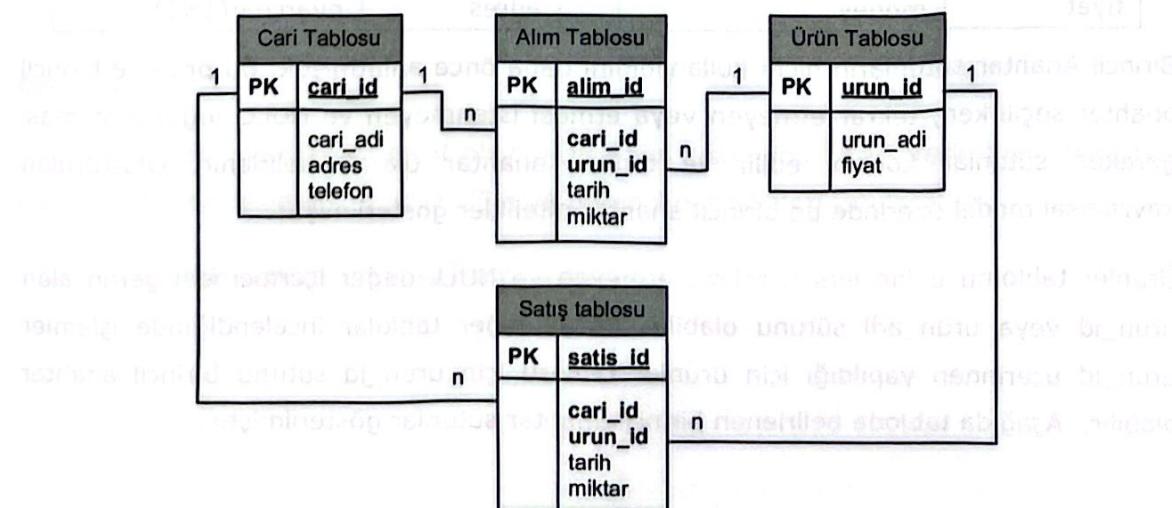
- Oluşturulan varlık-ilişki modelini tabloya dönüştürelim.

CARI(cari_id, cari_adi, adres, telefon)

URUN(urun_id, urun_adi, fiyat)

ALIM(alim_id, cari_id, urun_id, tarih, miktar)

SATIS(satis_id, cari_id, urun_id, tarih, miktar)



Cari tablosu firmanın alış veriş yaptığı firma veya kişilerin bilgilerini tutacaktır. Aynı isme sahip birden fazla cari olabileceği için carilerin işlemleri cari_id alanına göre yapılacaktır. Cari_id alanı kullanılmayıp sadece cari_adi kullanılırsa bilgilerde karışıklık meydana gelir. Örneğin, Ahmet ismine sahip birden fazla cariniz varsa yapılan satış veya alımın hangisine ait olduğunu tespit edemezsınız. Ayrıca satış yaparken ismin yanlış yazılması durumunda sorgulamalarda yanlış sonuçlarla karşılaşılabilir. Aynı şekilde satış ve alım tablosunda da bu yöntem kullanılmıştır.

Ürünler tablosunda da ürün isimleri ve urun_id leri ile tanımlanmıştır. Ürün isimlerinde sorunlar yaşanmaması için id bilgisi kullanılarak ürünlerde kod numarası verilecek ve karışıklık engellenecektir. Alım ve satışlarda ürünlerin isimleri yerine ürünlerin id bilgisi kullanılmıştır.

Aşağıda tablolarda tutacağımız bilgilere göre sütunların veri tipleri belirlenmiştir. Burada kullanılan veri tipleri MS SQL Server'a göre belirtilmiştir.

SATIS Tablosu	
Sütun Adı	Veri Tipi
satis_id	int(sayısal)
cari_id	nvarchar(25)
urun_id	nvarchar(25)
tarih	smalldatetime(tarih)
miktar	int(sayısal)

ALIM Tablosu	
Sütun Adı	Veri Tipi
alim_id	int(sayısal)
cari_id	nvarchar(25)
urun_id	nvarchar(25)
tarih	smalldatetime(tarih)
miktar	int(sayısal)

ÜRÜN Tablosu	
Sütun Adı	Veri Tipi
urun_id	nvarchar(25)
urun_adi	nvarchar(25)
fiyat	money

CARI Tablosu	
Sütun Adı	Veri Tipi
cari_id	nvarchar(25)
cari_adi	nvarchar(50)
adres	nvarchar(150)

Birincil Anahtar sütunların niçin kullanıldığı daha önce anlatılmıştır. Bu projede birincil anahtar seçilirken, tekrar etmeyen veya etmesi istenmeyen ve NULL değer almaması gereken sütunlar kontrol edilir ve birincil anahtar olarak belirlenir. Oluşturulan kavramsal model üzerinde de birincil anahtar nitelikler gösterilmiştir.

Ürünler tablomuzu incelersek tekrar etmeyen ve NULL değer içermemesi geren alan urun_id veya urun_adi sütunu olabilir. Ancak diğer tablolar incelendiğinde işlemler urun_id üzerinden yapıldığı için ürünler tablosu için urun_id sütunu birincil anahtar olabilir. Aşağıda tabloda belirlenen birincil anahtar sütunlar gösterilmiştir.

Tablo Adı	Birincil Anahtar (Primary Key)
SATIŞ	satis_id
ALIM	alim_id
ÜRÜN	urun_id
CARI	cari_id

Kavramsal modelde de gösterildiği gibi tablolar arasında ortak sütunlar vardır. Tabloların ortak sütunlarının birbiri ile ilişkilendirilmesi gerekmektedir. Örneğin, SATIŞ tablosunda urun_id sütunu ÜRÜN tablosundaki urun_id sütunu ile ilişkilendirilmesi gerekmektedir. Çünkü ürünler tablosunda var olmayan bir ürün id bilgisi satış veya alım esnasında kullanılmaması gereklidir. Aynı şekilde tanımlanmamış bir cariye satış işlemi veya cariden alım işlemi gerçekleşmemesi gereklidir.

Veritabanında sorgulama yapılrken birden fazla tablo üzerinde işlem yapılacaksa tablo_adi.alan şeklinde ifade edilir.

Aşağıda tablolarımız arasındaki ilişkiler verilmiştir. Unutulmaması gereken bir diğer nokta ilişkilendirilen sütunların veri tipleri aynı tanımlanmalıdır. İlişkilerde belirtilen n ve 1 değerleri ilişkinin türünü göstermektedir.

Örneğin, urun.urun_id $\leftarrow\text{-1-n}\rightarrow$ satis.urun_id ifadesi "Aynı id değerine sahip bir tek ürün olabilir ama bir satışta birden fazla ürün türü olabilir." anlamına gelmektedir.

Aşağıdaki tabloda tablolar arasındaki ilişkiler gösterilmiştir.

Tablolar Arasındaki İlişkiler		
satis.cari_id	$\leftarrow\text{-n-1}\rightarrow$	cari.cari_id
alim.cari_id	$\leftarrow\text{-n-1}\rightarrow$	cari.cari_id
satis.urun_id	$\leftarrow\text{-n-1}\rightarrow$	urun.urun_id
alim.urun_id	$\leftarrow\text{-n-1}\rightarrow$	urun.urun_id

Örnek 2:

Hazırlayacağımız veritabanı basit bir öğrenci bilgi sistemine aittir. Veritabanı içerisinde öğrenci, öğretim elemanı, ders, bölüm ve öğrenci not bilgileri tutulacaktır.

- İlk olarak veritabanının kavramsal modelini çkartalım.