VERİTABANI DERS NOTLARI

Yrd.Doç.Dr. Buket Doğan

Ders İçeriği

- Veritabanı ve ilişkisel veritabanı kavramı, tasarımı ve yönetimini anlamak,
- veri tabanı sistemlerinin denetimi ve erişimi yöntemlerini ve araçlarını öğrenmek, (SQL komutlarının kullanımı)
- verilecek teori bilgiler temelinde VTYS uygulamalarını (Microsoft Access) yapmaktır.

TEMEL KAVRAMLAR

- Veri
- Olguların, kavramların, veya talimatların, insan tarafından veya otomatik yolla iletişim, yorumlama ve işleme amacına uygun bir biçimde ifadesidir.
- Genellikle, biz veri veya veri birimleri üzerindeki işlemlerimizi varlık hakkında her hangi bilgi almak için gerçekleştiririz.
- Veri kaydedilebilir bilinen gerçeklerdir.
- Örneğin bir kişinin ismi, adresi, telefon numarası gibi.

VERİTABANI NEDİR

- **Veri tabanı**
  - Düzenli bilgiler topluluğudur.
  - Bilgisayar ortamında saklanan düzenli verilerdir.
  - **Bilgisayar terminolojisinde, sistematik erişim imkanı olan, yönetilebilir, güncellenebilir , taşınabilir, birbirleri arasında tanımlı ilişkiler bulunabilen bilgiler kümesidir**.
  - Bilgisayarda sistematik şekilde saklanmış, programlarca işlenebilecek veri yığınıdır

Veri Tabanı Yönetim Sistemi-VTYS

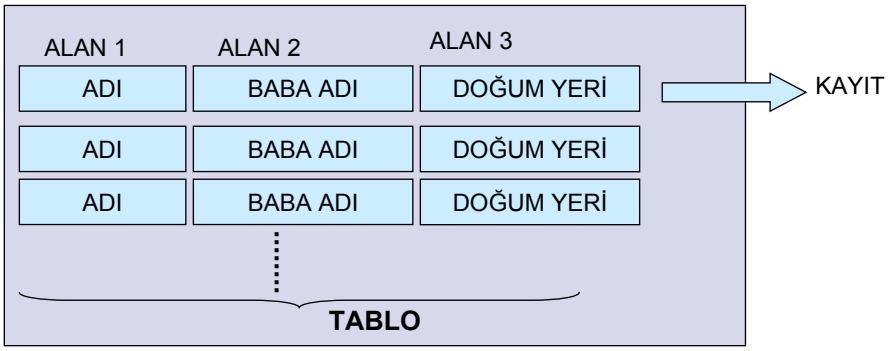
Veri tabanı tanımlamak, yaratmak, yaşatmak ve veri tabanına denetimli erişim sağlamak için kullanılan yazılım sistemidir.

TEMEL KAVRAMLAR

KLASİK DOSYA YAPILARI

- Veri saklama birimlerinde depolanan veri topluluklarına “dosya” adı verilmektedir.
- Dosyalar ise kendi içersinden kayıtlara bölünmüştür.
- Örneğin öğrencilerin bilgilerinin tutulduğu bir dosyayı düşünelim:

TEMEL KAVRAMLAR



Kayıtlar birbiri ile ilişkili alanlardan(field) oluşmaktadır.

Her kayıt farklı bilgileri içermektedir.

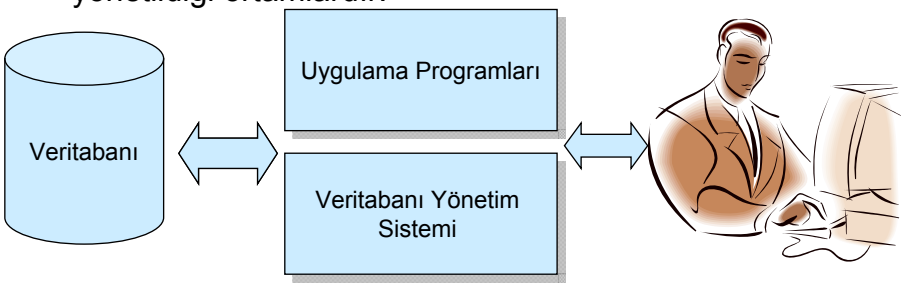
DOSYA SİSTEMLERİNİN SAKINCALARI

- Klasik dosya sistemleri kullanılmaya başlandıktan sonra bazı dezavantajları olduğu ortaya çıkmıştır. Bunlar şöyle sıralanabilir :

- **Veri tekrarı:** Aynı veri çeşitli dosyalarda birden fazla yer alabilmektedir buda sistemin hantallaşmasına neden olur. Mesela bir stok dosyasında stok numarası verisinin malzeme dosyasında, fatura dosyasında ve ambar girişi dosyasında yer alması gibi.
- **Verinin birkaç dosyada güncellemesi:** Veri birden fazla dosyada tekrar edilebildiği için, verinin bir dosyada güncellenip diğerlerinde güncellenmemesi Veri Bütünlüğünün (Data Integrity) bozulmasına neden olabilir. Buna bağlı olarak birbiri ile çelişen raporlar üretilebilir.
- **Belleğin tekrarlı bilgi nedeniyle israfı:** Aynı verinin birden fazla dosya içinde bulunması nedeniyle kullanılan veri hard diskte fazla yer işgal edecek. Yani hard disk tekrarlı veriler için kullanılmış olacaktır.
- **Sadece belirli bir dilin kullanılması** :Verilerin dosya sisteminde saklandığı ortamlar için değişik programlama dillerinden bir tanesi kullanılır. Kullanılan bu programlama dili ise SQL dili gibi esnek değildir.

VERİTABANI SİSTEMLERİ

- Veritabanı sistemleri, veri kümelerinin düzenli biçimde tutulduğu ve bu verilerin yazılımlar aracılığı ile yönetildiği ortamlardır.



VERİTABANI SİSTEMLERİ

- VYS'ler aşağıdaki bilgileri barındırmaktadır
  - İlişkili olan veriler (Collection of interrelated data)
  - Veriye ulaşmak için gerekli olan yazılımlar kümesi
- Veritabanı Uygulamaları (Database Applications)
  - Bankalar: tüm işlemler / hareketler
  - Havayolları:rezervasyonlar, zaman programları
  - Üniversiteler:Kayıt, notlar
  - Satış: müşteriler,ürünler, alımlar
  - Çevrimiçi Perakendicileri:Sipariş Kayıtlar, Kişiselleştirilmiş tavsiyeler.
  - Üretim: imalat, stok, siparişler, tedarik ihtiyaçları
  - İnsan Kaynakları:personel kayıtları,maaşlar, vergi kesintileri
- Veritabanları hayatımızın her alanında kullanılmaktadır.

Veritabanı Sistemlerinin Üstünlükleri

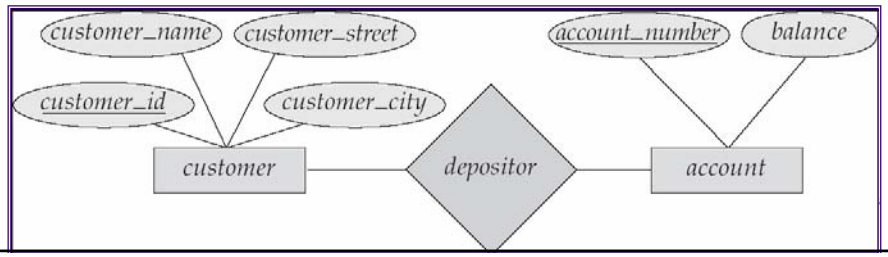
- Verinin tekrarlanmasını önler.
- Veritabanı sistemleri alt sistemler arasında ilişki kurulması ve birçok uygulamada verilerin aynı veritabanı içersinde ortak olarak tasarlanmasını öngörür.
- Verilerin tutarlı olmasını sağlar.
- Veri bütünlüğü(data integrity), verinin doğruluğunu ve tutarlığını ifade etmektedir. Veri girişlerine kısıtlar konularak sadece istenilen aralıkta değer girişi sağlanabilir.

Veritabanı Sistemlerinin Üstünlükleri

- Aynı andaki erişimlerde tutarsızlıkların ortaya çıkmasını önler.
- Veritabanı uygulamalarında, veritabanı nesneleri başka uygulamalar ve farklı kullanıcılar tarafından paylaşılabilir.
- Verilerin güvenliğini sağlar.
- Her kullanıcının erişeceği veriler ayrı ayrı tanımlanabilir. Yetkiler ve kısıtlamalar ile istenilen kullanıcı erişim ayarları gerçekleştirilir.

Varlık-İlişki Modeli (The Entity-Relationship Model)

- Veri çözümleme ve modellemede ilişkilerin ortaya konması için kullanılan araçtır.
  - Varlık (Entity): Bir alan içersinde diğer nesnelerden ayırt edilebilen bir şey (“thing”) yada “nesne” (“object”)
- Niteliklerin kümesi (set of attributes) tarafından tanımlanır.
  - İlişki (Relationship): Birden fazla varlığın arasındaki bağıntı-ilişki.
- Görsel olarak varlık-ilişki tablosu ile gösterilir:



Varlık-İlişki Modeli

- Varlık(Entity): Var olan ve diğer varlıklardan ayırt edilebilen nesnedir. (Bir kitap, öğrenci, veritabanı dersi birer varlıktır.

- Varlık Dizisi: Aynı türdeki varlıklar varlık kümesini oluştururlar. Bir okuldaki tüm öğrenciler “öğrenci” isimli varlık kümesi olarak değerlendirilir.

İlişki ve İlişki Kümeleri

- Varlıklar arasındaki bağlantıya ilişki adı verilir.örneğin “Burak” varlığı ile “Dersler” varlığı arasından ilişki vardır.
- İlişki kümesi, aynı türdeki ilişkilerin kümesidir, bu ilişki kümesi R ile gösterilir.
- E1,E2,…En varlık kümeleri, R ise ilişkiyi tanımlamaktadır.

İlişki ve İlişki Kümeleri

- E1={Ayşe,Burak}
- E2={Elektronik, İngilizce}
- Bu iki küme arasındaki ilişki, öğrenciler ve dersler arasındaki ilişkidir. Tüm öğrencilerle tüm dersler arasındaki ilişki kartezyen çarpımı yapılarak ifade edilir.
- E1xE2={(**Ayşe,Elektronik),(Ayşe,İngilizce),** **(Burak, Elektronik),(Burak,İngilizce)}**
- İki veri kümesi arasındaki geçerli tüm ilişkiler, R ilişki kümesinin bir alt kümesidir.

İlişki ve İlişki Kümeleri

Müşteri no	Müşteri adı
101	Ayşe
203	Mehmet
405	Derya
607	Selin

Hesaplar	Bakiye
33344	1.000,00 YTL
33567	2.500,00 YTL
33790	45.000,00 YTL
34013	5.000,00 YTL

R1={(Ayşe,33567),(Mehmet,33344)}

R2={(Derya,33790)}

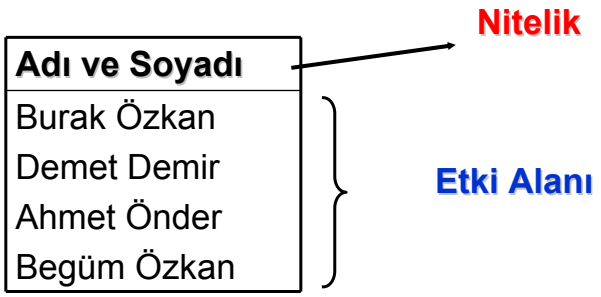
R3={(Selin,34013)}

Nitelikler

- Bir varlık çok sayıda nitelik yardımıyla tanımlanabilir. Örneğin, personel varlığının nitelikleri şu şekilde olabilir:
  - Personel No
  - Adı ve Soyadı
  - Adres
  - SSK no
  - Gelir

Etki Alanı

- Niteliğin aldığı değerlere etki alanı(domain) adı verilir.



Türetilen Nitelik

- Bir nitelik kullanılarak, bir başka varlık nitelik elde edilebiliyorsa bu yeni niteliğe “türetilen nitelik” adı verilir.
- Örneğin personel varlığının “doğum tarihi” niteliğinden yararlanılarak “yaş” niteliği elde edilebilir.

Çok Değere Sahip Nitelik

- Bir nitelik birden fazla değer ile eşleşebiliyor ise, “çok değere sahip nitelik” adı verilir.
- Örneğin, öğretmen varlığının **dersler** niteliği birden fazla değeri kapsar. Bir öğretmen birden fazla derse girmektedir.
- Öğrenci varlığının **okuduğu kitaplar** niteliği birden fazla kitabı kapsayabilir.

Birleşik Nitelik

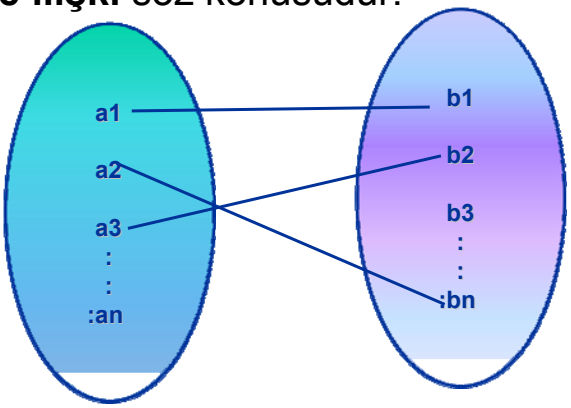
- Birden fazla nitelik birleştirilerek, yeni bir nitelik oluşturulabilir. Bu tür niteliklere birleşik nitelik denir
- Örneğin personelin “cadde” ve “şehir” nitelikleri birleştirilerek “ADRES” isimli yeni bir nitelik oluşturulabilir.

Varlıklar arası İlişkiler(Eşleme)

- Bir varlıkla ilişkili olabilecek varlıkların sayısına eşleme sayısı adı verilir.
- Eşleme sayısı n>=2 varlık için söz konusudur ve ikili ilişkilerin ortaya konulması açısından yararlıdır.
- A ve B gibi iki varlık kümesi arasındaki R ilişki kümesi için eşleme durumları şu şekilde ifade edilir:
  - **Birden-bire (One to One)**
  - **Birden-çoğa(One to Many)**
  - **Çoktan-bire (Many to One)**
  - **Çoktan-çoğa (Many to Many)**

Birden-bire İlişki

- A varlık kümesi içindeki bir varlık, B kümesi içindeki **sadece** bir varlık ile ilişkili ise **birden-bire ilişki** söz konusudur.



Birden-bire İlişki

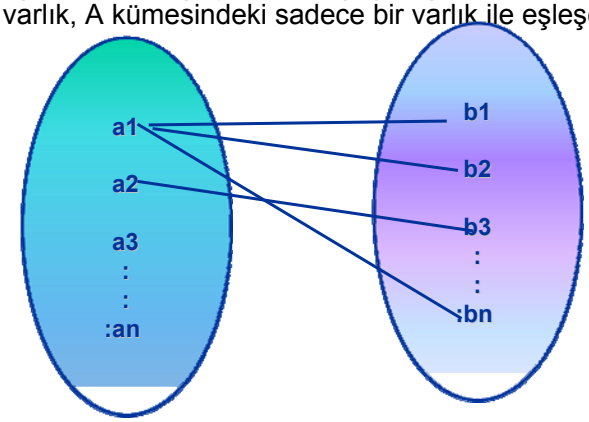
Müşteri no	Müşteri adı
101	Ayşe
203	Mehmet
405	Derya
607	Selin

Hesaplar	Bakiye
33344	1.000,00 YTL
33567	2.500,00 YTL
33790	45.000,00 YTL
34013	5.000,00 YTL

Her müşterinin bir hesabı olabilir.

Birden-çoğa İlişki

- A kümesi içindeki bir varlık B kümesi içindeki birden fazla varlık ile ilişkili ise, bu ilişkiye birden-çoğa ilişki adı verilir.B kümesindeki bir varlık, A kümesindeki sadece bir varlık ile eşleşebilir.



Birden-çoğa İlişki

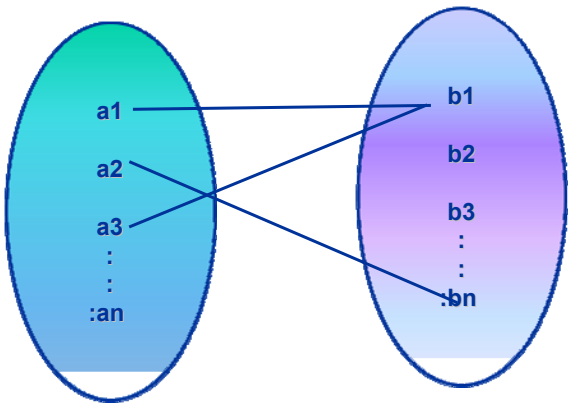
ID	Öğretmen adı
101	Ayşe
203	Ahmet
405	Derya
607	Selin

NO	OGR_ID	Girdiği DERS
1	101	Matematik
2	101	Geometri
3	405	Bilgisayar Prog.
4	405	Office Programları

Öğretmenler birden fazla derse girmektedir.

Çoktan-bire İlişki

- A varlık kümesindeki birden fazla varlık, B kümesindeki bir varlık ile ilişkili ise bu eşleşmeye çoktan-bire ilişki adı verilir.



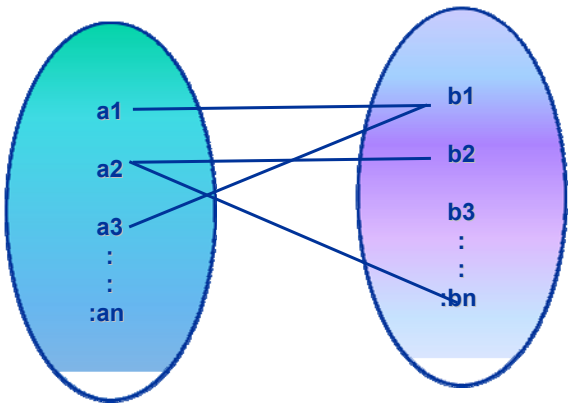
Çoktan-bire İlişki

No	Kişi_ID	DERS
1	101	Matematik
2	101	Geometri
3	405	Bilg. Prog.
4	405	Office Prog

ID	İsim	Doğum.Tarihi
101	Ayşe	3.06.1990
203	Ahmet	12.04.1980
405	Derya	15.04.1983
607	Selin	5.07.1981

Çoktan-Çoğa İlişki

- A varlık kümesindeki birden fazla varlık, B kümesindeki birden fazla varlık ile ilişkili ise bu eşleşmeye çoktan-çoğa ilişki adı verilir.



Çoktan-Çoğa İlişki

- Çoktan-çoğa ilişki en genel ilişki biçimidir. Bu ilişki herhangi bir sınırlamanın olduğu durumlar için geçerli olacaktır.
- Müşteri-hesap ilişkilerinde aile üyelerinin ortak hesap açabilmesi durumunda çoktan-çoğa ilişki söz konusu olacaktır.

Varoluş Koşulu

- Eğer bir X varlığının bulunması Y varlığının bulunmasına bağlı ise, X'in Y'ye bağlı olduğundan söz edilebilir.
- Y silinirse, X'in bir anlamı kalmayacaktır.
- Bu durumda Y baskın varlık(dominant entity)
- X ise bağımlı varlık(subordinate entity) adı verilir.
  - Örneğin, bir müşterinin hesabı silineceğinde, bu müşterinin hesap hareketlerinin de silinmesi gerekmektedir. Hesap hareketleri, hesap varolmadan var olamaz.

Anahtar

- Varlık kümesi içinde, varlıkları birbirinden ayırt etmek için kullanılan bu tür niteliklere varlık kümesinin anahtarı adı verilir. İki tür anahtar vardır.
- **Süper anahtar (superkey):**Varlık kümesinde yer alan bir varlığı kesin olarak tanımlamaya yarayan anahtara süper anahtar adı verilmketedir. Bu anahtar sadece bir nitelikten oluşabileceği gibi, birden fazla niteliğin birleşiminden de oluşabilir. Süper anahtarlar süper küme oluşturur. Bir süper anahtarın herhangi bir süper kümesi daima bir süper anahtar olarak kabul edilir.
- Örneğin SSK no süper anahtardır. Fakat isim alanı süper anahtar olamaz. SSk no ve isim alanı birlikte süper anahtar olarak kabul edilebilir.

Anahtar

- **Aday anahtar (candidate key) :** Varlık kümesinde bir varlığı tanımlamaya yarayan bir başka anahtar türü aday anahtar dır.
- Bir varlık kümesinin süper anahtarı bir veya daha fazla niteliğin birleşiminden oluşabilmektedir.
- Aday anahtar ise, süper anahtar özelliklerine sahip tek nitelikli anahtardır.

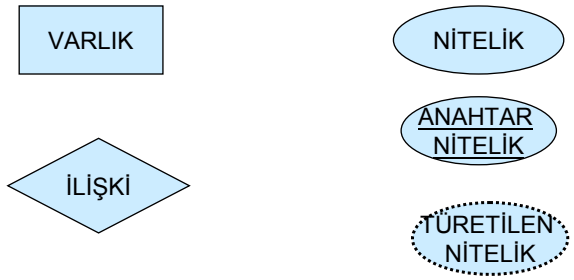
Anahtar

- Eger bir üniversitede tüm ögrencilerin numaraları birbirinden farklı ise ögrencileri belirlemek için ögrenci numarası yeterlidir.
- Bu durumda ögrenci numarası ögrenci varlık kümesi için aday anahtardır.
- İçinde ögrenci numarası bulunan her nitelik grubu ise(ögrenci numarası, adı, soyadı gibi) ise bu varlık kümesinin süper anahtarıdır.

Varlık-İlişki Şemaları (Entity-Relationship Model)

- Varlık-ilişki modeli ; Veritabanı modelleri içerisinde , varlık ve bu varlıkların birbirleri arasındaki ilişkilere dayanarak herhangi bir ön-veri olmaksızın model oluşturmakta kullanılan modeldir.
- Buradaki varlık; benzersiz bir şekilde tanımlanabilen ve bağımsız var olabilme yetisine sahip nesne ya da oluşum olarak tanımlanabilir.
- Varlıklar , ev, araba gibi fiziksel nesneler olabileceği gibi müşteri ödemesi veya sipariş gibi soyut nesneleri de içerirler

Varlık-İlişki Şemaları



Varlık-İlişki Şemaları

Varlıkların aralarında kurulabilecek ilişki türleri aşağıdaki gibi tanımlanır ve model olarak ifade edilir

a- Birden-bire

BÖLÜM-YÖNETİCİ

Çizim 1: Birden-bire

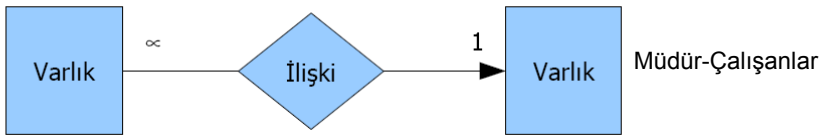
b- Birden-çoğa

BÖLÜM- PROJE

Çizim 2: Birden-Çoğa

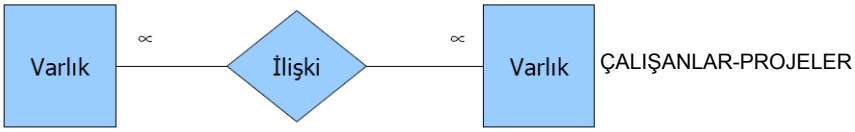
Varlık-İlişki Şemaları

c- Çoktan-bire



Çizim 4: Birden-bire

d- Çoktan- çoğa

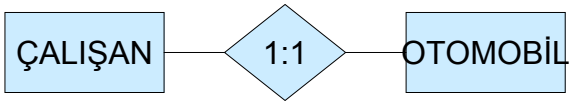


Çizim 5: Çoktan-çoğa

İkili İlişkiler

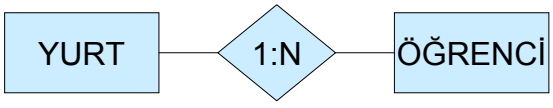
•İlişkinin *büyüklüğü* ile ilgilidir

Bire-bir:



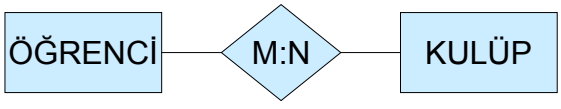
OTO-TAHSİS

Bire-çoklu:



YURTTA-KALMA

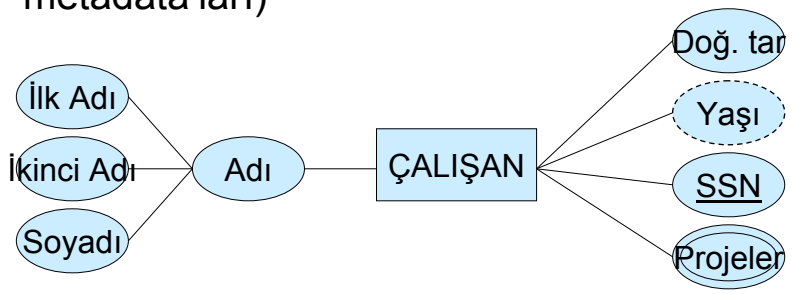
Çoka-çoklu:



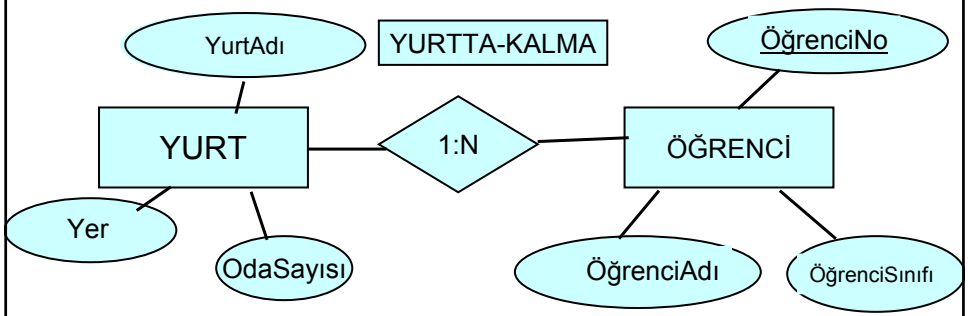
ÖĞRENCİ-KULÜP

Varlık-İlişki Şemaları

- Bir varlığı belirlemeye yarayan, o varlıkla etkileşim kurmak ya da o varlığı kullanmak için gerekli önemli özellikleri (yani varlıkların metadata'ları)



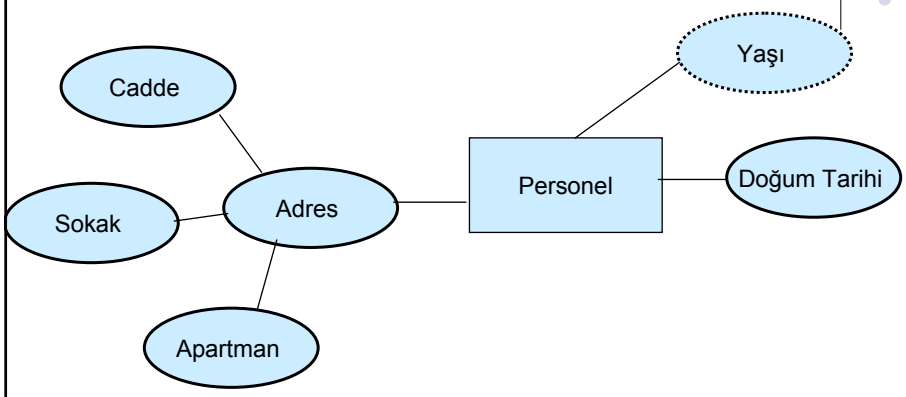
Varlık-İlişki Şemaları



Varlık-İlişki Şemaları

- Bir personel varlığının aşağıda belirtilen özelliklere sahip olduğu varsayılsın
- Adı
- Cadde
- Sokak
- Apartman
- Doğum Tarihi
- Cadde, sokak ve apartman nitelikleri adres ile birleştirilecektir.
- Yaş doğum tarihinden elde edilecektir.

Varlık-İlişki Şemaları

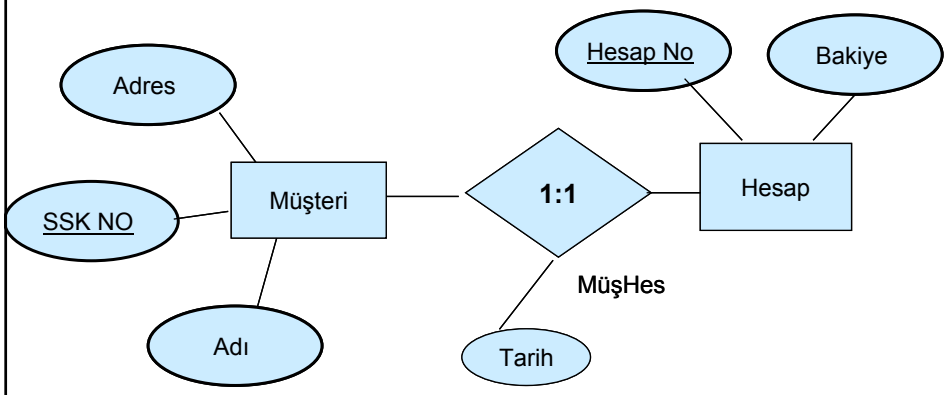


Varlık-İlişki Şemaları

- “Müşteri” ve “Hesap” isimli iki varlık kümesinin nitelikleri aşağıdaki gibidir:
- Müşteri: Adı, SskNo, , Adres
- Hesap: Hesap No, Bakiye

- Bu veriler ile varlık-ilişki şemasını oluşturunuz.

Varlık-İlişki Şemaları



Varlık-ilişki şemalarının tablo haline dönüştürülmesi

Müşteri={SskNo, adı,adres}

Adı	SSk No	Adres

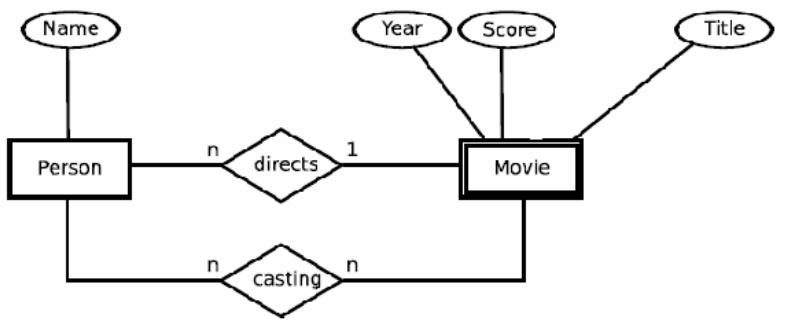
Hesap={Hesap No, Bakiye}

Hesap No	Bakiye

Müşteri Hesap ilişkisi için={SSkNo, Hesap No, Tarih}

SSk No	Hesap No	Tarih

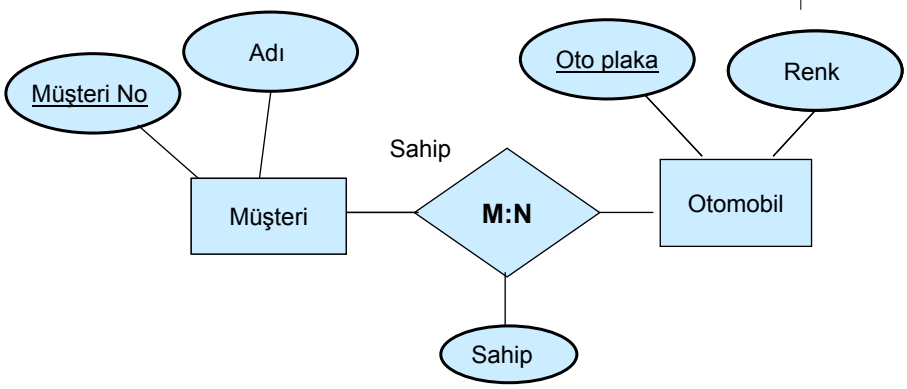
Varlık-İlişki Şemaları



Varlık-İlişki Şemaları

- Bir müşteri birden fazla otomobile sahip olabilir ve her otomobil modeline birden fazla müşteri sahip olabilir.
- Bu durumda otomobiller ve müşteriler arasındaki ilişki çoktan-çoğa biçimindedir.
- Bu ilişkinin varlık-ilişki şemasını çiziniz.

Varlık-İlişki Şemaları



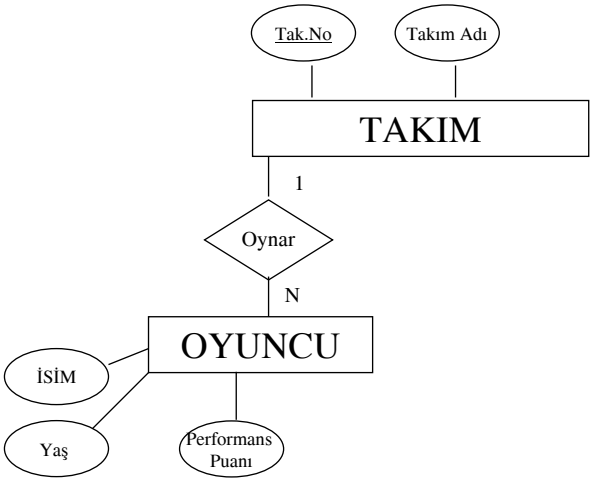
Müşteri ve otomobil varlıkları ve sahip ilişkisi için varlık-ilişki şeması

Varlık-İlişki Şemaları

- Müşteri={müşteri no, adı}
- Otomobil={oto model, renk}
- Sahip={müşteri no, oto plaka, tarih}
- Sahip isimli ilişkiye dayanarak aşağıdaki tablo oluşturulabilir.

Müşteri no	oto plaka	Tarih
345	34 GF 67	12.12.2005
346	45 HN 34	15.11.2002
347	36 BN 67	12.12.2005
348	34 AV 45	15.11.2002

Varlık-İlişki Şemaları



İlişkisel Model

- İlişkisel model, günümüzde en yaygın biçimde kullanılan bir modeldir.
- İlişkisel model varlıklar arasındaki bağlantının içerdiği değerlere göre sıralanması esasına dayanır.
- Bu yaklaşımda veri tabanındaki tüm ilişkiler tablolar biçiminde tanımlanmaktadır.

İlişkisel Veri Tabanı

- İlişkisel veri tabanı, her biri özel isimlere sahip tablolardan oluşur.
- Burada her bir tablo bir varlığa veya bir ilişkiye karşılık gelmektedir.
- Tablonun sütunları nitelikleri, satırları ise bu niteliklerin değerlerini ifade eder.
- Her bir satır bir “kayıt” olarak da düşünülebilir.
- Anahtar alan tablonun tanımlayıcısıdır.

Tablonun özellikleri

- Tablolar sütunlardan oluşur.
- Her bir sütunun ayrı bir adı vardır.
- Her bir sütun aynı niteliğin tanımladığı aynı etki alanının belirlediği değerleri içerir.
- Satırların ve sütunların sırası önemsizdir.
- Her bir satır birbirinden farklıdır.

Soyutsal Katmanlar (Levels of Abstraction

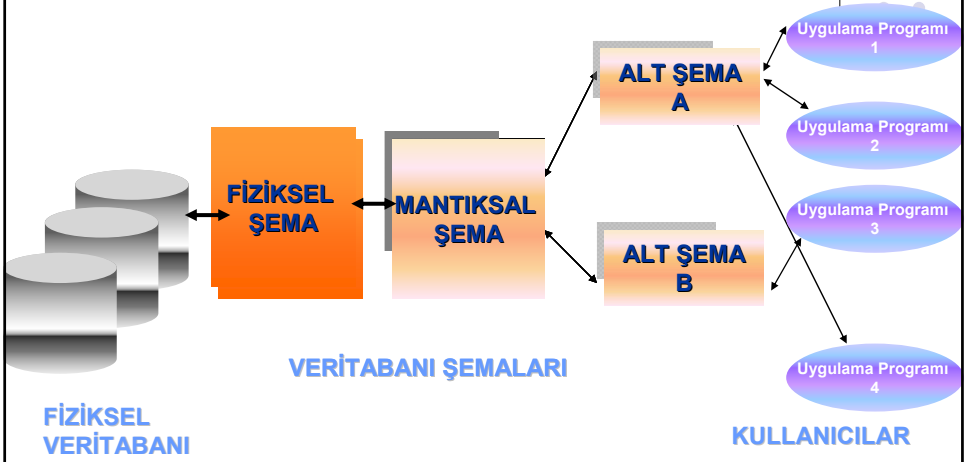
- **Fiziksel Katman (Physical level):** Bir kayıdın nasıl saklanacağını tanımlar(Örneğin,müşteri).
- **Mantıksal Katman (Logical level):** Bir verinin nasıl veritabanında saklanacağını ve veriler arasındaki ilişkileri tanımlar.

  tip (type) *musteri* = **kayıt (record)**
  musteri_id : string;
  musteri_adi : string;
  musteri_sokak : string;
  musteri_il : integer;
  bitiş (end)
- **Görüntü Katmanı (View level):** Tasarımı kullanıcıdan saklar(Örneğin veri tipi veya hangi bilgilerin görüntüleneceği)

Nesneler ve Şemalar

- Programlama dillerindeki tip ve değişkenlere benzerler
- **Şema (Schema)** – Veritabanının mantıksal yapısı
  - Örnek: Veritabanı,Müşteri ve hesap bilgileri ile bunlar arasındaki ilişkiyi barındıran bir kümedir.
  - **Fiziksel Şema (Physical schema)**: Fiziksel düzeyde veritabanı tasarımı(Dosyanın sabit dikteki yeri, büyüklüğü..)
  - **Mantıksal Şema (Logical schema)**: Mantıksal düzeyde veritabanı tasarımı (veri alanları, ilişkiler)
- **Nesneler (Instance)** – Zamandaki herhangi bir noktadaki veritabanı içersindeki içerik.
  - Bir değişkenin değerine benzemektedir.
- **Fiziksel Veri Bağımsızlığı (Physical Data Independence)** – Mantıksal şemayı değiştirmeden fiziksel şemayı değiştirme kabiliyeti
  - Uygulamalar mantıksal şemaya bağlıdır.
  - Genelde , değişik katmanlar ve bileşenler arasındaki arabirimler öyle tanımlanmalıdırki bazı bölümlerin değiştirilmesi diğerlerini fazla etkilememeli.

VERİTABANI ÖRNEĞİ



VERİ MODELLERİ

- Asağıdakilerini tanımlayan araç topluluğu
  - Veri (Data)
  - Veri İlişkileri (Data relationships)
  - Veri Kısıtlamaları (Data constraints)
- İlişkisel Model (Relational model)
- Varlık-ilişki veri modeli (Entity-Relationship data model) (Coğunlukla veri tabanı dizaynı için)

İlişkisel Veritabanı

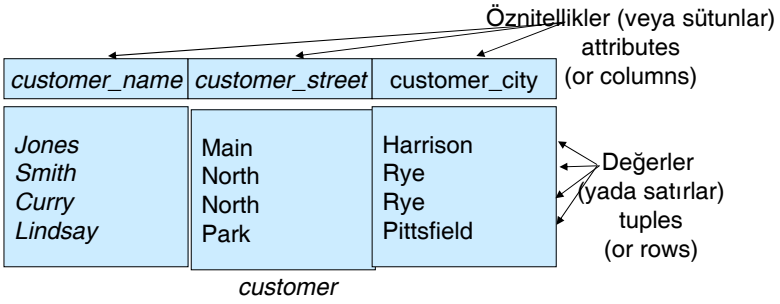
•İlişkisel veritabanı ilişkisel model bazlıdır.
•Veri etrafındaki bilgi ve ilişkiler tablolar tarafından gösterilir

Öznitelikler (Attributes)

customer_id	customer_name	customer_street	customer_city	account_number
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-101
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-201
677-89-9011	Hayes	3 Main St.	Harrison	A-102
182-73-6091	Turner	123 Putnam St.	Stamford	A-305
321-12-3123	Jones	100 Main St.	Harrison	A-217
336-66-9999	Lindsay	175 Park Ave.	Pittsfield	A-222
019-28-3746	Smith	72 North St.	Rye	A-201

İlişki Gösterimi

- İlişkinin o anki değerleri (*relation instance*) bir tablo tarafından gösterilir.
- r deki bir t elemanı bir değerdir ve tablodaki bir satır (row) ile gösterilir.
- Değerlerin sırası önemli değildir. (Değerler keyfi sırada olabilir)



Veritabanı

- Bir veritabanı birden fazla ilişkiye sahiptir.
- Bir şirketin bilgisi birden fazla parçaya bölünmüştür, her parça bilginin bazı bölgelerini ilişkilendirir

  account : hesaplar hakkındaki bilgiyi tutar.
  depositor : hangi müşterinin hangi hesabı tuttuğunu gösteren bilgiyi saklar
  customer : müşteri hakkındaki bilgileri tutar

- Bütün bilgilerin tek bir ilişkide saklanması örneğin
  bank(*account_number, balance, customer_name*, ..)
  aşağıdaki sonuçlara yol açabilir
  - Bilginin tekrarlanması (repetition of information)
    - Örneğin iki müşteri tek hesaba sahip (ne tekrarlar?)
  - Boş değerlerin ihtiyacı
    - Örneğin hesabı olmayan müşterinin gösterimi
- Normalizasyon teorisi (Normalization theory) ilişkisel veri tabanının tasarımından bahsetmektedir

TEMEL KAVRAMLAR

- **Alan(Field) :** Veritabanı tabloları içerisinde saklanacak verinin içeriğine göre, fiziksel tipi belirlenen (Sayı,String vb.) ve isimlendirilen bölümlere denir.(Örnek : Bir field içerisinde bir iş yerindeki personele ait "isim" bilgisi saklanacak ise, programın kontrolü açısından alan(field) ismininde içerik ile ilgili seçilmesi tercih edilir yani "PersonelIsim" veya benzer bir alan(field) ismi seçilmelidir.

- **Tablo(Table) :** İçeriklerine göre ayrıştırlımış alan(field) topluluklarına tablo denir, tablolar veri tabanlarını oluşturan bilgi depolarıdır.
- Örnek= Bir işyeri veritabanını ele alalım, iş yerinin bünyesinde barındırdığı departman adedince veri tabanın içerisinde tablo oluştururuz tablo isimlerinide departmalar ile ilişkili olarak isimlendiririz (Personel Müdürlüğü departmanı için "personel" tablo ismi ,
- Muhasebe departmanı için "muhasebe" vb.) daha sonra bu tabloların içerisine alanlar oluştururuz.(Personel tablosu içerisine
- PersonelIsim","PersonelYas" vb. isimleri taşıyan alanlar açmamız mümkündür.)

BİRİNCİL VE YABANCI ANAHTAR

- **Birincil Anahtar(Primary Key) :** Üzerinde işlem yapılan tabloya ait kayıtları benzersiz olarak tanımlayan alanlardır.
- Örneğin bir okulu ele alalım burda öğrencileri benzersiz biçimde tanımlayabilen en önemli öge şüphesiz ki ögrenci numarasıdır.Bir okulda,isim,soyisim gibi kimilik bilgilerl aynı olabilecek bir çok öğrenci mevcut olabilir fakat,hiç bir öğrencinin, o öğrenciyi tanımlayan, öğrenci numarası aynı olamaz benzer bir mantık ile telefon numaraları da düşünülebilir.

- **Yabancı Anahtarlar(Foreign Keys) :** Tablo içerisindeki verilerin birbirleri ile iletişim kurabilmeleri amacı ile kullanılan benzersiz olması gerekmeyen alanlardır.
- Örneğin içerisinde "Ogrenci_No" birincil anahtarını barındıran "Ogrenciler" isimli tablonun var olduğunu varsayalım ayrıca "Notlar" isimli bir tablonun içerisinde, aynı "Ogrenci_No" alanını çesitli defalar yabancı anahtar olarak kullanmamız gerekebilir (Çünkü, genellikle bir öğrencinin birden fazla dersi ve dolayısıyla "Notlar" isimli tabloya işlenmesi gereken birden fazla sınav notu olacaktır.)
- Bağımsız tablolarda bu şekilde organize edilmiş veriye **"ilişkisel(Relational)"** bu veriyi içeren veritabanına ise **"ilişkisel veritabanı"** ismi verilir.Veritabanlarındaki verinin okunması ve yönetilmesi için kullanılan ortak sorgulama diline **Yapısal Sorgulama Dili (Structured Query Language (SQL))** denir.

BİRİNCİL VE YABANCI ANAHTAR