VERİTABANI TASARIMI

Microsoft temelli yazılım geliştirme süreci (MSF) veritabanı tasarımını üç aşamada ele almayı uygun görür.

Süreç	İşlem	Sonuç
Kavramsal Model	**Niçin** temelli düşünün. Use-Case diyagramları ve kullanıcı gereksinim listeleri size yol gösterecektir.	Müşterinin ne istediğini onun gözünden düşünün.
Mantıksal Model	**Ne** temelli düşünün. Müşteri isteklerinden, yazılımcıların ve proje çalışanlarının anlayacağı bir modele geçiş yapın. Veritabanı tasarımı ile ilgili bilimsel kuralları dikkate alın	Teknik çalışanlar için bilimsel temelli ele alınan ve yeniden düzenleme yapılmış ara proje
Fiziksel Model	**Nasıl** temelli düşünün. Projeyi hayata geçirmek için yapacaklarınızı sıralayın. Örneğin, Normalizasyon sizi kısıtlıyorsa, mantıksal şemadan farklı bir fiziksel şema kullanabilirsiniz.	Gerçek anlamda veritabanının nasıl olacağı ile ilgili sonuç projesi
Gerçekleme	Sonuç projesinde yapılanları uygulayın	Projenin veritabanı seviyesinin tamamlanması

Tablo 1. MSF'ye göre üç aşamalı veritabanı tasarımı

Kavramsal Modelde, projenin uygulanacağı sahadaki kişilerle birebir temasa geçip analizler ve gereksinim listeleri yapmak gerekir. Bu seviyedeki bütün model, veritabanından anlamayan, sadece işine odaklanmış şekilde düşünen işverenin bakış açısıyla ifade edilir.

Arkasından elde edilen bu modelin, yazılımcıların bakış açısına ve bilimsel kurallara (normalizasyon, veri bütünlüğü vs.) göre yeniden düzenlenerek **mantıksal** bir proje ortaya koyulması gerekir.

Genellikle bir mantıksal proje, fiziksel proje olarak da kullanılabilir. Ancak **fiziksel** seviyede sistemin çalışacağı ortama dair gerçekleri göz önüne alıp, projeyi yeniden elden geçirmek gerekir. Genellikle ilk elde edilen fiziksel plan performans ve güvenlik gibi nedenlerle yeniden gözden geçirilerek en son hali verilir.

Ön Araştırma ve Gereksinim Analizi Safhası

> "Bir gereksinimi ilk yerde yakalamak, inşa anında veya daha sonra tamir etmekten 50 ile 200 defa daha ucuzdur" (Boehm ve Papaccio, 1988)

Bu safhalara başlamadan önce kapsamlı bir saha analizi yapmanız gerekir. Örneğin ürünlerin internetten satışı ile ilgili bir projeye başlayacaksanız, ilgili şirketin satış bölümünden üretim bölümüne kadar birçok yer ile görüşmeniz gerekecektir. Mümkün olduğunca farklı görevde çalışanlardan gereksinim dinleyin. İşleyiş ile ilgili standart form ve belgeler varsa bunları inceleyin. Eski sürüm bir yazılım varsa bulunan ekranları ve kullanıcıya sağladığı hizmetleri gözden geçirin. Daha iyi nelerin olabileceği hakkında kullanıcıdan fikir almak iyi bir seçim olabilir.

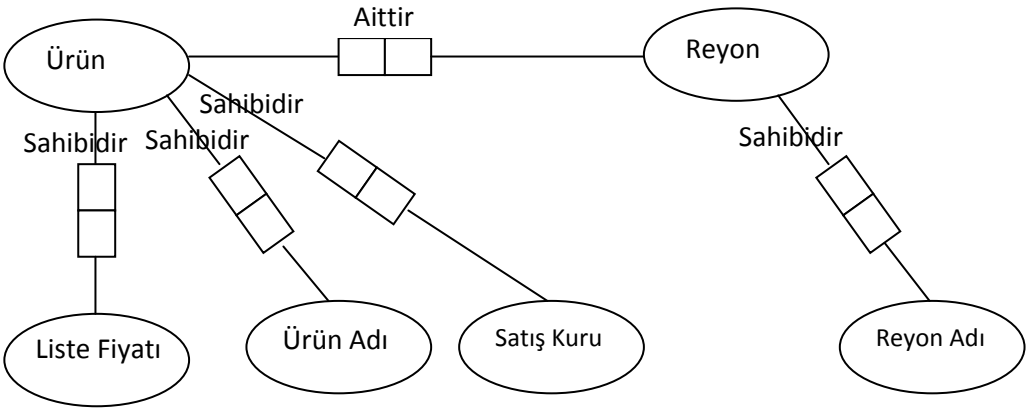
1. Kavramsal Model

Kavramsal (Consept) seviyede model oluşturulurken, kullanıcıların (ve müşterilerin) gereksinimlerini ifade eden, veri tabanına nazaran çok üst seviyeli bir model ortaya koymak gerekir. Model, saha analizinde elde edilecek gereksinimleri karşılayacak yeterlilikte olmalıdır. Eski sürüm bir veritabanı varsa, bu aşamaya başlamadan önce mutlaka gözden geçirin. Uygulamayı devreye almadan hemen önce bir veri dönüşümü durumunda kalacağınızı unutmayın. Ayrıca sahada unutulmuş bazı şeyler, bu şekilde kavram modeline yansıtılabilir.

Veritabanı tasarlarken kullanılabilecek birçok yöntem mevcuttur. Burada anlatılan yöntemler orta ve büyük ölçekteki veritabanı tasarımları için bilimsel yöntemleri içermektedir.

1.1. ORM ile Modelleme

ORM (Object-Role Modelling) kavramsal seviyede bir veritabanı modelleme yöntemidir. Özellikle Microsoft Visio ile veritabanı tasarlayacaksanız bu teknik işinize daha çok yarayabilir.



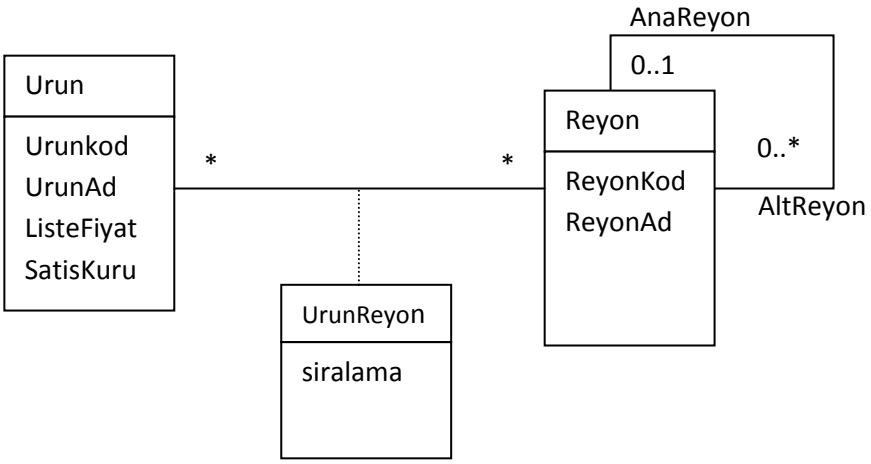
Şekil 1. Urun ve Reyon varlıklarının ORM gösterimi

1.2. UML ile Modelleme

Unified Modeling Language (UML) nesne temelli tasarım, tanımlama ve nesneye dayalı sistemleri ile konuşmayı sağlayan bir gösterim-modelleme dilidir. UML ile bir sistem analizinin çeşitli süreçleri çeşitli diyagramlar kullanarak ifade edilebilir.

- Use-Case Diyagramları
  - Etkileşim Diyagramları: Yazılımı kullanacak her kişinin sistemdeki rolünü ve ilişkilerini ifade etmek için bu diyagram tercih edilebilir. Sistemin, kullanıcılar ve diğer sistemler ile etkileşim noktasında nasıl davranacağını daha net görmemizi sağlar.
  - Etkileşim Hikâyeleri: Kullanıcılar arasındaki iş gereksinimlerine paralel diyaloglar hikâye haline getirilir. Hangi durumda hangi mesaj verilecek, hangi ekranda hangi bilgiler beklenecek gibi…

- Class Diyagram
  - Özet Desenler Oluşturulur: Özet desenler, etkileşim hikâyelerini kullanarak oluşturulabilir. Sınıflara karar verilirken, kullanıcı ve müşteri perspektifinden bakılır.
  - Nesne kısıtlarını ve iş kuralları tayin edilir
- Aktivite Diyagramları
  - Aktivite diyagramları, uygulamada yer alacak nesneler arasındaki etkileşimi ifade etmek için kullanılır.



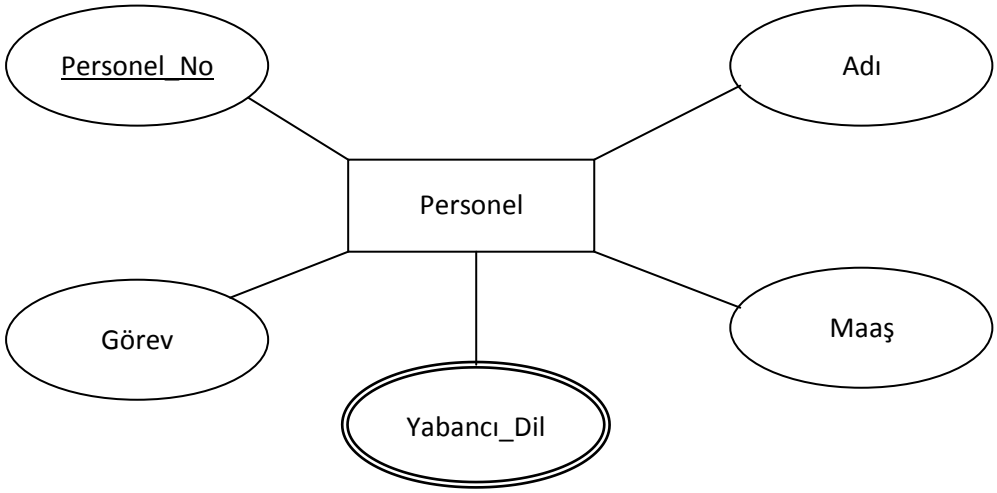
Şekil 2. Ürün ve Reyon bilgilerinin kavramsal olarak UML Class diyagramı tasarımı

1.3. ER ile Modelleme

ER (Entity Relationship) modelleme, ilişkisel veritabanı teoreminin ilk yıllarından bu yana kullanılan ve diğer tüm yöntemlere temel teşkil etmiş bir yöntemdir. Bu yönteme göre üç farklı öğe mevcuttur.

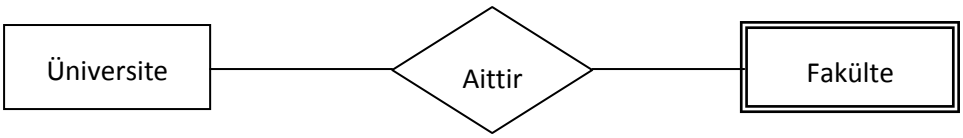
- **Varlık (Entity):** Modelin en temel öğesidir. Var olan ve benzerlerinden ayırt edilebilen her şey varlıktır. Örneğin; kitap, öğrenci, araba birer varlıktır. Birden fazla varlığın oluşturduğu kümeye **varlık kümesi** denir. Model içerisinde dikdörtgen ile gösterilir ve içerisine varlığın ismi yazılır. Veritabanı olarak düşünülürse her tablo bir varlık kümesidir.
- **Nitelik (Attribute):** Varlığın her bir özelliği bir nitelik olarak ifade edilir. Örneğin; öğrenci numarası ve bölümü öğrenci varlığının nitelikleridir. Model içerisinde nitelikler oval ile gösterilir ve içerisine nitelik ismi yazılır. Nitelik bağlı olduğu varlığa düz çizgi ile birleştirilir. Veritabanı olarak düşünülürse tablonun her bir sütunu bir niteliği gösterir. Bir niteliğin alt nitelikleri olabilir.
- **İlişki:** Farklı varlıklar arasındaki ilişkileri ifade eder. Örneğin öğrenci ve dersler ayrı varlık kümeleridir ama öğrenciler ders almak zorunda oldukları için iki varlık arasında bir ilişik vardır. Model içerisinde ilişkiler baklava dilimi ile gösterilir ve içerisine ilişki ismi yazılır. Baklava dilimi ilişkili olduğu varlıklarla düz çizgi ile bağlanır. İki varlık arasında birden fazla farklı ilişki olabilir.

- **Anahtar Nitelik:** Bir varlığın değeri her bir varlık için farklıysa bu nitelik **anahtar nitelik** olarak belirlenir. Şema içerisinde niteliğin altı çizilerek belirtilir.
- **Çok Değerli Nitelik:** Bazı durumlarda bir nitelik birden fazla değer içerebilir. Örneğin bir personel birden fazla yabancı dil bilebilir. Bu durumda yabancı dil niteliği **çok değerli niteliktir.** Çok değerli nitelikler çift çizgili oval ile gösterilirler.



Şekil 3. Personel varlığı ve bu varlığa ait olan bazı nitelikler

- **Domain (Etki Alanı) :** Niteliklerin alabileceği değer aralığıdır. Örneğin öğrenci notlarını içeren sınav niteliği için alacağı değerleri 0 ile 100 arasında belirlemek etki alanı oluşturmaktadır. Etik alanı ER şeması üzerinde gösterilmez.
- **Tanımlayıcı Nitelik :** Varlık kümeleri arasında oluşturulan ilişkilerde, ilişki sonucu nitelikler oluşabilir. Bu tür nitelikler tanımlayıcı nitelik olarak adlandırılır. ve oval olarak gösterilip ilişkiye düz bir çizgi ile bağlanır.
- **Zayıf Varlık Kümeleri :** Bir varlık kümesi anahtar niteliğe sahip değilse **zayıf varlık kümesi** olarak adlandırılır. Zayıf varlık kümeleri güçlü varlık kümeleri ile ilişkilendirilerek kullanılır. Zayıf varlık kümeleri güçlü varlık kümelerine var olma bağımlılığı vardır. Zayıf varlık kümeleri çift çizgili dörtgen ile gösterilir.



Şekil 4. Üniversite varlığı ve Fakülte zayıf varlığı

1.3.1. ER ile Modelleme Aşamaları

a. **Gereksinimleri göz önüne alarak varlıkları, nitelikleri ve ilişkileri tayin edin.**
b. **Bağlılıkları ve kısıtlayıcıları tayin edin.**
   Şu şekilde bir gereksinim cümlesinin ön analizde yazıldığını varsayalım.
   "Her bir reyon altında birçok ürün yer alabilir"
   Burada ürün ve reyon varlıkları arasında bir bağlantıdan söz edilebilir. Bu bağlantılar ER diyagramındaki varlıklar arasındaki ilişkilerin tayininde önemlidir.
   Aşağıdaki iki gereksinim ifadesi birbirinden farklı iki ilişki ortaya çıkaracaktır.
   "Her bir ürün birden fazla reyona dâhil olabilir"
   - bir önceki gereksinim ile birlikte düşünüldüğünde, hem ürün hem de reyon tarafından birden fazla bağlantı olacağını ifade eder.
   "Her bir ürün sadece bir reyona dâhil olabilir"
   - Her reyonda birden fazla ürün yer alırken, bir ürün sadece bir reyonda yer alıyorsa reyon tarafından birden fazla bağlantı olacağını ifade eder.
   Bu aşamada diğer kısıtlayıcılar (contraints) da netleştirilmelidir. Mesela her bir varlık için anahtar nitelik bulunmaya çalışılmalıdır. Bunun dışında kısıtlayıcılar varsa bulunmalıdır. Örneğin barkodlar da tekil olmalıdır. Stok adetleri sıfırdan az olmamalıdır. Her bir nitelik için etki alanı (domain) tespit etmek de bir sonraki süreci kolaylaştıracaktır.
c. **Genel tipleri ve özel tipleri tayin edin.**
   Bazen ele alınan veritabanı sistemi genel ve özel varlıklar barındırıyor olabilir. Örneğin çalışanlarımız, müşterilerimiz, tedarikçilerimiz için bir bağlantı defteri oluşturduğumuzu düşünelim.
   Bütün bu bağlantı bilgilerinde ad, soyad, TC Kimlik No nitelikleri ortak olacaktır. Ama bunun dışında çalışanlar için maaş bilgisi, müşteriler için toplam puan, tedarikçiler için bize sağladıkları vade gibi ek bilgiler olduğunu varsayalım. Bu bilgileri genel ve özel bilgiler diye ayırıp genel bilgilerin yer aldığı bir **kişi** varlığı ve kişi varlığına bağlı, özelleştirilmiş çalışan, müşteri, tedarikçi varlıkları oluşturabiliriz. Burada kişi genel tip, diğer üç varlık ise özel tip olarak düşünülebilir.
d. **Kavramsal bir ER diyagramı oluşturun.**

2.Mantıksal Model

Kavramsal seviyede olduğu gibi, mantıksal seviye veritabanı tasarımı için de tercih edilebilecek birçok yöntem vardır. Burada ER diyagramını ele alacağız.

Tasarlanacak veritabanı büyükse, bir tasarım destek yazılımı (MS Visio, IBM Rational Rose, Eclipse gibi) kullanmanız yerinde olacaktır. Basit ve orta seviyeli bir sistem için kağıt-kalem yeterli olacaktır. Mantıksal model geliştirirken şunlara dikkat etmek faydalı olacaktır;

- Başlangıç ER diyagramı oluşturun.
- Platformdan ve fiziksel gerçeklemeden bağımsız veri tipleri kullanın
- Doğruluk açısından planı gözden geçirin

2.1. Veri Normalizasyonu

İlişkisel veri tabanı yaklaşımını ilişkisel yapan asıl unsur, verilerin tablolara parçalanarak saklanmasıdır. Tabloların kaç tane olacağını ve birbiri ile nasıl ilişkilendirileceğine karar verirken normal formları kullanarak mantıksal model elde ederiz.

Veriler normalize edilmezse şu sıkıntılar doğabilir;

- **Tekrarlı Depolama:** Bazı bilgiler birden fazla tekrarlar.
- **Güncelleme Anomalileri:** Bir bilgiyi güncellemek için birden fazla yerde bulma ve değiştirme yapmak gerekir.
- **Ekleme Anomalileri:** Bir veriyi eklemek için birden fazla tabloya veri eklemek gerekir.
- **Silme Anomalileri:** Bir veriyi silmek için birden fazla tabloda bulup silmek gerekir.

2.1.1 Normalizasyonu Kuralları

Veri Tabanı normalleştirilmesi için birkaç kural bulunmaktadır. Her kurala "normal form" adı verilir. İlk üç kural birçok problemin çözümü için yeterlidir. Ama daha üst seviyede olan dört ve beşinci normalizasyon kuralları da vardır.

1. Normal Form (1NF)

Birinci normal form a uygun bir varlık şu şartları sağlamalıdır;

- Varlığın her bir niteliği atomik olmalıdır. Birden fazla türdeki bilgi tek bir sütunda olamaz.
- Bilgiler tekrarlayan gruplardan oluşmamalıdır. Örneğin bir alan içerisindeki bilgi özel karakterler ile ayrılarak tutulmamalıdır.

KullaniciKod	AdSoyad
1	Ali AK
2	Ahmet PAK
3	Yahya TAK

Tablo 2. Ad ve Soyad aynı sütunda

KullaniciKod	Ad	Soyad
1	Ali	AK
2	Ahmet	PAK
3	Yahya	TAK

Tablo 3. Tablo 2'de verilen tablonun normalize edilmiş hali, Ad ve Soyad farklı sütunlarda

Ogr_No	BolumKod	BolumAd	Ders_Kodu	Sinav
2012688001	1	Bilgisayar	BPR201, BPR202, BPR203	75, 85, 45
2012690005	2	Muhasebe	MVU202, MVU204, MVU206	25, 60, 55

Tablo 4. Ders_Kodu ve Sinav sütunları birden fazla değer içermektedir.

Ogr_No	BolumKod	BolumAd	Ders_Kodu	Sinav
2012688001	1	Bilgisayar	BPR201	75
2012688001	1	Bilgisayar	BPR202	85
2012688001	1	Bilgisayar	BPR203	45
2012690005	2	Muhasebe	MVU202	25
2012690005	2	Muhasebe	MVU204	60
2012690005	2	Muhasebe	MVU206	55

Tablo 5. Tablo 4'te verilen tablonun normalize edilmiş hali

2. Normal Form (2NF)

İkinci normal form a uygun bir varlık şu şartları sağlamalıdır;

- Birinci normal forma uygun olmalıdır.
- Bir tablo içerisindeki tüm sütunlar, birincil anahtar olarak tanımlı sütuna bağımlı olmalıdır. Anahtar sütunlara bağımlı olmayan bilgi varsa ayrı bir tablo oluşturulmalıdır.
- Anahtar sütun birden fazla sütunun birleşiminden oluşuyorsa tabloda yer alacak bilgiler tüm anahtar sütunlara bağımlı olmalıdır. Tek sütuna bağımlı ise ayrı bir tabloda tutulmalıdır.

SatNo	MusNo	MusteriAd	Tutar
2012001	002	Ahmet	5000
2012002	006	Murat	12500
2012003	005	Ayşe	700
2012004	002	Ahmet	1600
2012005	005	Ayşe	3000
2012006	004	Mehmet	6700

Tablo 6. Satis tablosu

Tablo 6'da verilen satis tablosunda SatNo ve MusNo sütunlarının beraber birincil anahtar oluşturduğunu varsayalım. Tablo, 1NF kuralına uymaktadır. Fakat 2NF kuralına uymadığı görülüyor. MusteriAd sütunu tek başına MusNo sütununa bağımlı iken Tutar sütunu ise tek başına SatNo sütununa bağımlıdır. Bu tabloyu ikinci normalizasyon kuralına uydurmak için tablo 7'deki gibi iki ayrı tabloya bölmeliyiz.

SatNo	MusNo	Tutar
2012001	002	5000
2012002	006	12500
2012003	005	700
2012004	002	1600
2012005	005	3000
2012006	004	6700

MusNo	MusteriAd
002	Ahmet
004	Mehmet
005	Ayşe
006	Murat

Tablo 7. Normalizasyon işlemi sonrasındaki Satis ve Müsteri tabloları

3. Normal Form (3NF)

Üçüncü normal form a uygun bir varlık şu şartları sağlamalıdır;

- İkinci normal forma uygun olmalıdır.
- Tablo içerisindeki geçiş bağımlılıkları olmamalıdır. Yani anahtar olmayan sütunlara bağımlı bilgi varsa ayrı bir tablo oluşturulmalıdır.

Örneğin; tablo 5'teki Bolum ve BolumKod sütunları arasında geçişli bağımlılık mevcuttur.

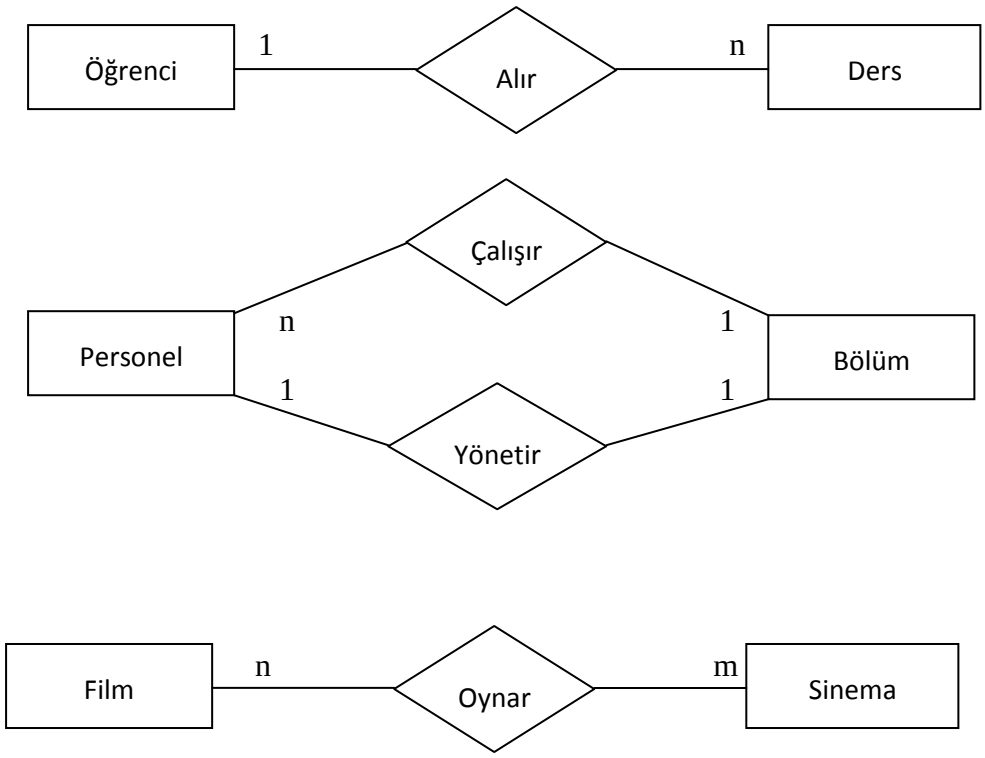
2.1.2 Kayıtların İlişkilendirilmesi

İlişki tipleri: Varlık kümelerinin ilişkilendirilmesi sonucu

1-n (bire çok ilişki)

1-1 (bire bir ilişki)

n-m (çoka çok ilişki) tipinde ilişkiler oluşabilir. Bu ilişkiler için örnekler şekil 5'te gösterilmiştir.



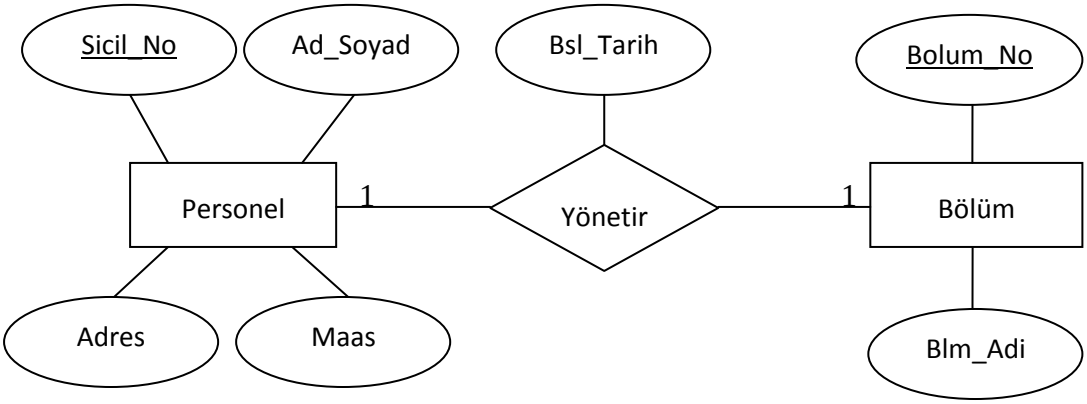
Şekil 5. (1-n), (1-1) ve (n-m) ilişki örnekleri

Varlık-İlişki modellerinin tabloya dökülmesi

a. Bire bir (1-1) ilişkilerin tabloya dönüştürülmesi

- Varlık kümelerinin her biri bir tablo olarak oluşturulur.
- Nitelikler tabloların sütunlarına dönüştürülür.
- İlişkide bir varlık kümesinin birincil anahtarı diğer varlık kümesinin yabancıl anahtarı olarak belirlenir. Hangisinin yabancıl olacağına tablonun içereceği bilgilere göre karar verilir.
- Oluşturulan ilişkide tanımlayıcı nitelik bulunuyorsa, tanımlayıcı nitelikler yabancıl anahtar olarak kullanılan tabloya sütun olarak eklenir.

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

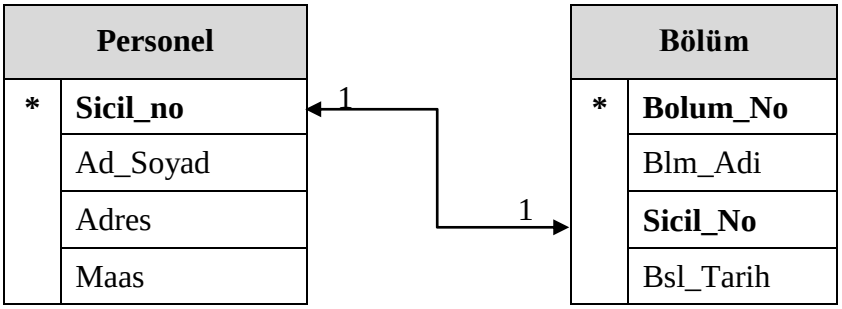


Personel varlık kümesi *personel* ve bölüm varlık kümesi de *bölüm* tablosu olarak isimlendirilir.

Personel tablosunun sütunları *sicil_no, ad_soyad, adres, maas*, bölüm tablosunun sütunları ise *bolum_no* ve *blm_adi* olacaktır.

İlişki 1-1 olduğu için istediğimiz bir tablonun birincil anahtarı diğer tabloda yabancıl anahtar olarak kullanılabilir. Personel tablosunun birincil anahtarını bölüm tablosunda yabancıl anahtar olarak kullanalım.

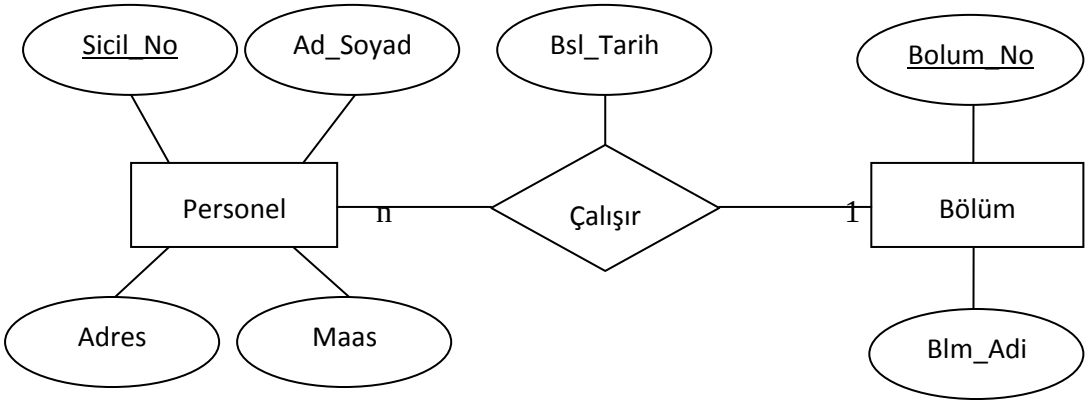
Son olarak yönetir ilişkisine ait tanımlayıcı nitelik, yabancıl anahtar kullandığımız bölüm tablosuna sütun olarak eklenir.



b. Bire çok (1-n) ilişkilerin tabloya dönüştürülmesi

- Varlık kümelerinin her biri bir tablo olarak oluşturulur.
- Nitelikler tabloların sütunlarına dönüştürülür.
- İlişkide n tarafındaki tabloya 1 tarafındaki tablonun birincil anahtarı yabancıl anahtar olarak eklenir.
- Oluşturulan ilişkide tanımlayıcı nitelik bulunuyorsa, tanımlayıcı nitelikler n tarafındaki tabloya sütun olarak eklenir.

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

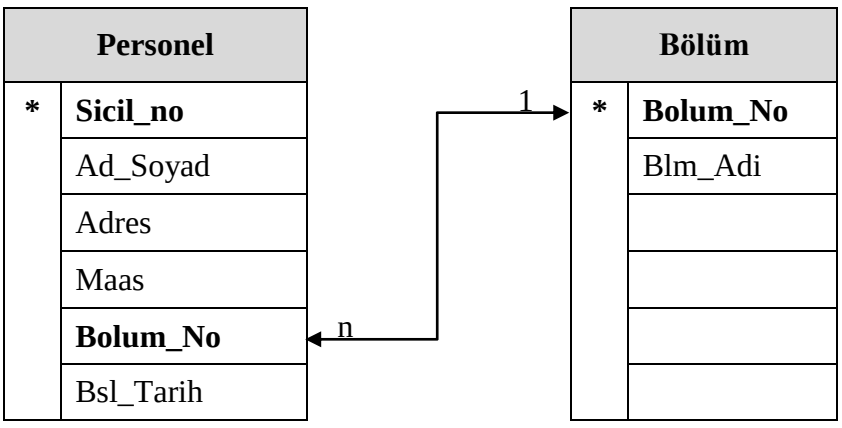


Personel varlık kümesi *personel* ve bölüm varlık kümesi de *bölüm* tablosu olarak isimlendirilir.

Personel tablosunun sütunları *sicil_no, ad_soyad, adres, maas*, bölüm tablosunun sütunları ise *bolum_no* ve *blm_adi* olacaktır.

İlişki 1-n olduğu için 1 tarafındaki bölüm tablosunun birincil anahtarı n tarafındaki personel tablosuna yabancıl anahtar olarak eklenir.

Son olarak çalışır ilişkisine ait tanımlayıcı nitelik, n tarafındaki (yabancıl anahtar kullandığımız) personel tablosuna sütun olarak eklenir.

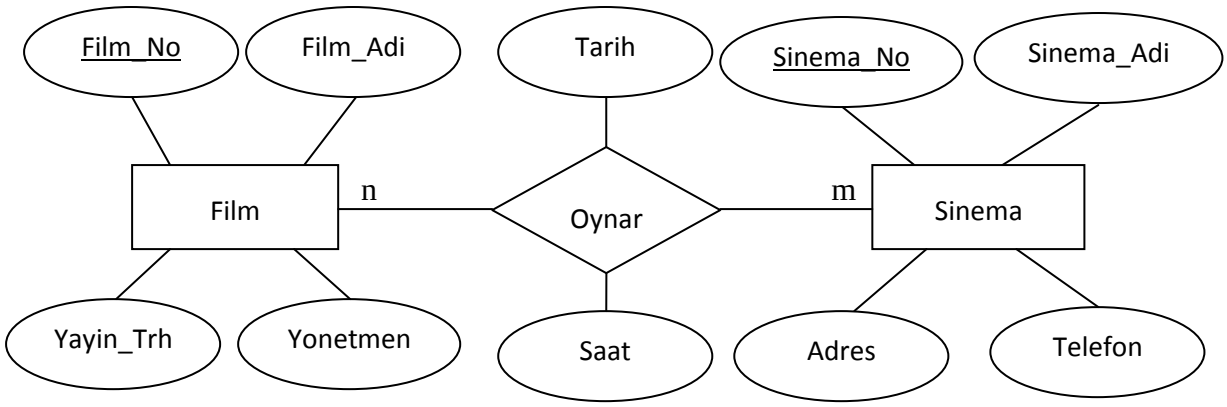


c. Çoka çok (n-m) ilişkilerin tabloya dönüştürülmesi

SQL Server birçok ilişkisel veritabanı yönetim sistemi gibi, çoka çok (n-m) ilişkiyi desteklemez. Bu nedenle iki tabloya (n-1) ve (1-m) ilişkileri ile bağlı üçüncü bir tablo üzerinden ilişkilendirilmesi gerekiyor. Bu üçüncü tabloya geçiş (junction) tablosu adı verilir.

- Varlık kümelerinin her biri bir tablo olarak oluşturulur.
- Aradaki ilişki isminde tablo oluşturulur. (geçiş tablosu)
- Nitelikler tabloların sütunlarına dönüştürülür.
- Oluşturulan ilişkide tanımlayıcı nitelik bulunuyorsa, ilişkiden oluşan tabloya sütun olarak eklenir.
- n ve m tarafındaki tabloların birincil anahtarları ilişkiden oluşan tabloya yabancıl anahtar olarak eklenir.
- İlişkiden oluşan tablonun birincil anahtarı oluşturulan yabancıl anahtarların birleşiminden oluşur. Eğer bu şekilde oluşan birincil anahtar ihtiyaca cevap vermiyorsa yeni bir sütun eklenerek birincil anahtar yapılır.

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

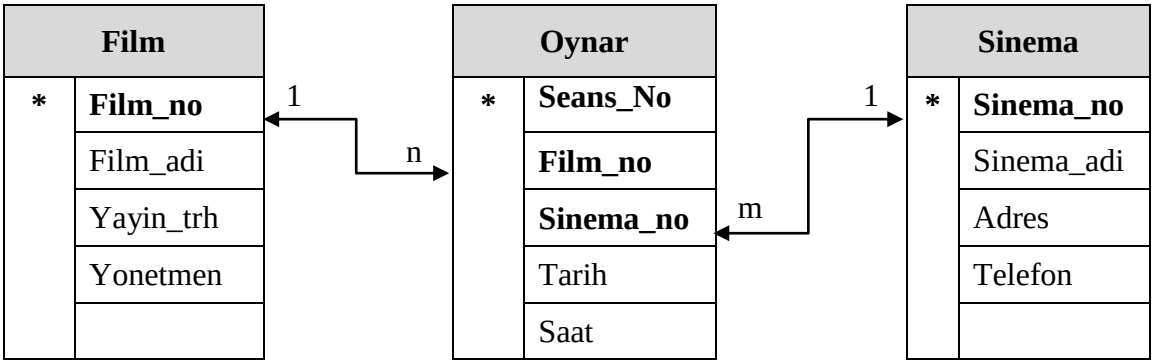


Film varlık kümesi *film,* sinema varlık kümesi *sinema* ve oynar ilişkisi de oynar tablosu olarak isimlendirilir.

Film tablosunun sütunları *film_no, film_adi, yayin_trh, yonetmen* sinema tablosunun sütunları ise *sinema_no, sinema_adi, adres* ve *telefon* olacaktır.

Oynar tablosuna ise *tarih* ve *saat* sütunları eklenir.

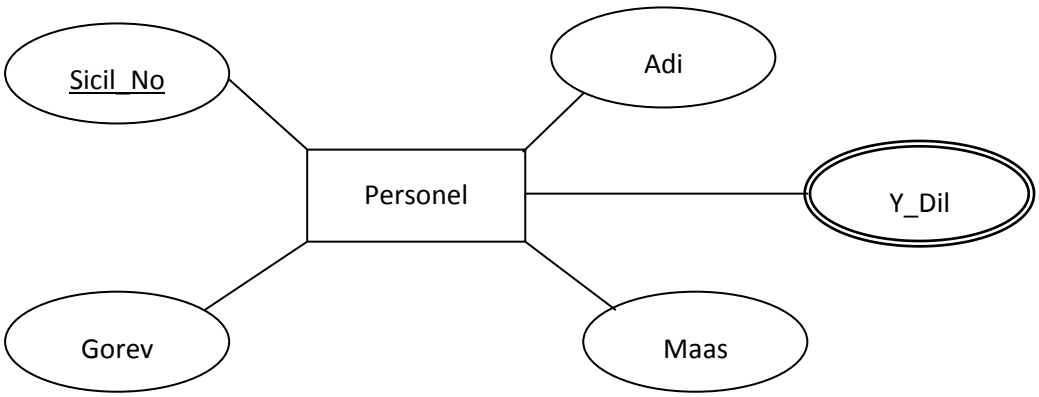
Oynar tablosuna film tablosunun birincil anahtarı *film_no* yabancıl anahtar olarak ve sinema tablosunun birincil anahtarı *sinema_no* yabancıl anahtar olarak eklenir. Oynar tablosunun birincil anahtarı film_no ve sinema_no sütunları olabilir ama bu örnekte *seans_no* isminde yeni bir sütun eklenerek birincil anahtar olarak belirlenmiştir.



d. Çok değerli niteliklerin tabloya dönüştürülmesi

- Varlık kümelerinin her biri bir tablo olarak oluşturulur.
- Nitelikler tabloların sütunlarına dönüştürülür.
- Çok değer içeren her bir değer için tablo oluşturulur. Oluşturulan tabloya çok değerli nitelik sütun olarak eklenir. Bağlı bulunduğu varlık kümesinin birincil anahtarı yabancıl anahtar olarak eklenir.
- Oluşan çok değerli nitelik tablosunun birincil anahtarı, eklenen çok değerli nitelik sütunu ile yabancıl anahtar sütununun birleşimidir.

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



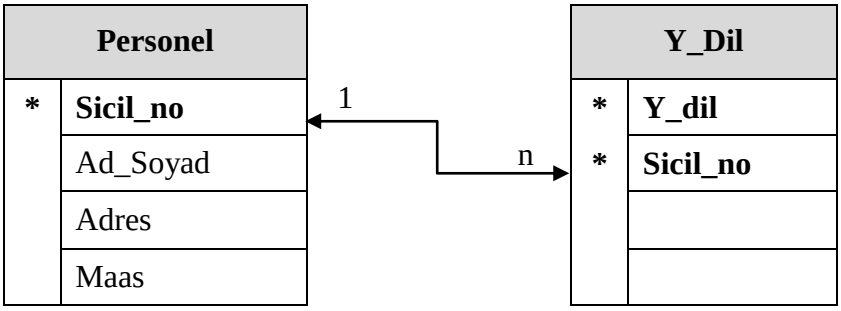
Personel varlık kümesi *personel* ve y_dil çok değerli niteliği de *y_dil* tablosu olarak isimlendirilir.

Personel tablosunun sütunları *sicil_no, ad_soyad, gorev, maas* olacaktır.

Y_dil tablosunun sütunları ise *y_dil* ve *sicil_no* olacaktır.

İlişki 1-n olduğu için 1 tarafındaki bölüm tablosunun birincil anahtarı n tarafındaki personel tablosuna yabancıl anahtar olarak eklenir.

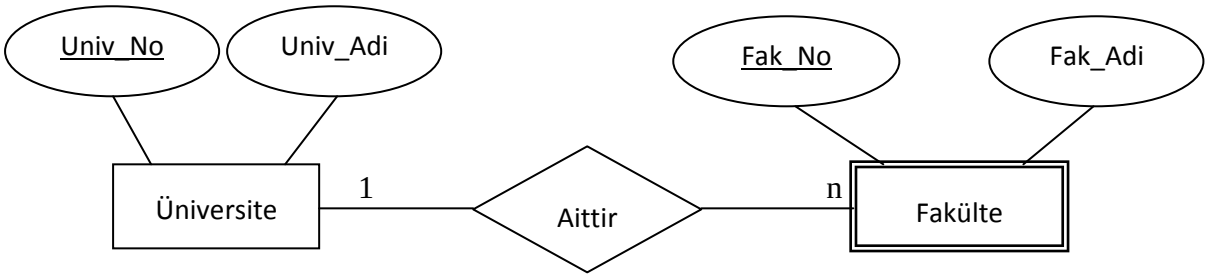
Son olarak çalışır ilişkisine ait tanımlayıcı nitelik, n tarafındaki (yabancıl anahtar kullandığımız) personel tablosuna sütun olarak eklenir.



e. Zayıf varlık kümelerinin tabloya dönüştürülmesi

- Varlık kümelerinin her biri bir tablo olarak oluşturulur.
- Nitelikler tabloların sütunlarına dönüştürülür.
- Zayıf varlık kümesinin olduğu tabloya güçlü varlık kümesinin birincil anahtarı yabancıl anahtar olarak eklenir.
- Zayıf varlık kümesinde her iki tablonun anahtar nitelikleri birleşerek birincil anahtarı oluşturur.
- Oluşturulan ilişkide tanımlayıcı nitelik bulunuyorsa, tanımlayıcı nitelikler n tarafındaki tabloya sütun olarak eklenir.

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

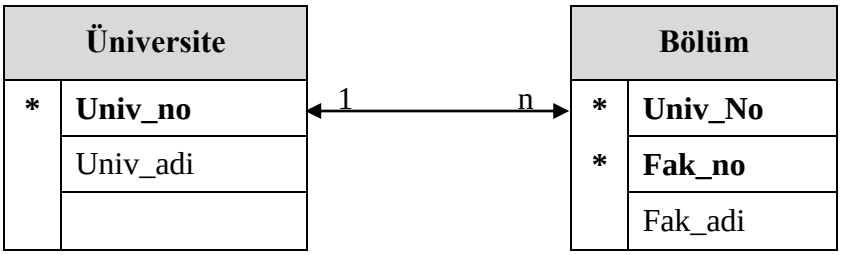


Üniversite varlık kümesi *üniversite* ve fakülte varlık kümesi de *fakülte* tablosu olarak isimlendirilir.

Üniversite tablosunun sütunları *univ_no, univ_adi,* fakülte tablosunun sütunları ise *fak_no* ve *fak_adi* olacaktır.

Üniversite tablosunun birincil anahtarı, zayıf varlık kümesi olan fakülte tablosuna yabancıl anahtar olarak eklenir.

Fakülte tablosunda univ_no ve fak_no sütunları birlikte birincil anahtar olarak belirlenir.



Kaynaklar

Gözüdeli Y., 2008, "Yazılımcılar için SQL Server 2005 ve Veritabanı Programlama", Seçkin Yayınevi
Özseven, T., 2011, "Veri Tabanı Yönetim Sistemleri-1", Murathan Yayınevi