VERİ TABANI YÖNETİMİ

Sunu 2: Veri Modelleri

Öğr. Gör. Selçuk ÖZKAN

Veri Modeli

- Verileri mantıksal düzeyde düzenlemek için kullanılan yapılar, kavramlar ve işlemler topluluğuna veri modeli (data model) denir.
- Her VTYS belirli bir veri modelini kullanır.
- Bir VTYS'yi kullanarak oluşturulacak her veri tabanında yer alacak veriler ve veriler arası ilişkiler, mantıksal düzeyde ilgili veri modeline göre düzenlenir; bu veri modeli kullanılarak veri tabanının kavramsal ve dış şemaları oluşturulur.

Veri Modelleri

- Bugüne kadar geliştirilmiş olan çok sayıda veri modeli vardır.
- Ancak geçmişte ve günümüzde yaygın kullanılan veri modellerini 4 grupta toplamak mümkündür:
  - Sıradüzensel Veri Modeli (Hierarchical Data Model)
  - Ağ Veri Modeli (Network Data Model)
  - İlişkisel Veri Modeli (Relational Data Model)
  - Nesneye-Yönelik Veri Modeli (Object-oriented Data Model)
- Yukarıdaki sıralama aynı zamanda kronolojik bir sıralamadır.

Veri Modelleri : Tarihçe

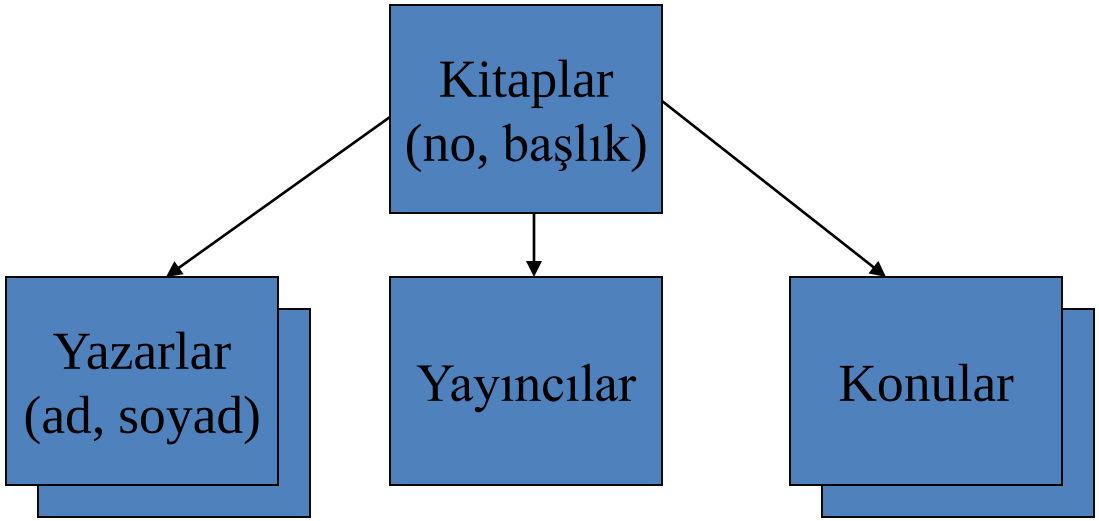
- Sıradüzensel (Hiyerarşik) Veri Modeli en eski model olup 60 ve 70'li yıllarda çok kullanılmıştır.
- 1969’da ortaya çıkan Ağ Veri Modeli 1970’li yıllarda ve 1980’li yılların ilk yarısında kullanılmıştır.
- İlişkisel veri modeli de ilk kez 1969 yılında ortaya atılmış, 1970’li yılların sonunda kullanılmaya başlanmış ve 1985 yılından sonra yaygınlaşmış bir yaklaşımdır.
- 1990’lı yıllarda yaygın kullanılan VTYS'lerin hemen hemen tümünün ilişkisel tabanlı olduğu söylenebilir.

Veri Modelleri : Tarihçe

- Nesneye-yönelik veri modeli yaklaşımı ise on yılı aşkın süredir gündemde olan, günümüzde çok yaygın kullanılmasa bile, kullanımı giderek yaygınlaşan bir yaklaşımdır.
- Geçmişe baktığımızda, ilişkisel yaklaşımın kullanılmaya başlanması ile sıradüzensel ve ağ yaklaşımlarının terk edildiği görülmektedir.
- Buna karşılık nesneye-yönelik yaklaşımın kullanılmaya başlanması ile ilişkisel yaklaşım terk edilmemiştir.
- Günümüzde hem ilişkisel hem de nesneye-yönelik yaklaşımı birlikte kullanan VTYS'lerinin yaygınlaştığı görülmektedir (ORDBMS).

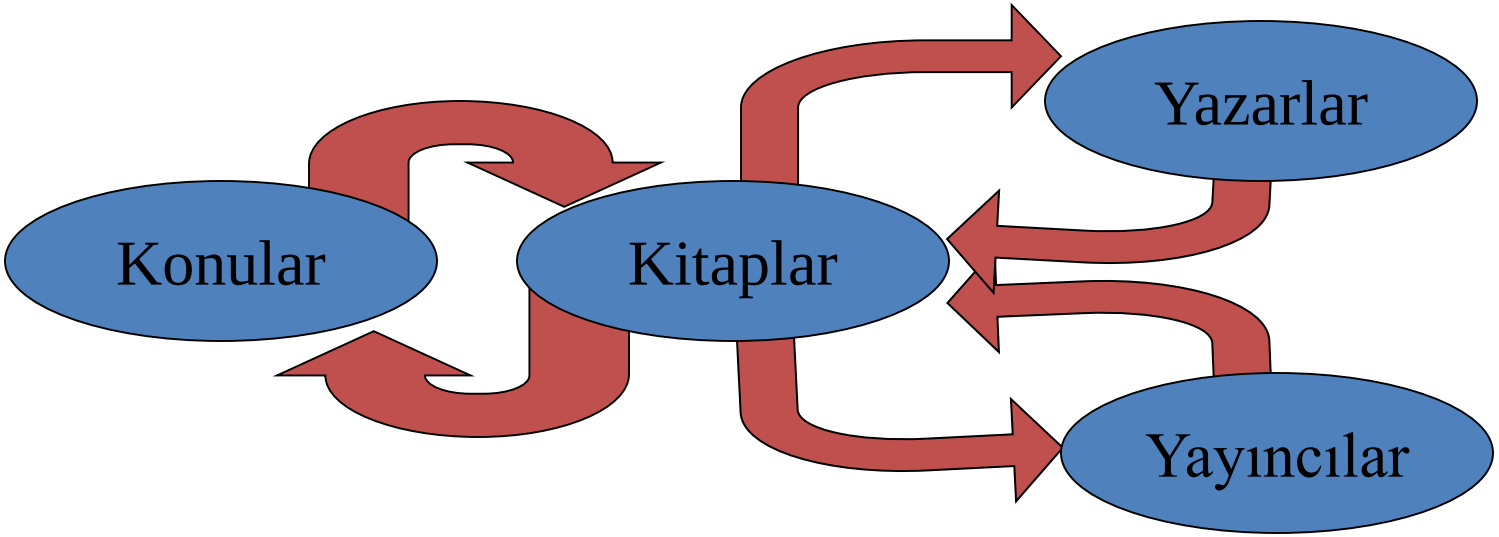
Sıradüzensel (Hiyerarşik) Model

- 1960'lar ve 1970'ler
  - Ağaç veri yapısına benzer. Her kaydın bir ebeveyn kaydı, birçok çocuk kaydı var (IBM IMS: *Information Management System*).



Ağ Modeli

- 1970'ler
  - Her kaydın birçok ebeveyn ve birçok çocuk kaydı bulunabilir. (TurboIMAGE, IDMS, RDM Embedded, RDM Server)



İlişkisel Model

- 1980'ler
  - Veriler için kavramsal olarak basit bir model; veriler ve ilişkiler "tablolar" üzerinde tanımlanır ve tüm bilgiler görülebilecek şekildedir.

Book ID	Title	pubid	Author id
1	Introductio	2	1
2	The history	4	2
3	New stuff a	3	3
4	Another tit	2	4
5	And yet m	1	5

pubid	pubname
1	Harper
2	Addison
3	Oxford
4	Que

Authorid	Author nan
1	Smith
2	Wynar
3	Jones
4	Duncan
5	Applegate

Book ID	Subid
1	2
2	1
3	3
4	2
4	3

Subid	Subject
1	cataloging
2	history
3	stuff

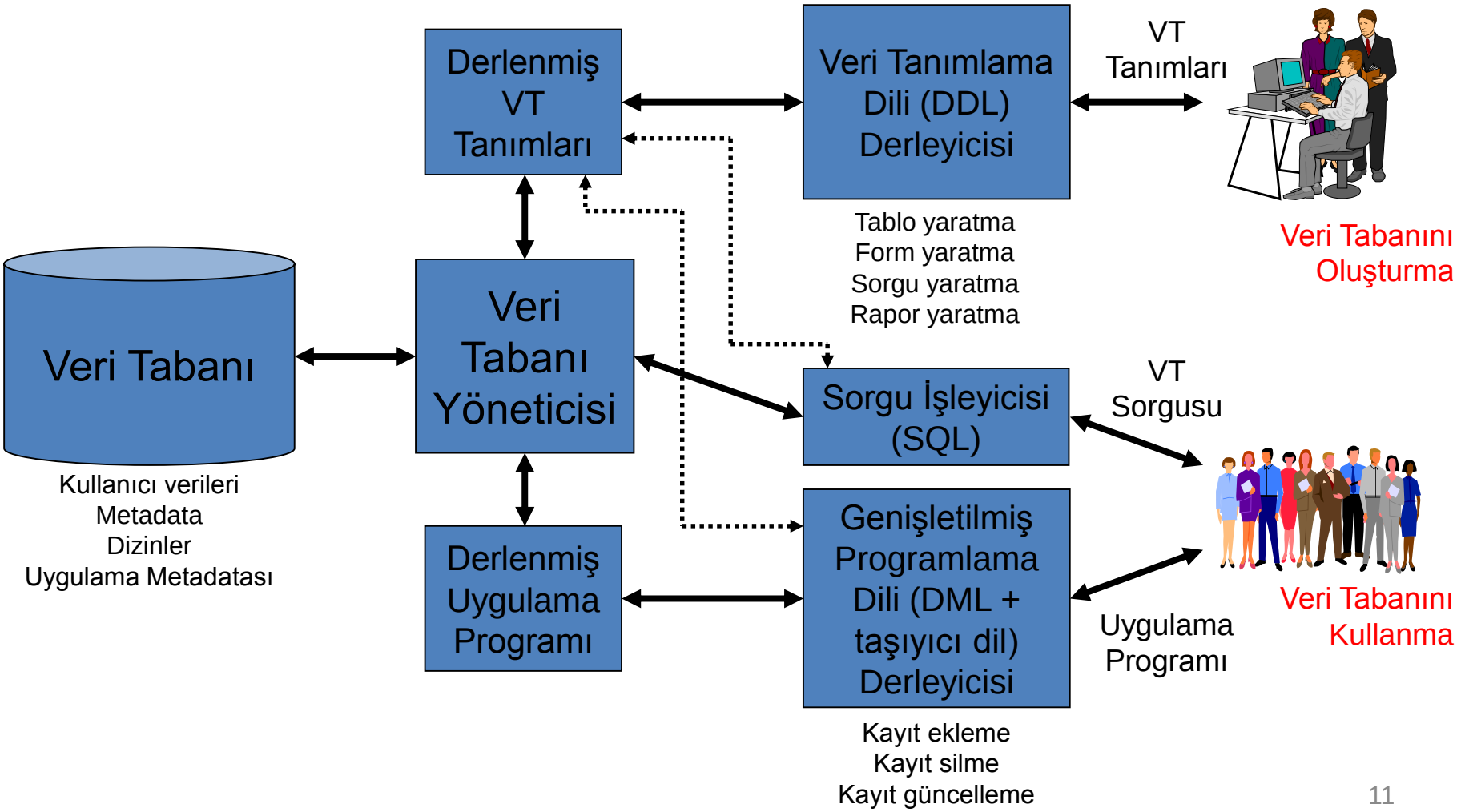
Nesneye-Yönelik Model

- 1990’lar
  - Veriler **nesne** olarak modellenir ve yaratılır.
  - Nesneye-Yönelik Programlama (OOP) da olan sınıf ve miras kavramlarına sahiptir.
  - Karmaşık veriler üzerinde işlem yaparken yüksek performans sunan bir yaklaşımdır.

Nesneye-Yönelik İlişkisel Model

- 1990'lar
  - İlişkisel Modelin iyi bilinen özelliklerini Nesne Tabanlı özelliklerle birleştirir. Bunlar:
    - Kullanıcı tanımlı veri türleri
    - Kullanıcı tanımlı fonksiyonlar
    - Kalıtım ve alt sınıflar

VTYS'nin Temel Bileşenleri (1)



Veri Tanımlama Dili (VTD)

- Veri tabanı tanımlarını VTYS'ye iletmek için kullanılan biçimsel dile Veri Tanımlama Dili (DDL: Data Definition Language) adı verilir.
- Veri Tanımlama Dili kullanılarak oluşturulan veri tabanı tanımları, VTYS'nin Veri Tanımlama Dili Derleyicisi olarak adlandırılabilecek bir bileşeni tarafından çözümlenir, varsa eksik ve yanlışları bulunarak kullanıcıya iletilir.
- Yanlışlardan arınmış veri tabanı tanımları VTYS tarafından uygun yapılara dönüştürülerek saklanır.

Veri Tanımlama Dili (VTD)

- Veri tabanı tanımlarının VTYS tarafından derlenerek saklanması veri tabanı yaklaşımının temel özelliklerinden biridir.
- Bu süreç sayesinde, veri tabanı tanımlarının yetkili kişiler tarafından bir kez yapılması, tanımların kalıcılığının sağlanması ve kullanıcıların bu tanımları kullanmaları ve bu tanımlara uygun işlem yapmaları sağlanmış olur.
- Programlama dili kullanılarak gerçekleştirilen dosya tabanlı uygulamalarda, bilgisayar belleklerinde yalnız veri saklanır. Uygulama programlarında, veri üzerinde yapılacak işlemler yanında verinin tanımı da yer alır.

Veri Tanımlama Dili (VTD)

- Veri tabanı yaklaşımında ise veri tanımlama ile veriler üzerindeki uygulama işlemleri birbirinden tamamen ayrılmıştır.
- Veri tanımlama ve daha önce yapılmış tanımları değiştirme yetkisi yalnız Veri Tabanı Sorumlusu (Database Administrator) olarak adlandırılan ve veri tabanının tümünden sorumlu olan kişi ya da kişilere aittir.
- Veri tabanı üzerinde işlem yapan, uygulamaları gerçekleştiren kullanıcıların ise veri tanımlama ya da mevcut tanımları değiştirme yetkisi yoktur.

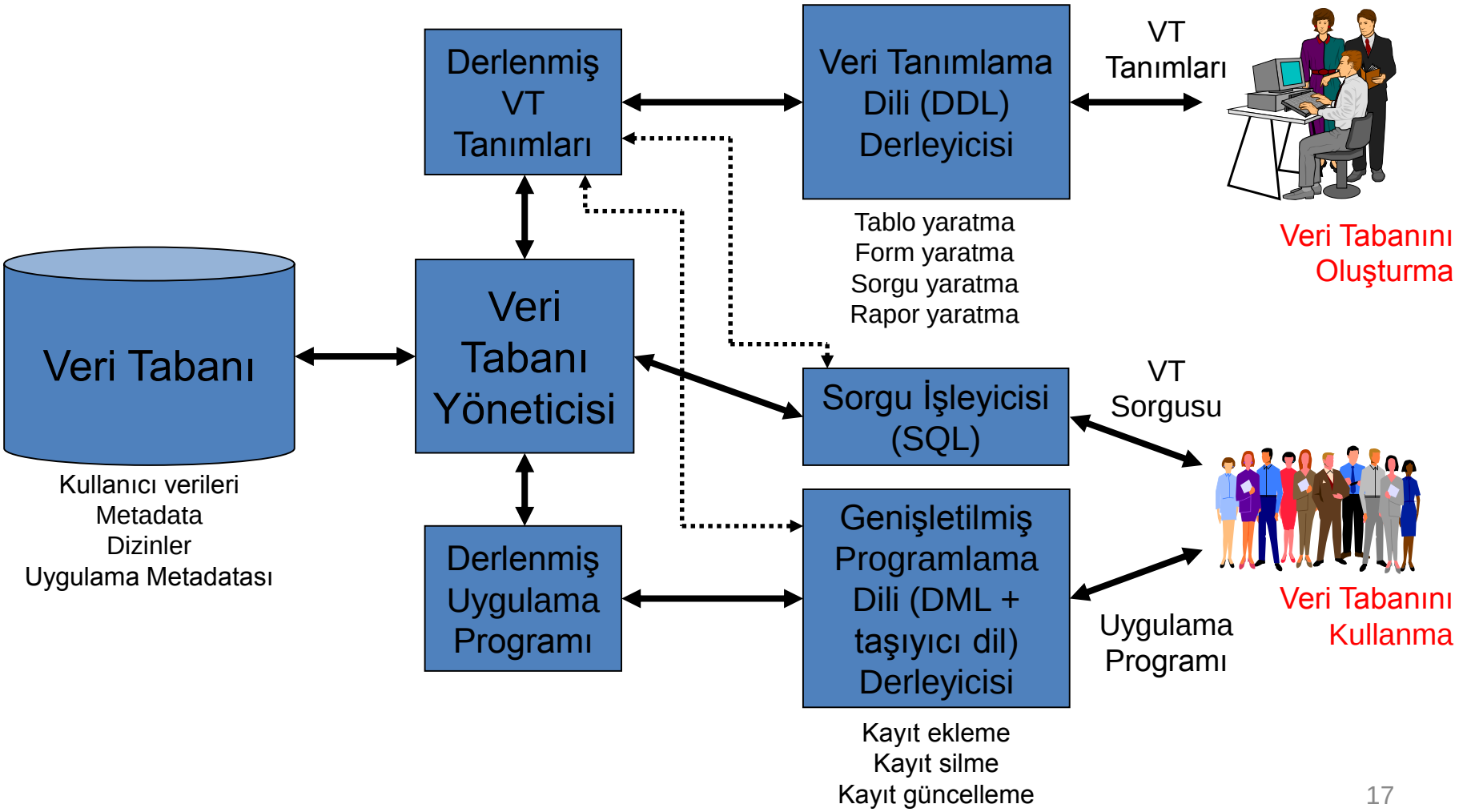
Veri Tanımlama Dili (VTD)

- VTD ile yapılan tanımlarda veri tabanı tanımları içinde yer alan en önemli tanımlar şunlardır:
  1. Mantıksal düzeyde yapılan düzenlemeler oluşturulan yapılar ve her yapıda hangi verilerin yer aldığı.
  2. Her verinin türü, uzunluğu, varsa varsayılan değeri ve diğer özellikleri.
  3. Veriler arası ilişkiler ve her türlü kısıtlamalar.
  4. Fiziksel veri yapıları ile ilgili tercihler ve parametreler.
  5. Kullanıcı tanımları ve kullanıcıların hangi veriler üzerinde hangi işlemleri yapmaya yetkili olduklarına ilişkin tanımlar.

Veri Tanımlama Dili (VTD)

- Veri tabanı tanımları veri sözlüğü (data dictionary) olarak da adlandırılır.
- VTYS'nin fiziksel ortamında aşağıda belirtilen çeşitli veriler saklanır:
  1. veri dosyaları
  2. veri tabanı tanımlarının yer aldığı veri sözlüğü
  3. veri dosyalarına ya da veri sözlüğüne erişim için kullanılan dizinler (indices)
  4. veri değerlerine ve uygulamalara ilişkin istatistiksel veriler ve günlük dosyaları (logfiles)

VTYS'nin Temel Bileşenleri (2)



Sorgu Dili ve Sorgu İşleyicisi

- Veri tabanı uygulamaları için kullanılan en yaygın araç sorgu dilidir.
- Kullanıcı, sorgu dili ile, gerçekleştirmek istediği işlemi yalın bir biçimde ifade eder.
- Kullanıcının oluşturduğu sorguda, neyin yapılmasının istendiği yer alır; bunun nasıl yapılacağı ise yer almaz.

Yapısal Sorgu Dili (SQL)

- IBM, deneysel bir ilişkisel veritabanı yönetim sistemi geliştirmek amacıyla System/R adlı bir proje başlatır.
- Bu sistem için SEQUEL (Structured English Query Language) adında bir sorgu dili geliştirilmeye başlanmıştır.
- 1979’ da tamamlanan bu proje sırasında geliştirilen dilin adı SQL (Structured Query Language) olarak değiştirilmiştir.

ORACLE

- System/R projesini izleyen bir grup mühendis, ilişkisel veritabanındaki potansiyeli görmüş ve Relational Software, Inc. adında bir şirket kurmuşlardır (1979)
- Oracle adını verdikleri ilk ticari ilişkisel veri tabanı yönetim sistemini geliştirmişler ve SQL'i bu sistemde sorgu dili olarak kullanmışlardır.

SQL Örneği

- İngilizce diline çok benzeyen SQL sorgu dilinde oluşturulmuş bir sorgu aşağıda yer almaktadır.

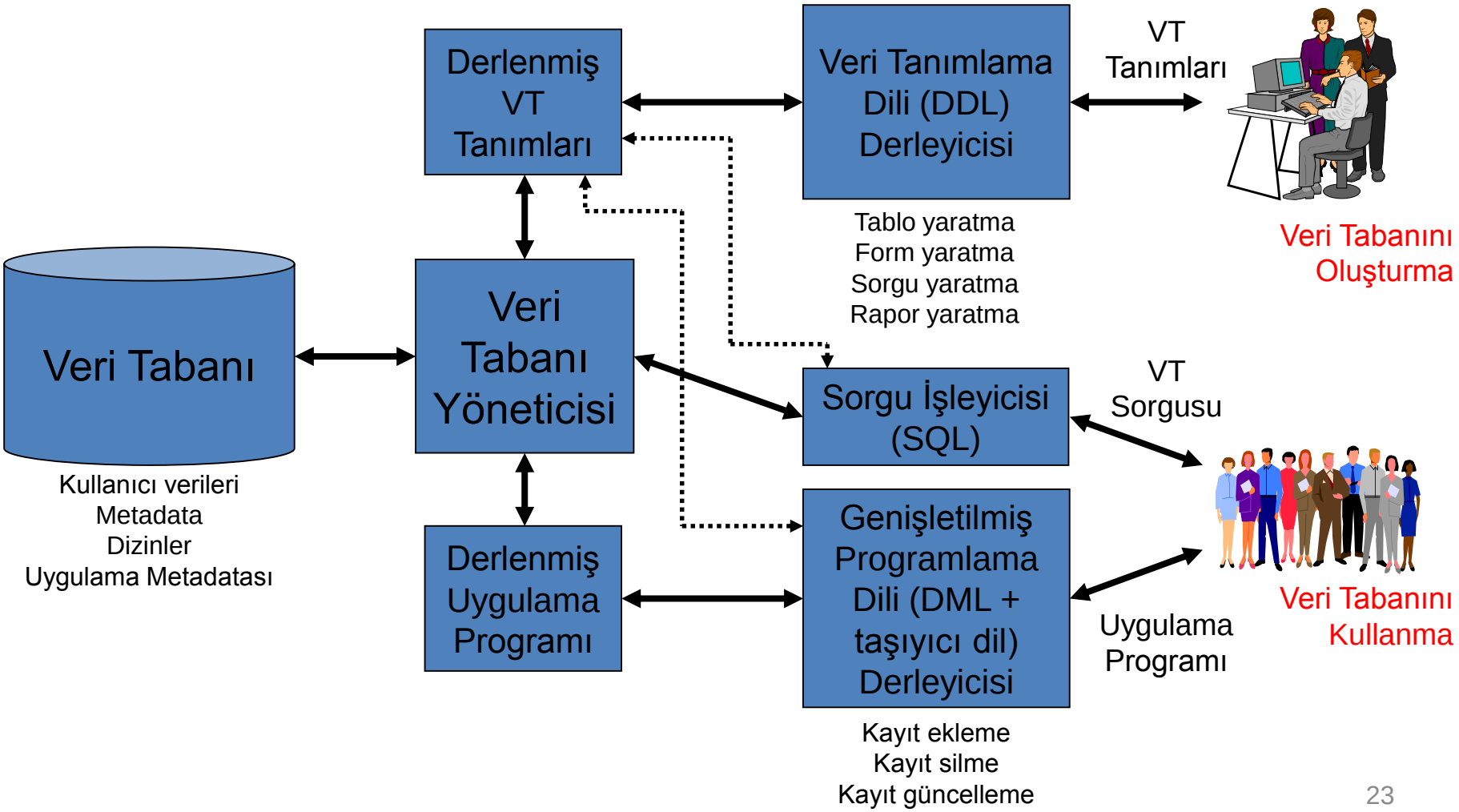
```
SELECT ADI, SOYADI, ADRESİ FROM
PERSONEL WHERE BÖLÜMNO = 17 AND
GÖREVİ = 'Sekreter';
```

- Yukarıdaki sorgu ile "17 numaralı bölümde çalışan sekreterlerin ad, soyad ve adreslerinin" bulunmak istendiği kolayca anlaşılmaktadır.

Sorgu İşleyicisi

- VTYS'nin, sorguların işlenmesi ile ilgili görevleri gerçekleştiren bileşenine Sorgu İşleyici (Query Processor) adı verilir.
- Sorgu işleyicinin görevleri:
  - Sorgunun sözdizimsel ve anlamsal özümlemesini yapmak.
  - Kullanıcının verilen işlemi yapmaya yetkili olup olmadığını denetlemek.
  - Sorguyu işletmek için kullanılabilecek algoritmaları (işletim senaryolarını) belirlemek ve “Query Optimizer” alt bileşeni yardımıyla en iyisini seçmek.
  - Sorgunun işletimini gerçekleştirdikten sonra yanıtını oluşturup kullanıcıya iletmek.

VTYS'nin Temel Bileşenleri (3)



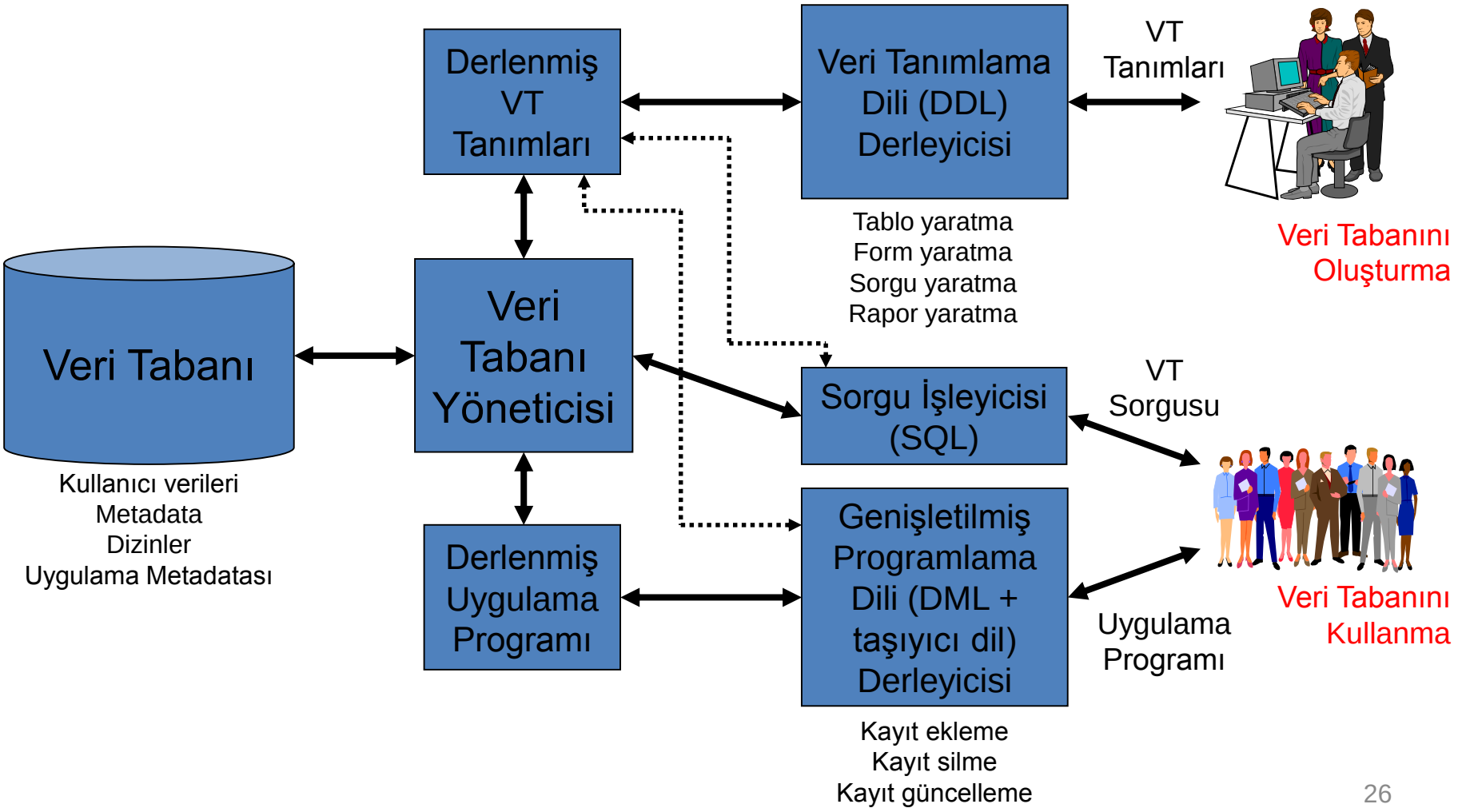
Veri İşleme Dili (DML)

- Veri tabanı üzerinde, veriyi değiştirme, silme ve güncelleme gibi sorgularla ifade edilemeyecek ya da sorgularla ifade edilmesi uygun olmayan işlemler de gerçekleştirilir.
- Bu işlemler için Veri İşleme Dili (DML: Data Manipulation Language) olarak adlandırılan bir dil kullanılır.

Genişletilmiş Programlama Dili

- Veri tabanı üzerindeki uygulamaları gerçekleştirmek için genişletilmiş bir dil oluşturulur.
- Bu dilde hem veri tabanı işlemlerini gerçekleştiren DML komutları, hem de diğer işlemleri gerçekleştiren C, Pascal, Cobol gibi genel amaçlı programlama dillerinden bir ya da birkaçı ile yazılmış taşıyıcı dil komutları yer alır.
- Oluşturulan genişletilmiş dil (DML + taşıyıcı dil) ile hazırlanan uygulama programları, VTYS'nin bileşeni olan genişletilmiş derleyicilerle derlenerek saklanır ve gerektiğinde çalıştırılarak kullanılır.

VTYS'nin Temel Bileşenleri (4)



Veri Tabanı Yöneticisi

- Veri Tabanı Yöneticisi (Database Manager), kullanıcı isteklerini veri tabanı üzerinde gerçekleştiren ve çok kullanıcılı ortamdaki işletimsel sorunları çözümleyen yazılımlar bütünüdür.
- VTYS'nin gerçekleştirdiği işlemlerin önemli bir bölümü bu katman tarafından gerçekleştirilmektedir.

Veri Tabanı Yöneticisi

- Veri Tabanı Yöneticisinin birçok bileşeni vardır. Bunlardan en önemli iki tanesi şunlardır:
- Bellek Yöneticisi (Storage Manager)
  - Dosya Yöneticisi (File Manager)
  - Tampon Yöneticisi (Buffer Manager)
- Hareket Yöneticisi (Transaction Manager)

Bellek Yöneticisi

- Veri Tabanı Yöneticisinin, verilerin ikincil belleklerde saklanması ile ilgili işlevlerini yöneten kesimdir.
- Bir VTYS'nin Bellek Yöneticisi olarak, VTYS'nin altında çalıştığı işletim sisteminin dosya sistemini kullanılabilir.
- Ancak büyük boyutlu veri tabanlarını kurmak ve işletmek için kullanılan VTYS'ler için verimlilik çok önemlidir ve gerek ana bellek, gerekse ikincil bellek yönetiminin VTYS tarafından gerçekleştirilmesinde yarar vardır.

Bellek Yöneticisi

- Bellek Yöneticisi aşağıdaki iki bileşenden oluşur:
- Dosya Yöneticisi (File Manager)
  - Verinin ikincil bellek dosyalarında saklanmasını sağlayan ve ana bellek ile ikincil bellek arasında verinin bloklar halinde aktarılmasını sağlayan yazılımlardır.
- Tampon Yöneticisi (Buffer Manager)
  - Dosya yöneticisi aracığıyla ikincil bellekten getirilen veri bloklarını ana bellek sayfalarında saklayan, ana bellek sayfalarını yöneten ve gerektiğinde ana bellek sayfalarındaki veri bloklarının, dosya yöneticisi aracılığıyla ikincil belleklere yazılmasını sağlayan yazılımdır.

Hareket Yöneticisi

- Veri Tabanı Yöneticisinin, veri tabanı yaklaşımının üstünlüklerinden faydalanmasını sağlayan kesimidir.
- Veri tabanı yaklaşımının üstünlükleri arasında:
  - Çok kullanıcılı ortamda çok çeşitli kullanıcı isteklerinin eşzamanlı gerçekleştirilmesi ve bu arada veri tabanı ve uygulama tutarlılıklarının korunması,
  - Veri tabanı üzerinde yapılan işlemlerin yazılım, donanım ya da güç kaynağı ile ilgili bozukluklar nedeniyle yitirilmemesi

  gibi özellikler bulunmaktadır.

Hareket Yöneticisi

- Hareket Yöneticisinde belirtilen hareket, bir bütün oluşturan ve tutarlılık açısından veri tabanı üzerinde birlikte gerçekleştirilmesi gereken işlemler bütünüdür.
- Tutarlılık açısından, bir hareketi oluşturan işlemlerin ya tümünün gerçekleştirilmesi, ya da hiçbirinin gerçekleştirilmemesi gerekir. Bu Özelliğe hareketin ACID (Atomicity, Consitency, Isolation, Durability) özelliği adı verilir.

Hareket Yöneticisi

- Diğer taraftan, veriler üzerinde değişikliğe neden olan (veri ekleyen, silen ya da verileri güncelleyen) hareketler birlikte işletildiğinde, henüz tamamlanmamış (ve belki de tamamlanmayarak geriye alınacak) bir hareket tarafından gerçekleştirilen değişiklik işlemleriyle oluşturulan veri değerlerinin diğer hareketler tarafından görülmemesi gerekir.

Hareket Yöneticisi

- Hareket yöneticisi, hem hareketlerin ACID özelliğinin bozulmamasını hem de birlikte (eşzamanlı) işletilmelerini sağlar.
- Bunu gerçekleştirmek için de kilitleme (locking), günlük tutma (logging) ve hareket tamamlama (transaction commitment) gibi teknikleri kullanır (daha sonra açıklanacaktır).

Varlık-İlişki Modeli (E-R Modeli)

- Varlık-ilişki modeli, ya da kısaca E-R modeli (Entity-Relationship model) 1976 yılında P.P. Chen tarafından geliştirilen bir modeldir.
- Bugüne kadar varlık-ilişki modeline dayalı hiçbir VTYS geliştirilmemiştir.
- Buna karşılık varlık-ilişki modeli, VTYS'den bağımsız veri çözümlemede ve semantik veri modellemede en çok kullanılan modeldir.

Varlık-İlişki Modeli

- Bu model kullanılarak önce;
  - VTYS'den bağımsız olarak veriler çözümlenir,
  - veri modellemesi yapılır,
  - veriler ve veriler arası ilişkilerin anlamları ve özellikleri incelenerek E-R çizelgeleri oluşturulur;
  - kullanılacak VTYS belirlenir

  sonra da E-R çizelgeleri bu sistemin veri modeline dönüştürülerek veri tabanı şemaları oluşturulur.

Varlık ve Varlık Kümesi

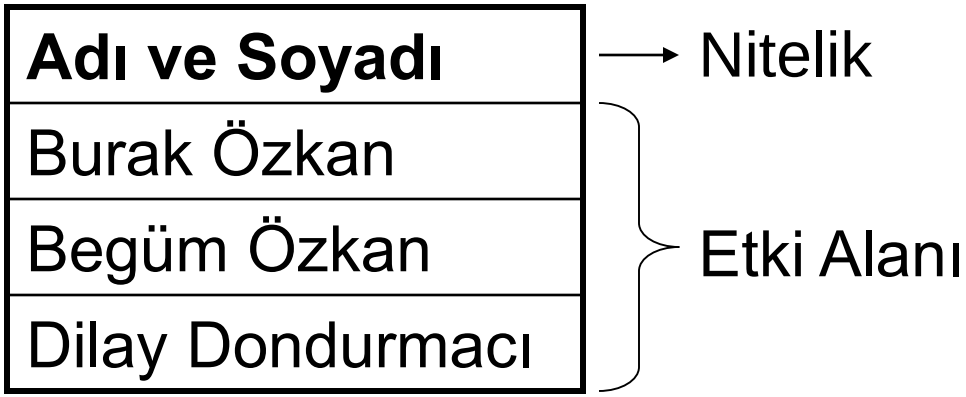
- Var olan ve benzerlerinden ayırt edilebilen her nesneye **varlık** (entity) denir.
  - Bir öğrenci, veri tabanı dersi, belirli bir kitap, Burak birer varlık olarak değerlendirilir.
- Aynı türden benzer varlıkların oluşturduğu kümeye ise **varlık kümesi** (entity set) adı verilir.Varlık kümeleri iç içe, kesişen ya da ayrık kümeler olabilir.
  - Öğrenciler, kız öğrenciler, Bilgisayar Mühendisliği Bölümü öğrencileri, yurtta kalan öğrenciler, renkler, dersler, yıllar, tarihler, satış miktarları,... vb. varlık kümesi örnekleri olarak sayılabilir.

Nitelik

- Bir varlık kümesindeki varlıkların özelliklerini göstermek ve varlıkları birbirinden ayırt etmek için nitelikler (attributes) kullanılır.
- Gerçek dünyada varlık kümelerinin çok sayıda nitelikleri olabilir, ancak veri modellemede, gerçek dünyanın soyut bir modeli oluşturulduğu için, bu niteliklerin yalnız küçük bir kısmı, uygulamalar için gerekli olanları (örn. Sicil no, Ad, Soyad, Adres, … vb) seçilerek kullanılır.

Etki Alanı (Değer Alanı)

- Her niteliğin bir etki alanı (domain) vardır. Etki alanı ilgili niteliğin olabilecek değerlerinin tümünü içeren bir kümedir.



Türetilen Nitelik

- Bir nitelik kullanılarak bir başka varlık niteliği elde edilebiliyorsa, bu yeni niteliğe “türetilen nitelik” adı verilir.
- Örneğin, “personel” varlığının “doğum tarihi” niteliğinden yararlanılarak “yaş” niteliği elde edilebilir. Bu örnekte “yaş” niteliği türetilen niteliktir, tasarımda ayrıca tanımlanmasına gerek yoktur.

Birleşik Nitelik

- Birden fazla nitelik birleştirilerek yeni bir nitelik oluşturulabilir. Bu tür niteliklere birleşik nitelik adı verilir.
- Örneğin, “mahalle”, “cadde”, “sokak”, “apartman”, “posta kodu” ve “şehir” gibi nitelikler birleştirilerek “adres” isimli yeni bir nitelik oluşturulabilir.

İlişki

- Varlıklar arasındaki bağıntıya ilişki adı verilir.
- İkili ilişki:
  - bir öğrenci ile bir ders
  - bir firma ile bir malzeme
- Üçlü ilişki:
  - Bir işçi, bir ürün ve bir makine (işçi bu ürünü üretirken bu makineyi kullandığı için)

İlişki Kümesi

- Aynı türdeki ilişkilerin oluşturduğu kümeye ilişki kümesi denir.
- Matematiksel olarak $E_1, E_2, E_3, \ldots, E_n$ varlık kümeleri arasındaki bir R ilişkisi aşağıdaki gibi tanımlanır:

$$R = \{(e_1, e_2, e_3, \ldots, e_n) :$$

$$e_1 \in E_1, e_2 \in E_2, e_3 \in E_3, \ldots, e_n \in E_n \}$$

İlişki Kümesi : Örnek

- Aşağıdaki iki varlık kümesini göz önüne alalım:

  E_1 = { Ali, Ayşe }

  E_2 = { Matematik, Fizik }

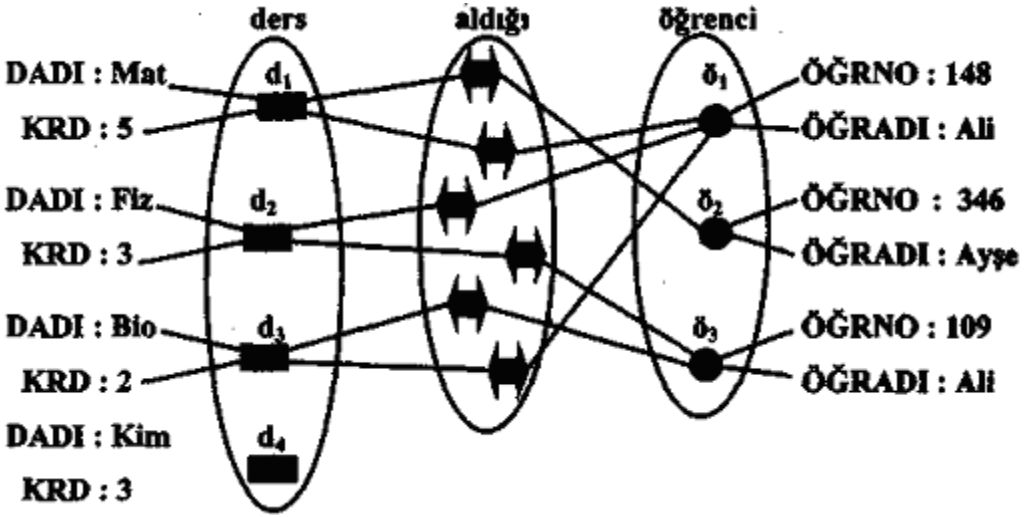
- Bu varlık kümeleri için öğrenci ve aldığı ders ilişkileri aşağıdaki gibi ifade edilebilir:

  R_1 = { (Ali, Matematik) } R_2 = { (Ali, Fizik) }

  R_3 = { (Ayşe, Matematik) } R_4= { (Ayşe, Fizik) }

İlişki Kümesi : Örnek

- Aşağıda şekilde “öğrenci” ve “ders” varlık kümeleri ile bu iki varlık kümesi arasındaki “aldığı” ilişki kümesi görülmektedir.

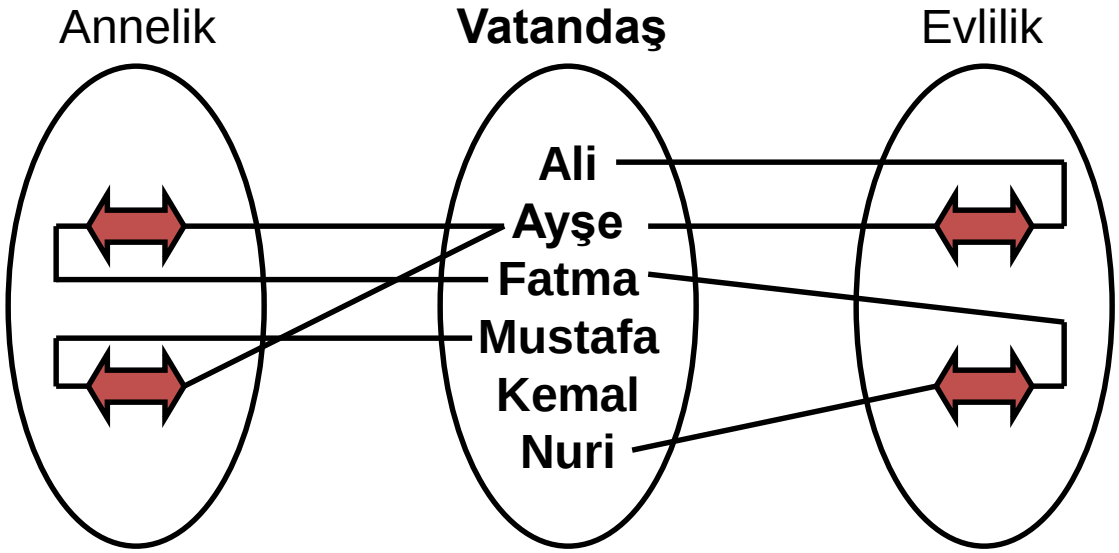


Çoklu İlişki Kümesi

- İlişki kümeleri ikili, üçlü, dörtlü, .. vb. olabilir.
- Ancak genellikle ikili ilişki kümeleri tercih edilir.
- Üçlü, dörtlü, ..vb. ilişki kümesi yerine, istenirse birkaç ikili ilişki kümesi kullanılabilir. Örneğin;
  - Öğrenci, ders ve öğretmen varlık kümeleri arasındaki üçlü ilişki yerine 2 ya da 3 ikili ilişki kullanılabilir.
  - “Ali, Mehmet Hoca’nın anlattığı Fizik dersini alıyor” yerine “Ali Fizik dersini alıyor” ve “Mehmet Hoca Fizik dersini anlatıyor” kullanılabilir.

Aynı Varlık Kümesi ile İlişki (Recursive)

- Aralarında ilişki tanımlanan varlık kümelerinden ikisi ya da daha çoğu aynı varlık kümesi de olabilir.



Rol

- Aralarında ilişki kurulan varlıklardan her birinin ilişkideki işlevine varlığın rolü denir.
- Farklı varlık kümeleri arasındaki ilişkilerde roller dolaylı yoldan anlaşılabildiği için çoğunlukla açıkça belirtilmez
  - Örneğin, öğrenci ve ders arasında kurulan “aldığı” ilişkisinde varlıkların rolleri bellidir: öğrenci dersi alan, ders ise öğrenci tarafından alınandır.

Rol

- Oysa kişi$_1$ ve kişi$_2$ arasında kurulan evlilik ilişkisinde kişilerden hangisinin erkek hangisinin kadın olduğunun belirtilmesi gerekebilir (bazı isimler hem erkek hem kadınlar tarafından kullanıldığı için).
- Benzer biçimde bir kurumda çalışan personel$_1$ ve personel$_2$ arasında kurulan yönetici ilişkisinde hangi personelin rolünün yönetici (üst), hangi personelin rolünün ise yönetilen (ast) olduğunun belirtilmesine gerek vardır.

Rol : Örnek

- İlişkilerdeki roller belirlenirken, başka niteliklere de bakmak gerekebilir.
  - Örneğin “Personel” varlığında “ast-üst” ilişkisini belirlemek için “Görevi” niteliği dışında “Bölümü” niteliğine de bakmak gerekebilir.

Personel varlığı:

Adı	Bölümü	Görevi
Burak	Muhasebe	İşçi
Begüm	Muhasebe	Yönetici
Dilay	Muhasebe	İşçi
Selin	Satış	Yönetici
Sezin	Satış	İşçi

Yönetici (üst, ast) ilişkileri:

(Begüm, Burak)

(Begüm, Dilay)

(Selin, Sezin)

İlişki Kümelerinin Sınırlandırılması

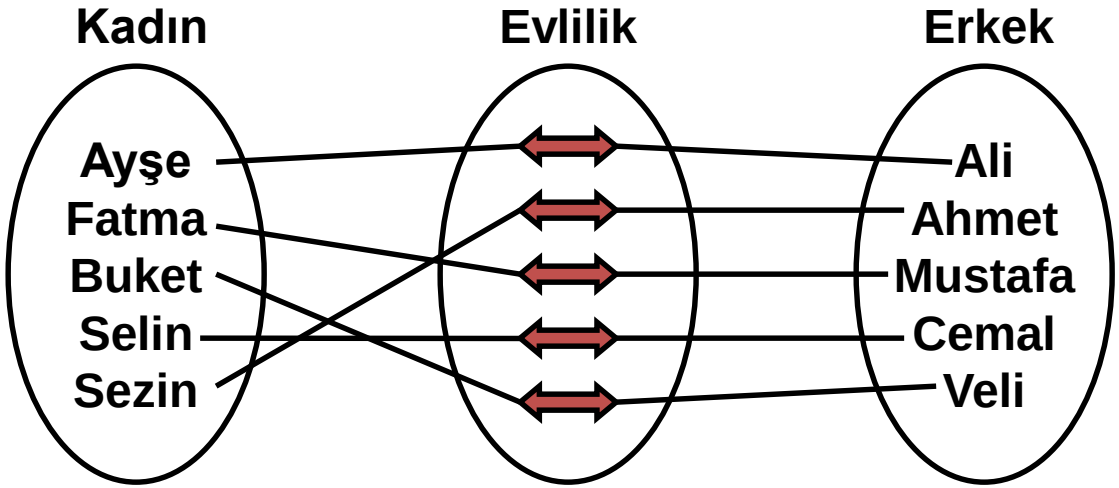
- İlişki kümeleri ile ilgili olarak bir dizi sınırlama tanımlanabilir.
- Bu sınırlamaların en önemlileri, aralarında ilişki kurulan varlık kümeleri arasındaki eşlemelerle ilgili sayısal sınırlamalardır.
- Bu tür sayısal sınırlamalar özellikle ikili ilişki kümeleri için çok önemlidir.

İlişki Türleri

- A ve B varlık kümeleri arasında tanımlanan (A ve B varlık kümeleri aynı da olabilir), A'dan B'ye bir ilişki kümesi, eşleme sınırlamaları açısından aşağıdaki dört türden birinde olabilir.
  - Birden-bire (one-to-one)
  - Birden-çoğa (one-to-many)
  - Çoktan-bire (many-to-one)
  - Çoktan-çoğa (many-to-many)

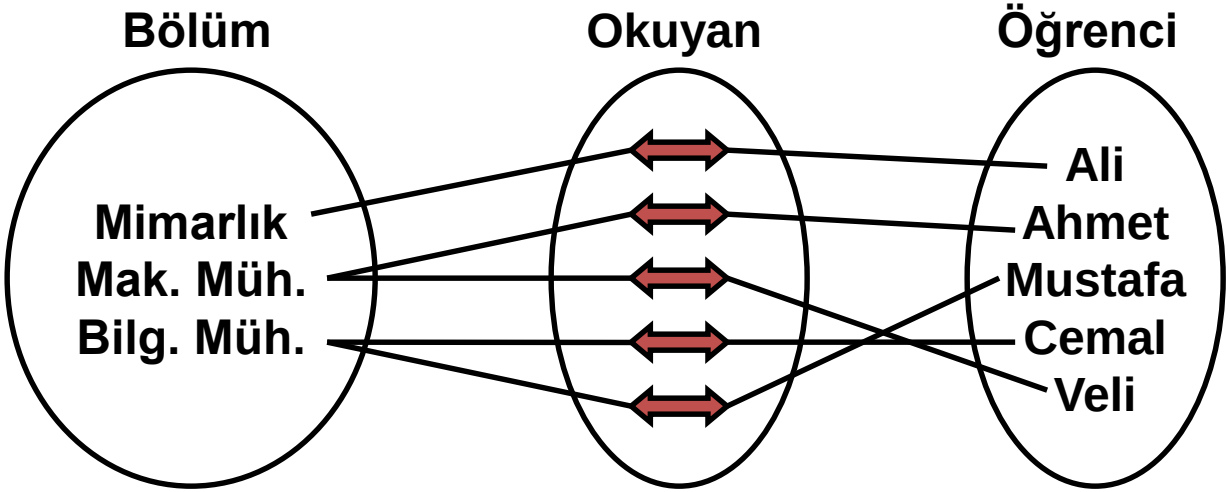
İlişki Türleri : Birden-bire (1-1)

- Her a ile en çok bir b ve her b ile de en çok bir a arasında ilişki kurulabilir ($a \in A$, $b \in B$).
- Örnek: “Evlilik” ilişkisi T.C. Medeni Kanunu’na göre birden-bire’dir.



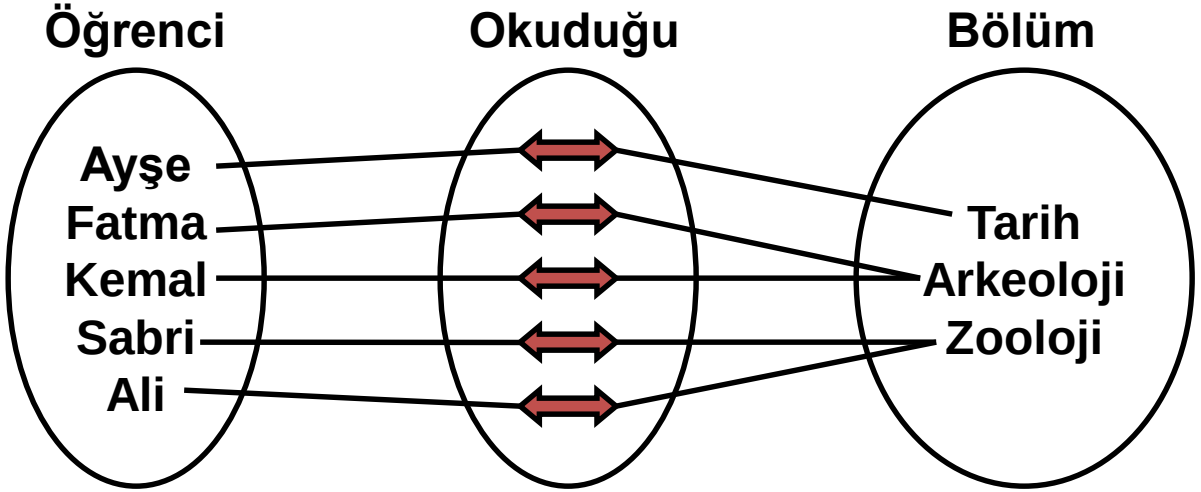
İlişki Türleri : Birden-çoğa (1-n)

- Her a ile sıfır, bir veya birçok b ve her b ile de en çok bir a arasında ilişki kurulabilir.
- Örnek: “Bölüm” ve “Öğrenci” varlık kümeleri arasındaki “Okuyan” ilişkisi, bölümden öğrenciye doğru birden-çoğa şeklindedir.



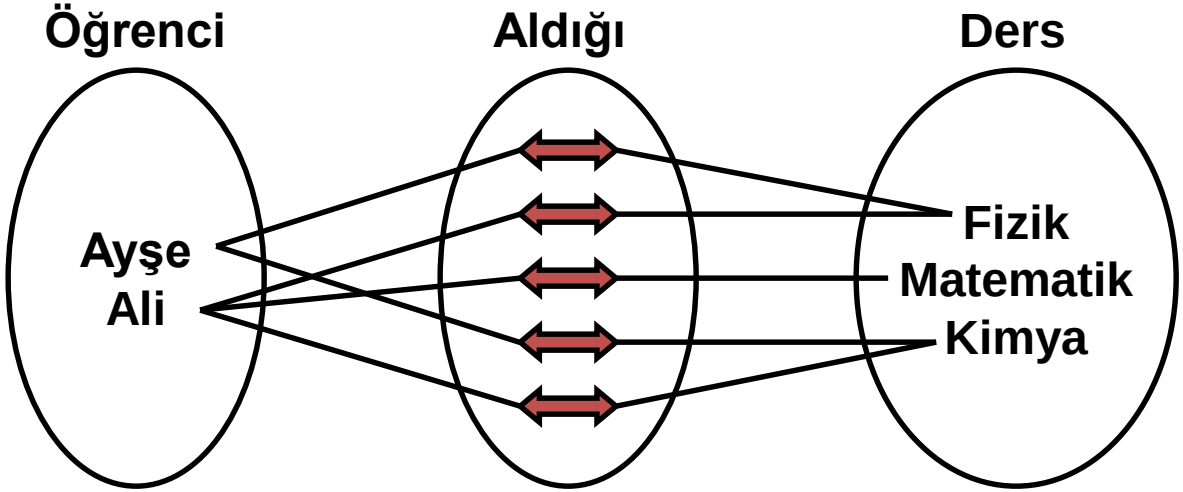
İlişki Türleri : Çoktan-bire (n-1)

- Her a ile en çok bir b ve her b ile de sıfır, bir veya birçok a arasında ilişki kurulabilir.
- Örnek: “Öğrenci” ve “Bölüm” varlık kümeleri arasındaki “Okuduğu” ilişkisi, öğrenciden bölüme doğru çoktan-bire şeklindedir.



İlişki Türleri : Çoktan-çoğa (n-m)

- Her a ile sıfır, bir veya birçok b ve her b ile de sıfır, bir veya birçok a arasında ilişki kurulabilir.
- Örnek: “Öğrenci” ve “Ders” varlık kümeleri arasındaki “Aldığı” ilişkisi, çoktan-çoğa şeklinde bir ilişkidir.



İlişki Türünü Belirlerken…

- Yapılan modellemenin gerçek dünyaya uygunluğunu sağlaması için tanımlanan ilişki kümesinin türünün doğru belirlenmesi önemlidir.
- İlişki kümelerinin türü evrensel ve değişmez değildir. Gerçek dünya kurallarının bir yansıması olarak bir ilişkinin türü bir kurumdan diğerine değişebilir.
- İlişkinin türünün aynı kurum içinde zamanla değişmesi de olasıdır.

Tür Seçimi için Örnek

- “Depo” ve “Malzeme” varlık kümeleri arasındaki “Bulunan” ilişkisi genelde çoktan-çoğa bir ilişkidir. Ancak eğer malzemeler sınıflandırılmış ve her sınıftaki malzemeler yalnız bir depoda bulunuyorsa bu ilişki depodan malzemeye bire-birçok şeklinde olacaktır.

Varolma Bağımlılığı

- “b” varlığının bulunması “a” varlığının bulunmasına bağlı ise, yani:
  - A ve B varlık kümeleri arasında birden-bire, ya da A'dan B'ye birden-çoğa bir R ilişkisi varsa … VE …
  - bir b'nin varolması bu b ile bir a arasında r ilişkisinin kurulmuş olmasına bağlı ise (r ilişkisi yüzünden bir a'ya bağlı olmayan b'ler var olamıyorsa)

  b a'ya varolma bağımlıdır denir.

Varolma Bağımlılığı

- Bu durumda;
  - a baskın (dominant) ya da birincil varlık,
  - b bağımlı (subordinate) ya da ikincil varlık olarak nitelenir.
  - ve a'nın silinmesi durumunda b'nin bir anlamı kalmayacaktır.

Varolma Bağımlılığı için Örnek

- Eğer ilgili öğretim kurumunda “bölümü belli olmayan öğrenci bulunamaz” kuralı geçerli ise, yani “her öğrenci mutlaka bir bölümün öğrencisi olmak zorunda” ise, öğrenci ve bölüm varlık kümeleri arasında varolma bağımlılığı vardır.
- Bu durumda bölüm üstün (dominant) varlık, öğrenci ise bağımlı (subordinate) varlıktır. İlgili öğretim kurumunda bir öğrencinin varolması ancak bir bölümün öğrencisi olması ile mümkündür.

Anahtarlar

- Bir varlık kümesi içindeki varlıkları ya da bir ilişki kümesi içindeki ilişkileri birbirinden ayırt etmek için kullanılan nitelik ya da nitelik grubuna bu varlık ya da ilişki kümesinin anahtarı denir.
- Anahtar, hem varlık kümeleri hem de ilişki kümeleri için geçerli bir kavram olsa da, daha çok varlık kümeleri için kullanılır.

Anahtar Türleri

- Süper anahtar (superkey)
  - Değerleri ile bir kümedeki varlıkları (veya ilişkileri) ayırt etmeyi sağlayan niteliğe (veya nitelik grubuna) bu varlık / ilişki kümesinin süper anahtarı denir. Ayırt etme özelliğine sahip olmak için gereğinden fazla nitelik içerebilir.
- Aday anahtar (candidate key)
  - Eğer bir varlık / ilişki kümesinin süper anahtarının bir altkümesi de bu varlık / ilişki kümesini ayırt edebiliyorsa, bu altküme aday anahtardır (ya da kısaca anahtardır).

Anahtar Türleri

- Birincil Anahtar (Primary Key- PK)

  Tabloda tutulan değerlerden benzersiz yani aynı değeri içermeyecek olan sütun birincil anahtar olarak belirlenir.

  Birincil anahtar olarak belirlenecek olan sütun NULL(boş) değerleri veya birbirinin aynı olan değerleri içeremez.

  Birincil anahtar birden fazla alanın birleşiminden oluşmuşsa alanların her biri aynı kaydı içerebilir fakat NULL değer içeremez.

  Birincil anahtar kayıt silme yada düzeltme gibi işlemlerde çok büyük kolaylık sağlar.

Anahtar Türleri

- Yabancı Anahtar (Foreign Key-FK)

  Yabancı anahtar bir sütundan yada birkaç sütunun birleşiminden oluşabilir.

  Birbiri arasında ilişki bulunan iki tablodan birisinden diğer tablodaki birincil anahtar olan sütuna başvuran sütun yabancı anahtar olarak adlandırılır.

  Yabancı anahtar olan sütun başvurduğu birincil anahtar olan sütunun değerlerini içermelidir.

  NULL olmamalıdır ancak birden fazla aynı değeri içerebilir.

Anahtar Türleri

- Tekil Anahtar(Unique Key – UK)

  Tekil anahtarda sütunlar NULL değeri alabilir.

  Diğer özelliklerinin hepsi Birincil Anahtar ile aynıdır.

Anahtar Türleri için Örnek

- Eğer bir üniversitede tüm öğrencilerin numaraları birbirinden farklı ise, öğrencileri ayırt etmek için öğrenci numarası yeterlidir.
- Bu durumda “öğrenci numarası”, “öğrenci” varlık kümesi için aday anahtar ya da kısaca anahtardır, içinde öğrenci numarası bulunan her nitelik grubu (örneğin “öğrenci numarası”, “adı” ve “soyadı”) ise bu varlık kümesinin süper anahtarıdır.

Güçlü & Zayıf Varlık Kümeleri

- Her varlık kümesi için bir anahtar bulmak mümkün olmayabilir.
- Eğer bir varlık kümesinin niteliklerinden en az bir anahtar oluşturulabiliyorsa, bu varlık kümesine **güçlü** (strong) varlık kümesi denir.
- Eğer bir varlık kümesinin niteliklerinin tümü alınsa bile bir anahtar oluşturmuyorsa bu varlık kümesine **zayıf** (weak) varlık kümesi denir.

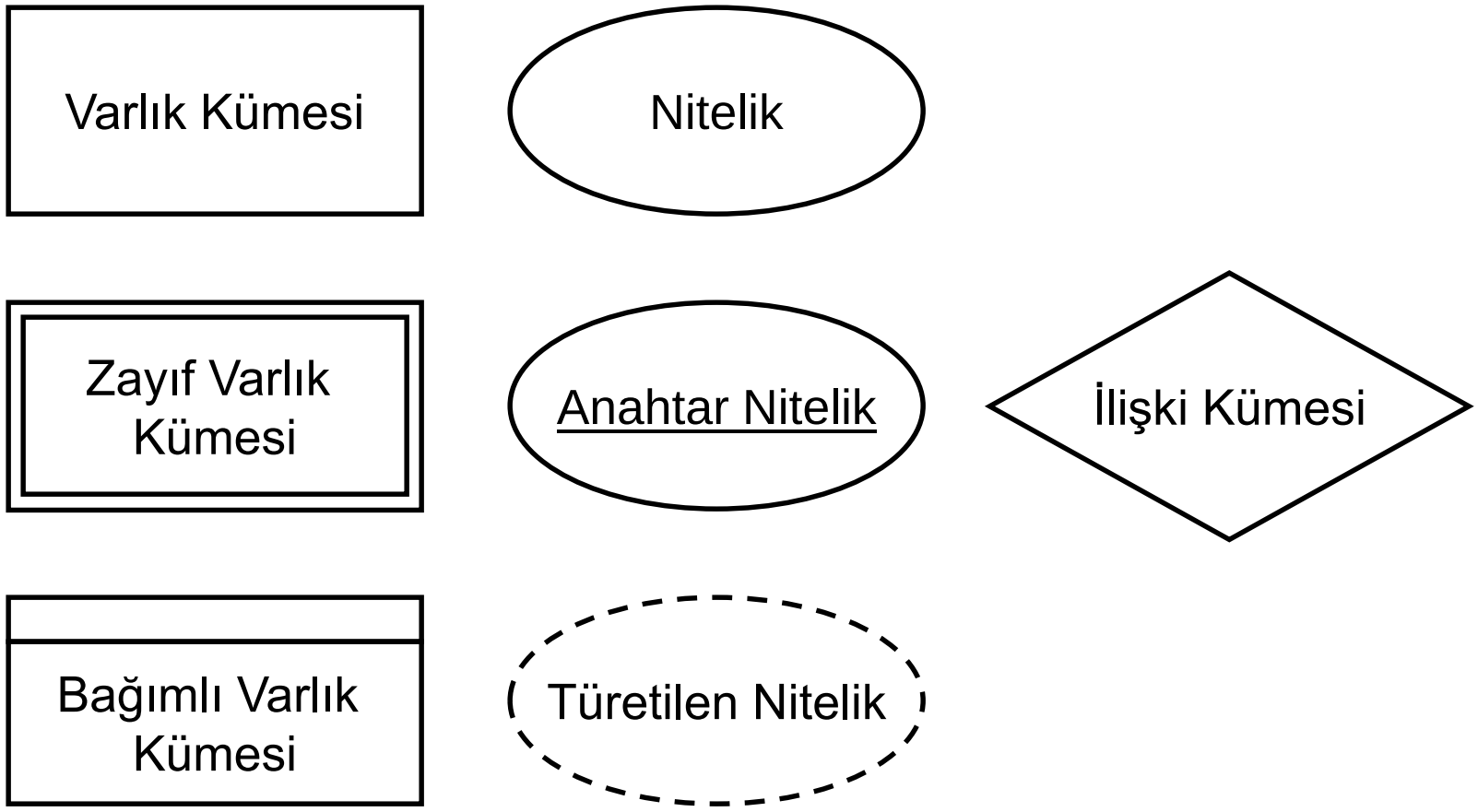
Zayıf Varlığı Güçlendirmek

- Zayıf bir varlık kümesinde, niteliklerin değerleri ile varlıkları birbirinden ayırdetmek mümkün değildir.
- Zayıf bir varlık kümesinin anlamlı olabilmesi için şu özelliklere sahip olması gerekir:
  - Bu varlık kümesi ile güçlü bir varlık kümesi arasında birden-bire ya da (güçlüden-zayıfa) birden-çoğa bir ilişki bulunmalıdır.
  - Zayıf varlıklar için bu ilişkinin var olma bağımlılığı oluşturmalıdır.
  - Zayıf varlık kümesinin nitelikleri arasında, aynı güçlü varlığa bağlı zayıf varlıkları birbirinden ayırt etmeyi sağlayan bir nitelik grubu (discriminator) bulunmalıdır.
- Böylece zayıf bir varlığın anahtarı, bağlı olduğu üstün varlığın anahtarına ayırıcı nitelikler eklenerek elde edilir.

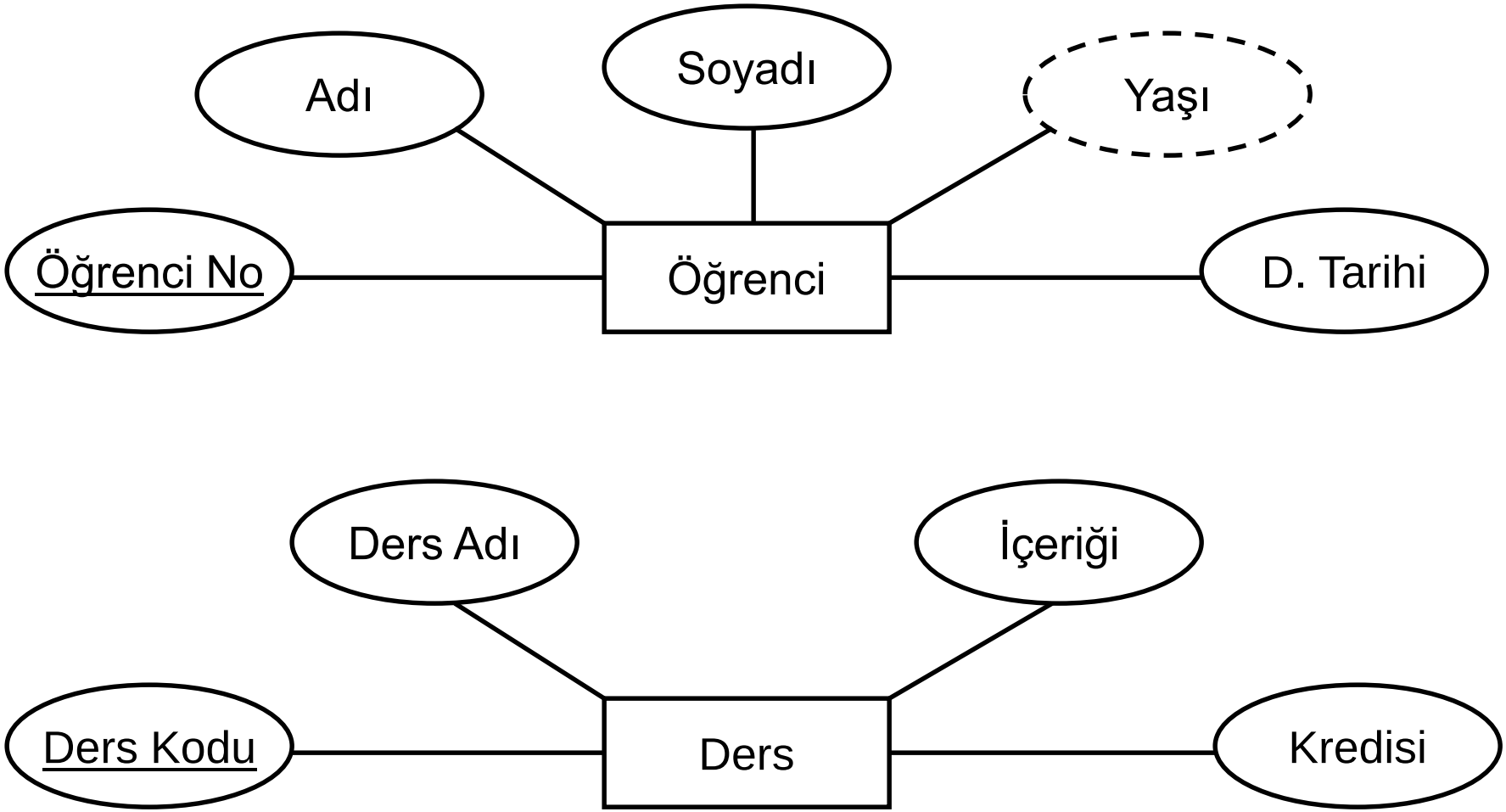
Zayıf Varlığı Güçlendirme Örneği

- Türkiye'deki tüm lise öğrencilerinin bilgilerini içeren ÖĞRENCİ varlık kümesi zayıf bir varlık kümesidir. Çünkü farklı liselerde öğrenci numarası, adı ve soyadı aynı olan öğrenciler bulunabilir.
- ÖĞRENCİ varlık kümesi ile LİSE varlık kümesi arasında bir OKUYAN ilişkisi kurulursa, öğrencileri birbirinden ayırdetmek için kullanılan ÖĞR_NO niteliğine, LİSE varlık kümesinin anahtarı olan LİSE_KODU eklenir. Bu durumda ÖĞRENCİ varlık kümesinin anahtarı (LİSE_KODU, ÖĞR_NO) ikilisi olur.

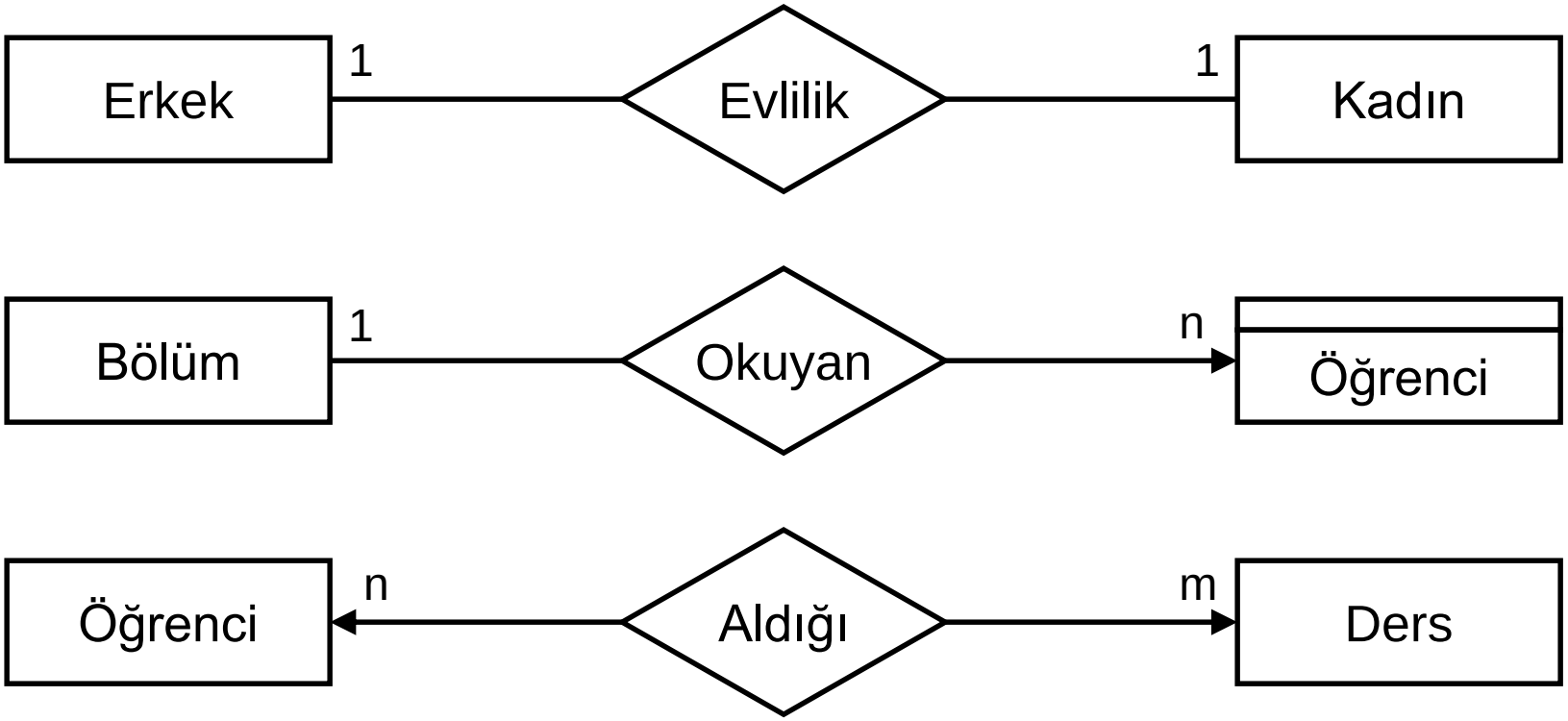
Varlık-İlişki Çizelgeleri



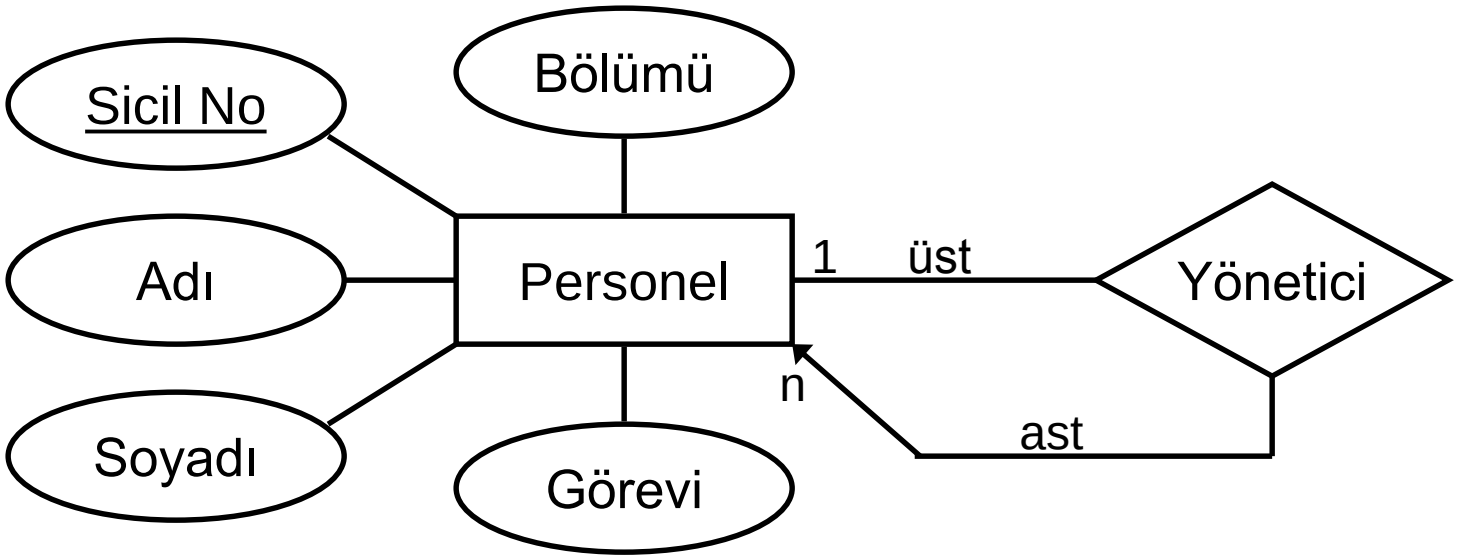
Varlık Kümesi ve Nitelik Örnekleri



İlişki Kümesi Örnekleri

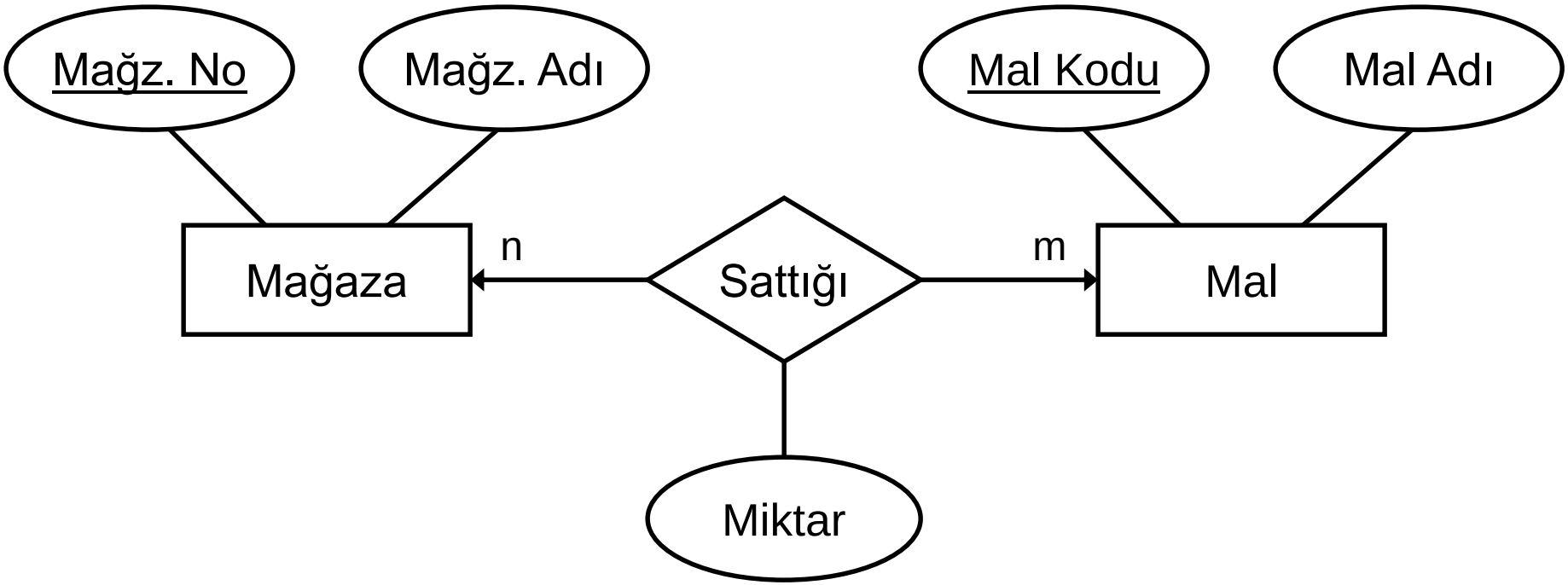


Rol Örnekleri



İlişkilerde Nitelik

- İlişkilerde de tanımlayıcı nitelikler bulunabilir.
- Aşağıdaki “Miktar” niteliği “Sattığı” ilişkisi için tanımlayıcı niteliktir.

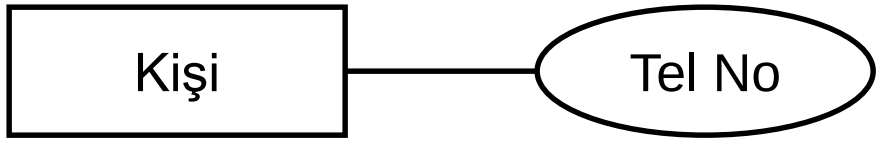


Varlık, İlişki ve Nitelik seçimleri

- Varlık-ilişki modelinin sunduğu 3 temel kavram: varlık kümesi, nitelik, ilişki kümesi kavramlarıdır.
- Gerçek dünyayı soyutlayarak semantik modelleme yapmak için kullanılabilen bu üç kavram birbirinden kesin çizgilerle ayrılmamaktadır.
- Varlık kümesi ile nitelik, ya da varlık kümesi ile ilişki kümeleri birbirinin yerine kullanılabilir; aynı konuda değişik düzenlemeler yapılabilir.

Örnek : Telefon Numarası (1)

- “telefon numarası” kişi varlık kümesinin bir niteliği olarak düşünülürse;
  - “telefon numarası” kişilerden bağımsız olarak varolamaz.
  - bir kişinin sadece bir telefon numarası bulunabilir.
  - birden çok kişinin telefon numarası aynı olabilir (telefon numarası kişi varlık kümesinin anahtarlarından biri olarak tanımlanmadığı sürece).

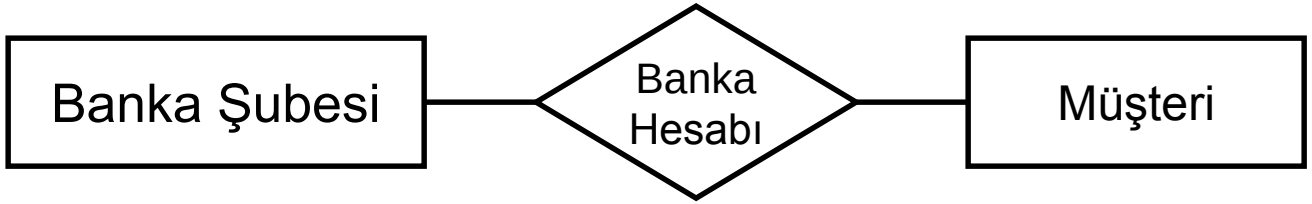


Örnek : Telefon Numarası (2)

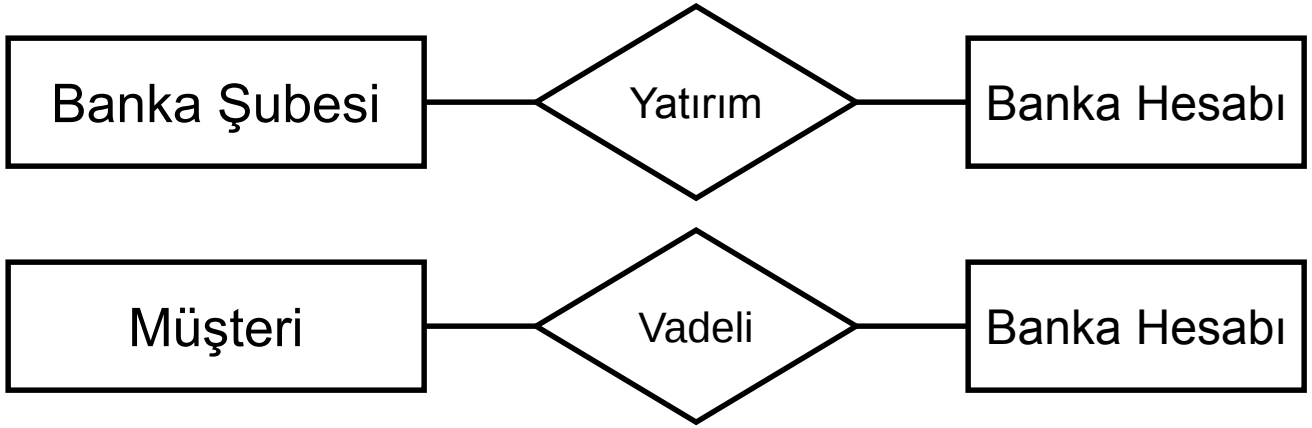
- “telefon numarası” ayrı bir varlık kümesi olarak düşünülüp, bu varlık kümesi ile kişi varlık kümesi arasında ilişki kurulursa;
  - telefonun numarası dışında nitelikleri de bulunabilir.
  - kişi ve telefon varlık kümeleri arasındaki ilişkinin türüne göre her kişinin bir ya da birçok telefonu olabilir.
  - bir telefon numarası bir ya da birçok kişiye verilebilir.

Örnek : Banka Hesabı

- "Banka hesabı" banka şubesi ve müşteri varlık kümeleri arasında bir ilişki olarak düşünülebilir.



- “Banka hesabı” ayrı bir varlık kümesi olarak düşünülüp bu varlık kümesi ile banka şubesi ve müşteri varlık kümeleri arasında birer ilişki de kurulabilir.



Seçerken Dikkat !

- Veri modellemede varlık kümelerinin, niteliklerin ve ilişki kümelerinin seçimi çok önemlidir, Ancak bunların nasıl seçileceğine ilişkin kesin kurallar da yoktur.
- Kuruluşun öncelikleri ve uygulamaların özellikleri yanında veri modellemeyi gerçekleştiren bilişim teknik personelinin anlayışı da düzenlemede etkili olmaktadır.