CSE 204
VERİ TABANI SİSTEMLERİNE GİRİŞ

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

KURS PROGRAMI

- Oda, Gün ve Saat:
  - AMFİ2 - Salı/Perşembe15:30
- Eğitmen Joseph LEDET
  - Ofis Bilgisayar Mühendisliği Binası 2. Kat
  - E-postajosephledet@akdeniz.edu.tr
  - Ofis Saatleri: Salı 12:30 - 13:30, 14:30 - 15:30 veya randevu ile
- Asistan: Doruk BAŞARAN
  - Ofis Bilgisayar Mühendisliği Binası 2. Kat
  - E-postadorukbasaran@akdeniz.edu.tr
  - Ofis Saatleri: Perşembe 10:00 - 12:00 veya randevu ile

KURS HEDEFLERİ

- Bu ders ilişkisel veri modeli, varlık/ilişki modeli, SQL, sorgu optimizasyonu, bütünlük kısıtlamaları ve normalizasyon dahil olmak üzere veritabanı sistemlerinin temellerini tanıtmaktadır. Bu dersi başarıyla tamamlayan öğrenciler aşağıdakileri yapabileceklerdir:
  - DBMS ve RDBMS'yi anlama
  - Bir E-R Modelinin temel bileşenlerini açıklamak
  - Veritabanını normalleştirme
  - SELECT SQL sorgularını kullanarak bir veritabanından veri elde etme
  - UPDATE, DELETE, INSERT SQL sorgularını kullanarak bir veritabanındaki verileri değiştirme
  - Bir veritabanının yapısını tasarlama, oluşturma ve değiştirme

DERS KİTABI

- T. Connolly ve C. Begg

Veritabanı Sistemleri: Tasarım, Uygulama ve Yönetim için Pratik Bir Yaklaşım, Global Baskı.

Pearson (Intl), 2014.

ISBN-13: 9781292061184

KURS BİLGİLERİ

- Ders slaytları, duyurular ve ödevler Microsoft Teams üzerinden erişilebilir olacaktır. Kursa eklendiğinizden emin olun

25B_CSE 204_Veritabanı Sistemleri_Ö_1

- www.microsoft.com/en-us/microsoft-365/microsoft-teams/group- https://chat-software/
- Bu kursla ilgili güncellemeleri sık sık kontrol etmek **sizin** sorumluluğunuzdadır.
- Mobil uygulamayı (Android veya IOS) telefonunuza yükleyebilir, bir duyuru yapıldığında bildirim alabilirsiniz.

SINIFLAND

Kategori	Ağırlık
Ödevler ve Kısa Sınavlar	20%
Ara Sınav	20%
Proje ve Sunum	20%
Final	40%

KATILIM

- Devamlılık ve sınıf tartışmaları öğrenme süreci için çok önemlidir
- Eğer bir dersi kaçırırsanız, derste ne konuşulduğunu öğrenmek sizin sorumluluğunuzdadır.
  - Soru sorun ve cevaplayın

EV ÖDEVLERİ & PROJELER

- Tüm ev ödevleri, eğitmen tarafından son tarih olarak belirlenen tarih ve saate kadar teslim edilmelidir. Ödevlerin geciktiği her gün için %10'luk bir kayıp yaşanacak ve iki günden sonra ödev teslimleri kabul edilmeyecektir.
- Bir dönem projesi verilecektir. Bu projenin detayları dersin üçüncü haftasına kadar tartışılacaktır.

SINAVLAR

- Tıbbi bir mazeret sunulmadığı sürece hiçbir ara sınav için telafi yapılmayacaktır.
- Final sınavı kapsamlı olacak ve dönem boyunca işlenen tüm konulardan sorumlu olacaksınız.
  - Harf notları açıklandıktan sonra son bir tekrar sınavı yapılacaktır. Bu sınav, final sınavı puanınız için bir yedek sınav olacaktır.

AKADEMİK DÜRÜSTLÜK

- Kopya çekerken yakalanırsanız, konunun uygun üniversite yetkililerine bildirilmesi de dahil olmak üzere mümkün olan en üst düzeyde işlem .
- Lütfen kendi işinizi yaparak ve sınıf arkadaşlarınızdan uygunsuz yardım istemeyerek işbirliği yapın.
- Ders konularını ve ödevleri kendi aranızda tartışmanız, bu tartışma bir çözüm alışverişine yol açmadığı sürece teşvik edilmektedir.

KURS ÇİZELGESİ

Hafta	Salı	Per	Bölüm	Geçici Konular
1	11 Şubat	13 Şubat	1, 4	Giriş, İlişkisel Model
2	Şubat 18	20 Şubat	6	SQL
3	25 Şubat	27 Şubat	6, 7	Gelişmiş SQL
4	Mar 04	06 Mart	10	Veritabanı Sistem Geliştirme
5	11 Mart	13 Mart	11	Veritabanı Analizi
6	18 Mart	20 Mart	12, 13	Varlık İlişki Modeli
7	25 Mart	27 Mart	13, 14	E-R Modeli, Normalizasyon
8	01 Nisan	03 Nisan		Ramazan Bayramı ve Normalleşme
9	Nisan 08	10 Nisan		Geçici Vize Haftası
10	15 Nisan	17 Nisan	14, 15	Normalleştirme
11	22 Nisan	24 Nisan	16	Kavramsal Veritabanı Tasarımı
12	29 Nisan	01 Mayıs	17, 18	Mantıksal ve Fiziksel Veritabanı Tasarımı Emek ve Dayanışma Günü
13	06 Mayıs	08 Mayıs		Grup Sunumları
14	13 Mayıs	15 Mayıs		Grup Sunumları

İLETİŞİM

- Bana e-posta göndermeniz gerekiyorsa, lütfen
  - Üniversite e-posta adresimi kullanın (josephledet@akdeniz.edu.tr)
  - Konu satırına "CSE 204" ile başlayın ve yorumunuzun veya sorunuzun kısa bir açıklamasını (en fazla birkaç kelime) ekleyin
    - CSE 204: Grup Projesi
    - CSE 204: Ödev 2
    - CSE 204: Sınav 01
  - Yanıtımı 24 saat içinde (iş günü) vermek için elimden her şeyi yapacağım
  - Sorunuzun ders materyallerinde (slaytlar, duyurular, vb.) daha önce ele alınmadığından emin olun.

İLETİŞİM

- Not verme konusunda endişeleriniz varsa, makul bir süre içinde benimle iletişime geçin
- Notunuzun yükseltilmesini talep ediyorsanız, bunun için geçerli bir nedeniniz olsun
  - Çözüm işe yarıyor, ancak belki de not veren kişi tam olarak anlamadı
  - Not veren çözümün bir kısmını başka bir sayfada görmemiştir
  - NOT: Bu dersi geçmeyi gerçekten, gerçekten, gerçekten istiyorum/ihtiyacım var geçerli bir neden DEĞİLDİR

GRUP PROJESİ

- Eğer bir grubunuz (5 kişilik) ve bir fikriniz varsa, grup liderinin bana aşağıdakileri içeren bir e-posta göndermesini sağlayın
  - Tüm grup üyelerinin isimleri (her üye e-postada cc'lenmelidir)
  - Grup projesi fikri
  - Bu projeyi neden yapmak istediğiniz hakkında biraz bilgi ekleyin
- Ayrıntılar daha sonraki derslerde verilecektir

ÖNEMLI NOTLAR

- Kursa devam edin.
- Şüpheye düştüğünüzde... SORUN!
- Denemek ve başarısız olmak, hiç denememekten daha iyidir.
  - Kısmi kredi sizin dostunuzdur!
- Pratik yap!
- Bir sınıf çakışması varsa veya önceki yılın sınav notlarını veya proje notlarını kullanmak istiyorsanız, en geç üçüncü ders haftasının sonuna kadar (28-Şubat-2025) e-posta ile bize bildirmeniz gerekir.

BU HAFTA

- SQL veya MySQL/PHP için bir araç yükleme
  - https://en.wikipedia.org/wiki/List_of_Apache%E2%80%93MySQL%E2%80%93PHP_packages
- Öneriler
  - XAMPP
  - ORACLE
  - Pencereler
    - Microsoft Access
    - WAMP
  - Mac OSX
    - MAMP
  - Linux
    - LAMBA

CSE 204 - VERİ TABANI SİSTEMLERİNE GİRİŞ

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

GÜNÜN SÖZÜ

- Veri bilgi değildir
- Bilgi, bilgi değildir
- Bilgi anlamak değildir
- Anlayış bilgelik değildir.
  - Clifford Stoll

HEDEFLER

- Veritabanı sistemlerinin bazı yaygın kullanımları.
- Dosya tabanlı sistemlerin özellikleri.
- Dosya tabanlı yaklaşımla ilgili sorunlar.
- Veritabanı teriminin anlamı.
- Veritabanı Yönetim Sistemi (DBMS) teriminin anlamı.

HEDEFLER

- Bir DBMS'nin tipik işlevleri.
- DBMS ortamının başlıca bileşenleri.
- DBMS ortamında yer alan personel.
- DBMS'lerin gelişim tarihi.
- VTYS'lerin avantajları ve dezavantajları.

VERİ TABANI UYGULAMALARI ÖRNEKLERİ

- Süpermarketten yapılan alışverişler
- Kredi kartınızla yapılan alışverişler
- Seyahat acentelerinde tatil rezervasyonu
- Yerel kütüphaneyi kullanma
- Sigorta yaptırmak
- Video kiralama
- İnternet Kullanımı
- Üniversite eğitimi

DOSYA TABANLI SISTEMLER

- Son kullanıcılar için hizmetler gerçekleştiren uygulama programları topluluğu (örn. raporlar).
- Her program kendi verilerini tanımlar ve yönetir.

DOSYA TABANLI IŞLEME

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo
PA14	16 Holhead Rd	Aberdeen	AB7 5SU	House	6	650	CO46
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93

PrivateOwner

ownerNo	fName	lName	address	telNo
CO46	Joe	Keogh	2 Fergus Dr, Aberdeen AB2 7SX	01224-861212
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025

Client

clientNo	fName	lName	address	telNo	prefType	maxRent
CR76	John	Kay	56 High St, London SW1 4EH	0207-774-5632	Flat	425
CR56	Aline	Stewart	64 Fern Dr, Glasgow G42 0BL	0141-848-1825	Flat	350
CR74	Mike	Ritchie	18 Tain St, PA1G 1YQ	01475-392178	House	750
CR62	Mary	Tregear	5 Tarbot Rd, Aberdeen AB9 3ST	01224-196720	Flat	600

DOSYA TABANLI IŞLEME

Lease

leaseNo	propertyNo	clientNo	rent	payment Method	deposit	paid	rentStart	rentFinish	duration
10024	PA14	CR62	650	Visa	1300	Y	1-Jun-13	31-May-14	12
10075	PL94	CR76	400	Cash	800	N	1-Aug-13	31-Jan-14	6
10012	PG21	CR74	600	Cheque	1200	Y	1-Jul-13	30-Jun-14	12

PropertyForRent

propertyNo	street	city	postcode	rent
PA14	16 Holhead	Aberdeen	AB7 5SU	650
PL94	6 Argyll St	London	NW2	400
PG21	18 Dale Rd	Glasgow	G12	600

Client

clientNo	fName	lName	address	telNo
CR76	John	Kay	56 High St, London SW1 4EH	0171-774-5632
CR74	Mike	Ritchie	18 Tain St, PA1G 1YQ	01475-392178
CR62	Mary	Tregear	5 Tarbot Rd, Aberdeen AB9 3ST	01224-196720

DOSYA TABANLI IŞLEME

Sales Files
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)

Contracts Files
Lease (leaseNo, propertyNo, clientNo, rent, paymentMethod, deposit, paid, rentStart, rentFinish, duration)
PropertyForRent (propertyNo, street, city, postcode, rent)
Client (clientNo, fName, lName, address, telNo)

Sales Files
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)

Contracts Files
Lease (leaseNo, propertyNo, clientNo, rent, paymentMethod, deposit, paid, rentStart, rentFinish, duration)
PropertyForRent (propertyNo, street, city, postcode, rent)
Client (clientNo, fName, lName, address, telNo)

DOSYA TABANLI YAKLAŞIMIN SINIRLAMALARI

- Verilerin ayrılması ve izolasyonu
  - Her program kendi veri setini muhafaza eder.
  - Bir programın kullanıcıları, diğer programlar tarafından tutulan potansiyel olarak yararlı verilerin farkında olmayabilir.
- Verilerin çoğaltılması
  - Aynı veriler farklı programlar tarafından tutulmaktadır.
  - Alan israfı ve aynı öğe için potansiyel olarak farklı değerler ve/veya farklı formatlar.

DOSYA TABANLI YAKLAŞIMIN SINIRLAMALARI

- Veri bağımlılığı
  - Dosya yapısı program kodunda tanımlanmıştır.
- Uyumsuz dosya formatları
  - Programlar farklı dillerde yazılmıştır ve bu nedenle birbirlerinin dosyalarına kolayca erişemezler.
- Sabit Sorgular/Uygulama programlarının çoğalması
  - Programlar belirli işlevleri yerine getirmek için yazılır.
  - Her yeni gereksinim yeni bir program gerektirir.

VERITABANI YAKLAŞIMI

- Ortaya çıktı çünkü:
  - Verilerin tanımı, ayrı ve bağımsız olarak depolanmak yerine uygulama programlarına gömülmüştür.
  - Uygulama programları tarafından dayatılanın ötesinde verilere erişim ve manipülasyon üzerinde kontrol yoktur.
- Sonuç:
  - veritabanı ve Veritabanı Yönetim Sistemi (DBMS).

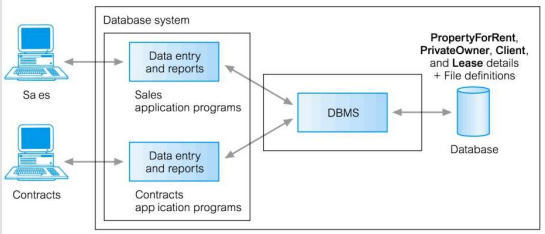
VERİTABANI

- Bir kuruluşun bilgi ihtiyaçlarını karşılamak üzere tasarlanmış, mantıksal olarak ilişkili verilerin (ve bu verilerin bir açıklamasının) paylaşılan koleksiyonu.
- Sistem kataloğu (metadata), program-veri bağımsızlığını için verilerin tanımını sağlar.
- Mantıksal olarak ilişkili veriler, bir kuruluşun bilgilerinin varlıklarını, niteliklerini ve ilişkilerini içerir.

VERITABANI YÖNETIM SISTEMI (DBMS)

- Kullanıcıların veritabanını tanımlamasını, oluşturmasını, sürdürmesini ve veritabanına erişimi kontrol etmesini sağlayan bir yazılım sistemi.
- (Veritabanı) uygulama programı: DBMS'ye uygun bir istek (SQL deyimi) göndererek veritabanı ile etkileşime giren bir bilgisayar programı.

VERITABANI YÖNETIM SISTEMI (DBMS)



PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)
Lease (leaseNo, propertyNo, clientNo, paymentMethod, deposit, paid, rentStart, rentF nish)

VERITABANI YAKLAŞIMI

- Veri tanımlama dili (DDL).
  - Veri türlerinin, yapılarının ve herhangi bir veri kısıtlamasının belirtilmesine izin verir.
  - Tüm spesifikasyonlar veritabanında saklanır.
- Veri manipülasyon dili (DML).
  - Verilerin genel sorgulama olanağı (sorgu dili).

VERITABANI YAKLAŞIMI

- Veritabanına kontrollü erişim şunları içerebilir:
  - bir güvenlik sistemi
  - bir bütünlük sistemi
  - bir eşzamanlılık kontrol sistemi
  - bir kurtarma kontrol sistemi
  - kullanıcı tarafından erişilebilir bir katalog.

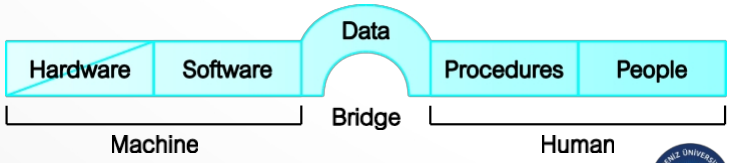
GÖRÜNÜMLER

- Her kullanıcının kendi veritabanı görünümüne sahip olmasını sağlar.
- Görünüm, esasen veritabanının bir alt kümesidir.

GÖRÜŞLER - FAYDALAR

- Karmaşıklığı azaltın
- Bir güvenlik seviyesi sağlayın
- Veritabanının görünümünü özelleştirmek için bir mekanizma sağlayın
- Altta yatan veritabanı değiştirilse bile veritabanının yapısının tutarlı ve değişmeyen bir resmini sunma

DBMS ORTAMININ BİLEŞENLERİ



DBMS ORTAMININ BİLEŞENLERİ

- Donanım
  - Bir PC'den bir bilgisayar ağına kadar değişebilir.
- Yazılım
  - DBMS, işletim sistemi, ağ yazılımı (gerekirse) ve ayrıca uygulama programları.
- Veri
  - Kuruluş tarafından kullanılan ve şema olarak adlandırılan bu verilerin bir açıklaması.
- Prosedürler
  - Veritabanı ve VTYS'nin tasarımı ve kullanımında uygulanması gereken talimatlar ve kurallar.
- İnsanlar

VERITABANI ORTAMINDAKI ROLLER

- Veri Yöneticisi (DA)
- Veritabanı Yöneticisi (DBA)
- Veritabanı Tasarımcıları (Mantıksal ve Fiziksel)
- Uygulama Programcıları
- Son Kullanıcılar (naif ve sofistike)

VERİ TABANI SİSTEMLERİNİN TARİHÇESİ

- Birinci nesil
  - Hiyerarşik ve Ağ
- İkinci nesil
  - İlişkisel
- Üçüncü nesil
  - Nesne İlişkisel
  - Nesne Yönelimli

DBMSS'NİN AVANTAJLARI

- Veri yedekliliğinin kontrolü
- Veri tutarlılığı
- Aynı miktarda veriden daha fazla bilgi
- Veri paylaşımı
- Geliştirilmiş veri bütünlüğü
- Geliştirilmiş güvenlik
- Standartların uygulanması
- Ölçek ekonomisi

DBMSS'NİN AVANTAJLARI

- Çelişen gereksinimleri dengeleyin
- Geliştirilmiş veri erişilebilirliği ve yanıt verebilirliği
- Artan üretkenlik
- Veri bağımsızlığı sayesinde iyileştirilmiş bakım
- Artan eşzamanlılık
- Geliştirilmiş yedekleme ve kurtarma hizmetleri

DBMSS'NİN DEZAVANTAJLARI

- Karmaşıklık
- Boyut
- DBMS Maliyeti
- Ek donanım maliyetleri
- Dönüşüm maliyeti
- Performans
- Bir başarısızlığın daha yüksek etkisi

GÖREV 1

- Seçtiğiniz bir dili kullanarak kurslar için bilgi tutan bir dosyayı işleyeceğiz. Dosya virgüllerle ayrılmış aşağıdaki formata sahiptir
  - Kurs Departmanı
  - Kurs Kodu (numara)
  - Dersin Adı
  - Kurs AKTS
- Uygulamanız gereken işlemler aşağıdaki gibidir:
  - Dosyadaki kursların bir listesini alın (sadece departman ve kod birleştirilmiş olarak)
  - Yeni bir kurs ekleyin (yukarıdaki parametreleri sırayla alır)
  - Bir kursu kaldırır (parametre olarak yalnızca bölüm ve kodunu alır)
  - Bir ders için AKTS'yi değiştirir (bölüm, ders kodu ve yeni AKTS'yi parametre olarak alır)
  - Kurs kodunu değiştirir (departman, mevcut kurs kodu ve yeni kodunu parametre olarak alır)

CSE 204 - VERİ TABANI SİSTEMLERİNE GİRİŞ İLİŞKİSEL MODEL

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- İlişkisel model terminolojisi.
- Verileri temsil etmek için tabloların nasıl kullanıldığı.
- Matematiksel ilişkiler ve ilişkisel modeldeki ilişkiler arasındaki bağlantı.
- Veritabanı ilişkilerinin özellikleri.
- CK, PK ve FK'ler nasıl tanımlanır?
- Varlık bütünlüğü ve referans bütünlüğünün anlamı.
- Görüşlerin amacı ve avantajları.

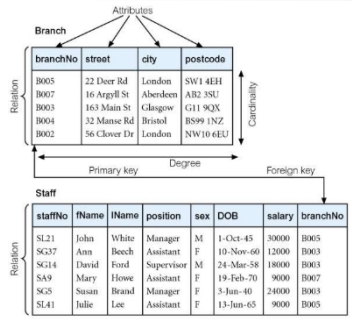
ILIŞKISEL MODEL TERMINOLOJISI

- Bir ilişki, sütunları ve satırları olan bir tablodur.
  - Yalnızca veritabanının mantıksal yapısı için geçerlidir, fiziksel yapısı için geçerli değildir.
- Öznitelik, bir ilişkinin adlandırılmış bir sütunudur.
- Etki alanı, bir veya daha fazla öznitelik için izin verilen değerler kümesidir.

ILIŞKISEL MODEL TERMINOLOJISI

- Tuple, bir ilişkinin bir satırıdır.
- Derece, bir ilişkideki niteliklerin sayısıdır.
- Kardinalite, bir ilişkideki çiftlerin sayısıdır.
- İlişkisel Veritabanı, farklı ilişki adlarına sahip normalleştirilmiş ilişkilerin bir koleksiyonudur.

ŞUBE VE PERSONEL ILIŞKILERI ÖRNEKLERI



branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

ÖZNITELIK ALANLARINA ÖRNEKLER

Attribute	Domain Name	Meaning	Domain Definition
branchNo	BranchNumbers	The set of all possible branch numbers	character: size 4, range B001–B999
street	StreetNames	The set of all street names in Britain	character: size 25
city	CityNames	The set of all city names in Britain	character: size 15
postcode	Postcodes	The set of all postcodes in Britain	character: size 8
sex	Sex	The sex of a person	character: size 1, value M or F
DOB	DatesOfBirth	Possible values of staff birth dates	date, range from 1-Jan-20, format dd-mmm-yy
salary	Salaries	Possible values of staff salaries	monetary: 7 digits, range 6000.00–40000.00

ILIŞKISEL MODEL IÇIN ALTERNATIF TERMINOLOJI

Formal terms	Alternative 1	Alternative 2
Relation	Table	File
Tuple	Row	Record
Attribute	Column	Field

ILIŞKININ MATEMATIKSEL TANIMI

- $D_1 = \{2, 4\}$ ve $D_2 = \{1, 3, 5\}$ olmak üzere iki $D_{(1)}$ & $D_{(2)}$ kümesi düşünün.
- Kartezyen çarpım, $D_1 X D_2$, ilk elemanı D_1'in ve ikinci elemanı $D_{(2)}$'nin üyesi olan tüm sıralı çiftlerin kümesidir.

  $D_1 X D_2 = \{(2, 1), 2, 3), (2, 5), (4, 1), (4, 3), 4, 5)\}$
- Alternatif yol, birincisi $D_{(1)}$'den ve ikincisi D_2'den olan tüm eleman kombinasyonlarını bulmaktır.

ILIŞKININ MATEMATIKSEL TANIMI

- Kartezyen çarpımın herhangi bir alt kümesi bir bağıntıdır; örn.

  R= {(2, 1), (4, 1)}
- Seçim için bazı koşullar kullanarak hangi çiftlerin ilişki içinde olduğunu belirtebilir; örn.
  - ikinci eleman 1'dir:

    R = {(x, y) | x $D_{(1)}$, y $D_{(2)}$ ve y= 1}
  - ilk eleman her zaman ikincinin iki katıdır:

    ∈ ∈

    S = {(x, y) | x ∈ , y ∈ , ve x= 2y}

ILIŞKININ MATEMATIKSEL TANIMI

- Kartezyen Çarpımı ile üç D_1, $D_{(2)}$, D_3 kümesi düşünün

  $D_1 X D X_2 D$; $_3$

Örn eğin . $D_1 = \{1, 3\}$ $D_2 = \{2, 4\}$ $D_3 = \{5, 6\}$

$D_1 X D_{(2)} X D_3 = \{(1,2,5), 1,2,6), (1,4,5), (1,4,6), (3,2,5), (3,2,6), (3,4,5), (3,4,6)\}$

- Bu sıralı üçlülerin herhangi bir alt kümesi bir bağıntıdır.

ILIŞKININ MATEMATIKSEL TANIMI

- *N* kümenin $(D_1, D_2, . . , D_n)$ kartezyen :

- $D_1 X D_2 X . . . X D_n = \{(d_1, d_2, . . . , d_n) \mid d_1 \in D_{(1)}, d_2 \in D_{(2)}, . . . , d_n \in D_{(n)}\}$

  genellikle şu şekilde

  yazılır: n

  XD_i

  i= 1
- Bu Kartezyen çarpımdan elde edilen herhangi bir n-tuple kümesi, n setle

  r.

VERITABANI ILIŞKILERI

- İlişki şeması
  - Bir dizi öznitelik ve etki alanı adı çiftiyle tanımlanan adlandırılmış ilişki.
- İlişkisel veritabanı şeması
  - Her biri ayrı bir isme sahip ilişki şemaları kümesi.

ILIŞKILERIN ÖZELLIKLERI

- İlişki adı, ilişkisel şemadaki diğer tüm ilişki adlarından farklıdır.
- Her bir ilişki hücresi tam olarak bir atomik (tek) değer içerir.
- Her özniteliğin ayrı bir adı vardır.
- Bir özniteliğin değerlerinin tümü aynı etki alanından gelir.

ILIŞKILERIN ÖZELLIKLERI

- Her bir kuple farklıdır; yinelenen yoktur.
- Niteliklerin sırasının hiçbir önemi yoktur.
- Teorik olarak çiftlerin sırasının bir önemi yoktur.

ILIŞKISEL ANAHTARLAR

- Superkey
  - Bir ilişki içindeki bir tuple'ı benzersiz bir şekilde tanımlayan bir öznitelik veya öznitelikler kümesi.
- Aday Anahtarı
  - Superkey (K) öyle ki hiçbir uygun alt küme bağıntı içinde bir superkey değildir.
  - R'nin her bir çiftinde, K değerleri o çiftleri benzersiz bir şekilde tanımlar (teklik).
  - K'nın hiçbir uygun alt kümesi teklik özelliğine (indirgenemezlik) sahip değildir.

ILIŞKISEL ANAHTARLAR

- Birincil Anahtar
  - İlişkideki çiftleri benzersiz bir şekilde tanımlamak için seçilen aday anahtar.
- Alternatif Anahtarlar
  - Birincil anahtar olarak seçilmeyen aday .
- Yabancı Anahtar
  - Bir ilişki içinde, başka bir ilişkinin (muhtemelen aynı) aday anahtarıyla eşleşen öznitelik veya öznitelik kümesi.

BÜTÜNLÜK KISITLAMALARI

- Null
  - Şu anda bilinmeyen veya tuple için geçerli olmayan bir özniteliğin değerini temsil eder.
  - Eksik veya istisnai verilerle ilgilenir.
  - Bir değerin yokluğunu temsil eder ve değer olan sıfır veya boşluklarla aynı değildir.

BÜTÜNLÜK KISITLAMALARI

- Varlık Bütünlüğü
  - Bir temel ilişkide, bir birincil anahtarın hiçbir niteliği boş olamaz.
- Referans Bütünlüğü
  - Bir ilişkide yabancı anahtar varsa, ya yabancı anahtar değeri kendi ana ilişkisindeki bir tuple'ın aday anahtar değeriyle eşleşmelidir ya da yabancı anahtar tamamen boş olmalıdır.

BÜTÜNLÜK KISITLAMALARI

- Genel Kısıtlamalar
  - Kullanıcılar veya veritabanı yöneticileri tarafından belirlenen ve kuruluşun bazı yönlerini tanımlayan veya kısıtlayan ek kurallar.

GÖRÜNÜMLER

- Temel İlişki
  - Kavramsal şemadaki bir varlığa karşılık gelen ve tuple'ları fiziksel olarak veritabanında depolanan isimlendirilmiş ilişki.
- Görünüm
  - Başka bir ilişki üretmek için temel üzerinde çalışan bir veya daha fazla ilişkisel işlemin dinamik sonucu.

GÖRÜNÜMLER

- Veritabanında gerçekte var olması gerekmeyen ancak istek üzerine, istek anında üretilen sanal bir ilişki.
- Bir görünümün içeriği, bir veya daha fazla temel ilişki üzerinde bir sorgu olarak tanımlanır.
- Görünümler dinamiktir, yani görünüm niteliklerini etkileyen temel ilişkilerde yapılan değişiklikler anında görünüme yansıtılır.

GÖRÜNÜMLERIN AMACI

- Veritabanının bazı bölümlerini belirli kullanıcılardan gizleyerek güçlü ve esnek bir güvenlik mekanizması sağlar.
- Kullanıcıların verilere özelleştirilmiş bir şekilde erişmesine izin verir, böylece aynı veriler farklı kullanıcılar tarafından aynı anda farklı şekillerde görülebilir.
- Temel ilişkiler üzerindeki karmaşık işlemleri basitleştirebilir.

GÖRÜNÜMLERİ GÜNCELLEME

- Bir temel ilişkide yapılan tüm güncellemeler, bu temel ilişkiyi referans alan tüm görünümlere hemen yansıtılmalıdır.
- Görünüm güncellenirse, altta yatan temel ilişki değişikliği yansıtmalıdır.

GÖRÜNÜMLERİ GÜNCELLEME

- Görünümler aracılığıyla yapılabilecek değişiklik türlerinde kısıtlamalar vardır:
- Sorgu tek bir temel ilişki içeriyorsa ve temel ilişkinin bir aday anahtarını içeriyorsa güncellemelere izin verilir.
- Birden fazla temel ilişkiyi içeren güncellemelere izin verilmez.
- Toplama veya gruplama işlemlerini içeren güncellemelere izin verilmez.

GÖRÜNÜMLERİ GÜNCELLEME

- Görünüm sınıfları şu şekilde tanımlanır:
  - teorik olarak güncellenemez;
  - teorik olarak güncellenebilir;
  - kısmen güncellenebilir.

CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ SQL

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- SQL'in amacı ve önemi.
- SELECT ve kullanarak veritabanından veri nasıl alınır:
  - Bileşik WHERE koşullarını kullanın.
  - ORDER BY kullanarak sorgu sonuçlarını sıralayın.
  - Toplama işlevlerini kullanın.
  - GROUP BY ve HAVING kullanarak verileri gruplayın.
  - Alt sorguları kullanın.
  - Tabloları .
  - Küme işlemleri gerçekleştirin (UNION, INTERSECT, EXCEPT).
- INSERT, UPDATE ve DELETE kullanarak veritabanı nasıl güncellenir.

SQL'İN AMAÇLARI

- İdeal olarak, veritabanı dili kullanıcının şunları izin vermelidir:
  - veritabanı ve ilişki yapılarını oluşturun;
  - İlişkilerden veri ekleme, değiştirme, silme işlemlerini gerçekleştirir;
  - basit ve karmaşık sorgular gerçekleştirebilir.
- Bu görevleri minimum kullanıcı çabası ile gerçekleştirmeli ve komut yapısı/sözdizimi kolay öğrenilebilir olmalıdır.
- Taşınabilir olmalı.

SQL'İN AMAÇLARI

- SQL, 2 ana bileşeni olan dönüşüm odaklı bir dildir:
  - Veritabanı yapısını tanımlamak için bir DDL.
  - Verileri almak ve güncellemek için bir DML.
- SQL:1999'a kadar SQL kontrol akışı komutları içermiyordu. Bunların bir programlama veya iş kontrol dili kullanılarak ya da kullanıcının kararlarıyla etkileşimli olarak uygulanması gerekiyordu.

SQL'İN AMAÇLARI

- SQL'i öğrenmek nispeten kolaydır:
  - Prosedürel değildir - nasıl elde edeceğinizden ziyade hangi bilgiye ihtiyacınız olduğunu belirtirsiniz;
  - esasen serbest formattır.
  - Standart İngilizce kelimelerden oluşur:

```
1. CREATE TABLE Staff(staffNo VARCHAR(5), lName
VARCHAR(15),
maaş DECIMAL(7,2));
2.     INSERT INTO Staff VALUES ('SG16', 'Brown', 8300);
3. SELECT staffNo, lName, salary FROM
     Personel
     WHERE salary> 10000;
```


SQL'İN AMAÇLARI

- DBA'lar, yönetim, uygulama geliştiricileri ve diğer son kullanıcı türleri dahil olmak üzere çeşitli kullanıcılar tarafından kullanılabilir.
- SQL için artık bir ISO standardı mevcuttur ve bu da onu ilişkisel veritabanları için hem resmi hem de fiili standart dil haline getirmektedir.

SQL TARİHÇESİ

- 1974 yılında D. Chamberlin (IBM San Jose Laboratuvarı) 'Yapılandırılmış İngilizce Sorgu Dili' (SEQUEL) adlı bir dil tanımlamıştır.
- Gözden geçirilmiş bir sürüm olan SEQUEL/2 1976'da tanımlanmış ancak daha sonra yasal nedenlerle adı SQL olarak değiştirilmiştir.
- Resmi telaffuz 'S-Q-L' olmasına rağmen hala 'see-quel' olarak telaffuz edilmektedir.
- IBM daha sonra SEQUEL/2'yi temel alan System R adında bir DBMS prototipi üretmiştir.
- Ancak SQL'in kökleri, System R projesinden önce gelen SQUARE'e (Sorguları İlişkisel İfadeler Olarak Belirleme) dayanmaktadır.

SQL TARİHÇESİ

- 70'lerin sonunda ORACLE ortaya çıktı ve muhtemelen SQL tabanlı ilk ticari RDBMS oldu.
- 1987 yılında ANSI ve ISO SQL için bir ilk standart yayınladı.
- 1989 yılında ISO, "Bütünlük Geliştirme Özelliği "ni tanımlayan bir ek yayınlamıştır.
- 1992 yılında, SQL2 veya SQL/92 olarak adlandırılan ISO standardında ilk büyük revizyon gerçekleşti.
- 1999 yılında, SQL:1999 nesne yönelimli veri yönetimi desteği ile piyasaya sürüldü.
- 2003 yılının sonlarında SQL:2003 piyasaya sürüldü.
- 2008 yazında SQL:2008 piyasaya sürüldü.
- 2011'in sonlarında SQL:2011 piyasaya sürüldü.

SQL'İN ÖNEMİ

- SQL, IBM'in Sistem Uygulama Mimarisi gibi uygulama mimarilerinin bir parçası haline gelmiştir.
- Birçok büyük ve etkili kuruluşun (örneğin X/OPEN) stratejik tercihidir.
- SQL, Amerikan Hükümetine yapılan tüm veri tabanı satışlarında uyulması gereken Federal Bilgi İşleme Standardıdır (FIPS).
- SQL diğer standartlarda kullanılır ve hatta bir tanımlama aracı olarak diğer standartların etkiler. Örnekler şunları içerir:
  - ISO'nun Bilgi Kaynağı Dizin Sistemi (IRDS) Standardı
  - Uzaktan Veri Erişimi (RDA) Standardı.

SQL KOMUTLARI YAZMA

- SQL ifadesi ayrılmış kelimeler ve kullanıcı tanımlı kelimelerden oluşur.
- ❑ Ayrılmış kelimeler SQL'in sabit bir parçasıdır ve tam olarak gerektiği gibi yazılmalıdır ve satırlara bölünemez.
- ❑ Kullanıcı tanımlı kelimeler kullanıcı tarafından oluşturulur ve ilişkiler, sütunlar, görünümler gibi çeşitli veritabanı nesnelerinin adlarını temsil .

SQL KOMUTLARI YAZMA

- Bir SQL deyiminin çoğu bileşeni, değişmez karakter verileri dışında büyük/küçük harfe duyarsızdır.
- Girinti ve çizgilerle daha okunabilir:
  - Her cümle yeni bir satırda başlamalıdır.
  - Bir cümlenin başlangıcı diğer cümlelerin başlangıcı ile aynı hizada olmalıdır.
  - Eğer cümlenin birkaç bölümü varsa, her biri ayrı bir satırda görünmeli ve cümle başlangıcının altında girintili .

SQL KOMUTLARI YAZMA

- BNF gösteriminin genişletilmiş formunu kullanın:
  - ❑ Büyük harfler ayrılmış kelimeleri temsil eder.
  - ❑ Küçük harfler kullanıcı tanımlı kelimeleri temsil eder.
  - ❑ | alternatifler arasında bir seçim yapıldığını gösterir.
  - ❑ Küme parantezleri gerekli bir öğeyi gösterir.
  - ❑ Köşeli parantezler isteğe bağlı bir öğeyi gösterir.
  - ❑ ... isteğe bağlı tekrarı gösterir (0 veya daha fazla).

LİTERALLER

- Değişmezler SQL deyimlerinde kullanılan sabitlerdir.
- Sayısal olmayan tüm değişmezler tek tırnak içine alınmalıdır (örn. 'London').
- Tüm sayısal değişmezler tırnak içine alınmamalıdır (örn. 650.00).

SELECT IFADESI

```
SELECT [DISTINCT | ALL]
      {*| [columnExpression [AS newName]] [,...] } FROM
            TableName [alias] [, ...]
[NEREDE     koşul]
[GROUP BY columnList] [HAVING         condition]
[ORDER BY columnList]
```


SELECT IFADESI

- SEÇİNİZ Çıktıda hangi sütunların görüneceğini belirtir.
- FROMKullanılacak tablo(lar)ı belirtir.
- NEREDE Satırları filtreler.
- GROUP BY Aynı sütun değerine sahip satır gruplarını oluşturur.
- SAHİP OLMAK Filtreler bazı koşullara tabi gruplar.
- ORDER BY Çıktının sırasını belirtir.
- Maddelerin sırası değiştirilemez.
- Yalnızca SELECT ve FROM zorunludur.

ÖRNEK - TÜM SÜTUNLAR, TÜM SATIRLAR

- Tüm personelin tam ayrıntılarını listeleyin.

```
SELECT personelNo, fAdi, lAdi, adres, pozisyon,
            cinsiyet, DOB, maaş, şubeNo
      Personelden;
```

- 'Tüm sütunlar' için kısaltma olarak * kullanabilir:

```
      SELECT *
      Personelden;
```


ÖRNEK - TÜM SÜTUNLAR, TÜM SATIRLAR

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000.00	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000.00	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000.00	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000.00	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000.00	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000.00	B005

ÖRNEK 2 - BELIRLI SÜTUNLAR, TÜM SATIRLAR

- Tüm personel için yalnızca numarası, adı ve soyadı ile maaşı gösteren bir maaş listesi hazırlayın.

```
      SELECT staffNo, fName, lName, salary FROM
      Personel;
```


ÖRNEK 2 - BELIRLI SÜTUNLAR, TÜM SATIRLAR

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG37	Ann	Beech	12000.00
SG14	David	Ford	18000.00
SA9	Mary	Howe	9000.00
SG5	Susan	Brand	24000.00
SL41	Julie	Lee	9000.00

ÖRNEK 3 - FARKLI KULLANIM

- Görüntülenen tüm mülklerin mülk numaralarını listeleyin.

```
      SELECT propertyNo FROM
      Görüntüleme;
```

propertyNo
PA14
PG4
PG4
PA14
PG36

ÖRNEK 3 - FARKLI KULLANIM

- Yinelenenleri ortadan kaldırmak için DISTINCT kullanın:

```
      SELECT
      DISTINCT propertyNo FROM
      Görüntüleme;
```

propertyNo
PA14
PG4
PG36

ÖRNEK 4 - HESAPLANAN ALANLAR

- Tüm personel için personel numarası, adı/soyadı ve gösteren aylık maaş listesini hazırlayın.

```
      SELECT staffNo, fName, lName, salary/12 FROM
      Personel;
```

staffNo	fName	lName	col4
SL21	John	White	2500.00
SG37	Ann	Beech	1000.00
SG14	David	Ford	1500.00
SA9	Mary	Howe	750.00
SG5	Susan	Brand	2000.00
SL41	Julie	Lee	750.00

ÖRNEK 4 - HESAPLANAN ALANLAR

- Sütunu adlandırmak için AS cümlesini kullanın:

```
SELECT staffNo, fName, lName, salary/12 AS monthlySalary
Personelden;
```


ÖRNEK 5 - KARŞILAŞTIRMALI ARAMA

- Maaşı 10.000'den fazla olan tüm personeli listeleyin.

```
SELECT staffNo, fName, lName, position, salary
FROM Personel
WHERE salary>  10000;
```

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG37	Ann	Beech	Assistant	12000.00
SG14	David	Ford	Supervisor	18000.00
SG5	Susan	Brand	Manager	24000.00

ÖRNEK 6 - BİLEŞİK KARŞILAŞTIRMA ARAMASI

- Londra veya Glasgow'daki tüm şubelerin adreslerini listeleyin.

```
SELECT *
Şube'den
WHERE şehir = 'Londra' VEYA şehir = 'Glasgow';
```

branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B003	163 Main St	Glasgow	G11 9QX
B002	56 Clover Dr	London	NW10 6EU

ÖRNEK 7 - ARALIK ARAMASI

- Maaşı 20.000 ile 30.000 arasında olan tüm personeli listeleyin.

```
SELECT staffNo, fName, lName, position, salary FROM Personel
NEREDE maaş 20000 İLE 30000 ARASINDA;
```

- BETWEEN testi aralığın uç noktalarını içerir.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG5	Susan	Brand	Manager	24000.00

ÖRNEK 7 - ARALIK ARAMASI

- Ayrıca olumsuzlanmış bir versiyon NOT BETWEEN.
- BETWEEN SQL'in ifade gücüne fazla bir şey katmaz. Şöyle de yazılabilir:

```
SELECT staffNo, fName, lName, position, salary FROM Personel
WHERE salary>=20000 AND salary <= 30000;
```

- Yine de bir dizi değer için kullanışlıdır.

ÖRNEK 8 - ÜYELİĞİ AYARLA

- Tüm yöneticileri ve amirleri listeleyin.

```
SELECT staffNo, fName, lName, position FROM Personel
WHERE position IN ('Manager', 'Supervisor');
```

staffNo	fName	lName	position
SL21	John	White	Manager
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

ÖRNEK 8 - ÜYELİĞİ AYARLA

- Olumsuzlanmış bir versiyonu vardır (NOT IN).
- IN SQL'in ifade gücüne fazla bir şey katmaz. Bunu şu şekilde ifade edebilirdim:

```
SELECT staffNo, fName, lName, position FROM
    Personel
WHERE position='Manager' OR
      position='Supervisor';
```

- IN, küme çok sayıda değer içerdiğinde daha verimlidir.

ÖRNEK 9 - DESEN EŞLEŞTİRME

- Adreslerinde 'Glasgow' dizesi bulunan tüm sahiplerini bulun.

```
SELECT sahipNo, fName, lName, adres, telNo FROM PrivateOwner
WHERE adres LIKE '%Glasgow%';
```

ownerNo	fName	lName	address	telNo
CO87	Carol	Farrel	6 Achray St, Glasgow G32 9DX	0141-357-7419
CO40	Tina	Murphy	63 Well St, Glasgow G42	0141-943-1728
CO93	Tony	Shaw	12 Park Pl, Glasgow G4 0QR	0141-225-7025

ÖRNEK 9 - DESEN EŞLEŞTİRME

- SQL'de iki özel kalıp eşleştirme sembolü vardır:
  - %: sıfır veya daha fazla karakterden oluşan dizi;
  - _ (alt çizgi): herhangi bir tek karakter.
- LIKE '%Glasgow%', 'Glasgow' içeren herhangi bir uzunluktaki karakter dizisi anlamına gelir.

ÖRNEK 10 - NULL ARAMA KOŞULU

- PG4 numaralı mülkle ilgili yorum yapılmamış tüm görüntülemelerin ayrıntılarını listeleyin.
- PG4 numaralı mülk için biri yorumlu, diğeri yorumsuz olmak üzere 2 görüntüleme vardır.
- IS NULL özel anahtar sözcüğünü kullanarak null için açıkça test etmeniz gerekir:

```
SELECT clientNo, viewDate FROM
Görüntüleme
WHERE propertyNo= 'PG4' AND
                    yorum NULL'dur;
```

ÖRNEK 10 - NULL ARAMA KOŞULU

clientNo	viewDate
CR56	26-May-04

- Negatifleştirilmiş sürüm (IS NOT NULL), null olmayan değerleri test edebilir.

ÖRNEK 11 - SIRALAMA (TEK SÜTUN)

- Tüm personel için maaşları, azalan maaş sırasına göre düzenleyerek listeleyin.

```
SELECT staffNo, fName, lName, salary FROM
Personel
ORDER BY maaş DESC;
```

staffNo	fName	lName	salary
SL21	John	White	30000.00
SG5	Susan	Brand	24000.00
SG14	David	Ford	18000.00
SG37	Ann	Beech	12000.00
SA9	Mary	Howe	9000.00
SL41	Julie	Lee	9000.00

ÖRNEK 12 - SIRALAMA (ÇOKLU SÜTUN)

- Mülk türüne göre sıralanmış kısaltılmış mülk listesi üretin.

```
SELECT mülkNo, tip, odalar, kira FROM
PropertyForRent
ORDER BY tip;
```

propertyNo	type	rooms	rent
PL94	Flat	4	400
PG4	Flat	3	350
PG36	Flat	3	375
PG16	Flat	4	450
PA14	House	6	650
PG21	House	5	600

ÖRNEK 12 - SIRALAMA (ÇOKLU SÜTUN)

- Bu listedeki dört daire - minör sıralama anahtarı belirtilmediği için, sistem bu satırları istediği sıraya göre düzenler.
- Kira sırasına göre düzenlemek için küçük sıra belirtin:

```
SELECT mülkNo, tip, odalar, kira FROM
PropertyForRent
ORDER BY tip, kira DESC;
```

propertyNo	type	rooms	rent
PG16	Flat	4	450
PL94	Flat	4	400
PG36	Flat	3	375
PG4	Flat	3	350
PA14	House	6	650
PG21	House	5	600

SELECT DEYIM TOPLAMLARI

- ISO standardı beş toplam fonksiyon tanımlar:
  - COUNT, belirtilen sütundaki değerlerin sayısını döndürür.
  - SUM belirtilen sütundaki değerlerin toplamını döndürür.
  - AVG belirtilen sütundaki değerlerin ortalamasını döndürür.
  - MIN belirtilen sütundaki en küçük değeri döndürür.
  - MAX belirtilen sütundaki en büyük değeri döndürür.

SELECT DEYIM TOPLAMLARI

- Her biri bir tablonun tek bir sütunu üzerinde çalışır ve tek bir değer döndürür.
- COUNT, MIN ve MAX sayısal ve sayısal olmayan alanlar için geçerlidir, ancak SUM ve AVG yalnızca sayısal alanlarda kullanılabilir.
- COUNT(*) dışında, her fonksiyon önce null'ları eleyip yalnızca kalan null olmayan değerler üzerinde çalışır.
- COUNT(*), boş veya değerler olup olmadığına bakmaksızın bir tablonun tüm satırlarını sayar.
- ortadan kaldırmak için sütun adından önce DISTINCT kullanabilir.
- DISTINCT'in MIN/MAX ile bir etkisi yoktur, ancak SUM/AVG ile olabilir.

SELECT DEYIM TOPLAMLARI

- Toplama işlevleri yalnızca SELECT listesinde ve HAVING cümlesinde kullanılabilir.
- SELECT listesi bir toplama işlevi içeriyorsa ve GROUP BY cümlesi yoksa, SELECT listesi bir toplama işleviyle bir sütuna başvuramaz. Örneğin, aşağıdakiler yasa dışıdır:

```
SELECT personelNo, COUNT(maaş) FROM
Personel;
```


ÖRNEK 13 - COUNT(*) KULLANIMI

- Kaç mülkün kirası aylık 350'den fazla tutuyor?

```
SELECT COUNT(*) AS myCount
    FROM PropertyForRent
    WHERE rent > 350;
```

myCount
5

ÖRNEK 14 - COUNT(DISTINCT) KULLANIMI

- Mayıs '13'te kaç farklı mülk görüntülendi?

```
SELECT COUNT(DISTINCT propertyNo) AS myCount FROM Görüntüleme
WHERE viewDate BETWEEN '1-May-13' VE '31-Mayıs-13';
```

myCount
2

ÖRNEK 16 - MIN, MAX, AVG KULLANIMI

- Minimum, maksimum ve ortalama personel maaşını bulun.

```
SELECT MIN(salary) AS myMin,
       MAX(salary) AS myMax,
       AVG(salary) AS myAvg
Personelden;
```

myMin	myMax	myAvg
9000.00	30000.00	17000.00

SELECT DEYIMI GRUPLAMA

- Alt toplamları almak için GROUP BY cümlesini kullanın.
- SELECT ve GROUP BY yakından entegre edilmiştir: SELECT listesindeki her öğe grup başına tek değerli olmalıdır ve SELECT cümlesi yalnızca şunları :
  - sütun adları
  - toplu fonksiyonlar
  - sabitler
  - Yukarıdakilerin kombinasyonlarını içeren ifade.

SELECT DEYIMI GRUPLAMA

- SELECT listesindeki tüm sütun adları, ad yalnızca bir toplama işlevinde kullanılmadığı sürece GROUP BY görünmelidir.
- WHERE, GROUP BY ile birlikte kullanılırsa, önce WHERE uygulanır, ardından yüklemi karşılayan kalan satırlardan gruplar oluşturulur.
- ISO, GROUP BY amaçları doğrultusunda iki null değerini eşit olarak kabul eder.

ÖRNEK 17 - GROUP BY KULLANIMI

- Her bir şubedeki personel sayısını ve toplam maaşlarını bulun.

```
SELECT şubeNo,
       COUNT(staffNo) AS myCount,
       SUM(salary) AS mySum
Personelden
GROUP BY şubeNo ORDER
BY şubeNo;
```

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00
B007	1	9000.00

KISITLI GRUPLANDIRMALAR - SAHIP OLMA MADDESI

- HAVING cümlesi, nihai sonuç tablosunda görünen grupları kısıtlamak için GROUP BY ile birlikte kullanılmak üzere tasarlanmıştır.
- WHERE'ye benzer, ancak WHERE tek tek satırları filtrelerken HAVING grupları filtreler.
- HAVING cümlesindeki sütun adları GROUP BY listesinde de görünmeli veya bir toplama işlevi içinde yer almalıdır.

ÖRNEK 18 - SAHIP OLMANIN KULLANIMI

- Birden fazla personeli olan her şube için, her şubedeki sayısını ve maaşlarının toplamını bulun.

```
SELECT şubeNo,
       COUNT(staffNo) AS myCount, SUM(salary)
 AS mySum
Personelden
GROUP BY şubeNo
HAVING COUNT(personelNo)> 1
ORDER BY şubeNo;
```

branchNo	myCount	mySum
B003	3	54000.00
B005	2	39000.00

SUBQUERIES

- Bazı SQL deyimleri, içgömülü bir SELECT içerebilir.
- Bir alt seçim, bir dış SELECT'in WHERE ve HAVING cümlelerinde kullanılabilir; burada alt sorgu veya iç içe sorgu olarak adlandırılır.
- Alt seçimler INSERT, UPDATE ve DELETE deyimlerinde de görünebilir.

ÖRNEK 19 - EŞİTLİK İLE ALT SORGU

- '163 Main St' adresindeki şubede çalışan personeli listeleyin.

```
SELECT staffNo, fName, lName, position FROM Personel
NEREDE şubeNo =
        (SELECT şubeNo FROM
         Şube
         WHERE sokak = '163 Main St');
```

ÖRNEK 19 - EŞİTLİK İLE ALT SORGU

- İç SELECT, '163 Main St' ('B003') adresindeki şube için şube numarasını bulur.
- Outer SELECT daha sonra bu şubede çalışan tüm personelin ayrıntılarını alır.
- Dış SEÇİM o zaman olur:

```
SELECT staffNo, fName, lName, position FROM
Personel
WHERE şubeNo = 'B003';
```

ÖRNEK 19 - EŞİTLİK İLE ALT SORGU

staffNo	fName	lName	position
SG37	Ann	Beech	Assistant
SG14	David	Ford	Supervisor
SG5	Susan	Brand	Manager

ÖRNEK 20 - AGGREGATE İLE ALT SORGU

- Maaşı ortalama maaştan fazla olan tüm personeli listeleyin ve ne kadar fazla olduğunu gösterin.

```
SELECT staffNo, fName, lName, position, salary -
  (SELECT AVG(salary) FROM Staff) As
Personelden
SalDiff
WHERE maaş > (SELECT
AVG(maaş)
 Personelden);
```

ÖRNEK 20 - AGGREGATE İLE ALT SORGU

- 'WHERE salary> AVG(salary)' yazılamıyor
- Bunun yerine, ortalama maaşı (17000) bulmak için alt sorgu kullanın ve ardından maaşı bundan daha yüksek olan personeli bulmak için dış SELECT kullanın:

```
SELECT staffNo, fName, lName, position, salary -
     17000 As salDiff
Personelden
NEREDE maaş>  17000;
```

staffNo	fName	lName	position	salDiff
SL21	John	White	Manager	13000.00
SG14	David	Ford	Supervisor	1000.00
SG5	Susan	Brand	Manager	7000.00

SUBQUERY KURALLARI

- ORDER BY cümlesi bir alt sorguda kullanılamaz (en dıştaki SELECT'te kullanılabilmesine rağmen).
- EXISTS kullanan alt sorgular hariç, alt sorgu SELECT listesi tek bir sütun adı veya ifadeden oluşmalıdır.
- Varsayılan olarak, sütun adları alt sorgunun FROM tablo adına başvurur. Bir takma ad kullanarak FROM içindeki bir tabloya başvurabilir.
- Alt sorgu bir karşılaştırmada işlenen olduğunda, alt sorgu sağ tarafta görünmelidir.
- Bir alt sorgu, bir ifadede işlenen olarak kullanılamaz.

ÖRNEK 21 - İÇİÇE GEÇMİŞ ALT SORGU: IN KULLANIMI

- '163 Main St' adresindeki personel tarafından idare edilen mülkleri listeleyin.

```
SELECT mülkNo, sokak, şehir, posta kodu, tür, odalar, kira FROM
PropertyForRent
WHERE personelNo IN
(SELECT personelNo
 FROM Personel
 WHERE =
(SELECT şubeNo FROM
 Şube
 WHERE sokak = '163 Main St'));
```

propertyNo	street	city	postcode	type	rooms	rent
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375
PG21	18 Dale Rd	Glasgow	G12	House	5	600

HERKES VE HERKES

- ANY ve ALL, tek bir sayı sütunu üreten alt sorgularla kullanılabilir.
- ALL ile, koşul yalnızca alt sorgu tarafından üretilen tüm değerler tarafından karşılanırsa doğru olur.
- ANY ile koşul, alt sorgu tarafından üretilen herhangi bir değer tarafından karşılanırsa doğru .
- Alt sorgu boşsa, ALL doğru, ANY yanlış döndürür.
- ANY yerine SOME kullanılabilir.

ÖRNEK 22 - ANY/SOME KULLANIMI

- Maaşı B003 şubesinde çalışan en az bir personelin maaşından daha fazla olan personeli bulun.

```
SELECT staffNo, fName, lName, position, salary FROM
Personel
NEREDE maaş BAZI>
          (SELECT salary
           FROM Personel
           WHERE branchNo=  'B003');
```

ÖRNEK 22 - ANY/SOME KULLANIMI

- İç sorgu {12000, 18000, 24000} üretir ve dış sorgu, maaşları bu kümedeki herhangi bir değerden büyük olan personeli seçer.

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00
SG14	David	Ford	Supervisor	18000.00
SG5	Susan	Brand	Manager	24000.00

ÖRNEK 23 - TÜMÜNÜN KULLANIMI

- Maaşı B003 şubesinde çalışan tüm personelin maaşından daha fazla olan bulun.

```
SELECT staffNo, fName, lName, position, salary FROM
Personel
WHERE salary > ALL (SELECT
      salary
       Personelden
       WHERE branchNo=  'B003');
```

staffNo	fName	lName	position	salary
SL21	John	White	Manager	30000.00

ÇOKLU TABLO SORGULARI

- Sonuç sütunlarının aynı tablodan gelmesi koşuluyla alt sorguları kullanabilir.
- Sonuç sütunları birden fazla tablodan geliyorsa bir birleştirme kullanılmalıdır.
- Birleştirme yapmak için FROM cümlesine birden fazla tablo ekleyin.
- Ayırıcı olarak virgül kullanın ve genellikle birleştirme sütun(lar)ını belirtmek için WHERE cümlesini ekleyin.

ÇOKLU TABLO SORGULARI

- FROM cümlesinde adlandırılan bir tablo için bir takma ad kullanmak da mümkündür.
- Takma ad, tablo adından bir boşlukla ayrılır.
- Takma adlar, belirsizlik olduğunda sütun adlarını nitelemek için kullanılabilir.

ÖRNEK 24 - BASİT BİRLEŞTİRME

- Bir mülkü inceleyen tüm müşterilerin adlarını, verilen herhangi bir yorumla listeleyin.

```
SELECT c.clientNo, fName, lName,
              propertyNo, comment FROM
Client c, Viewing v
WHERE c.clientNo = v.clientNo;
```


ÖRNEK 24 - BASİT BİRLEŞTİRME

- Her iki tablodan yalnızca clientNo sütunlarında aynı değerlere sahip olan satırlar (c.clientNo= v.clientNo) sonuca dahil edilir.
- İlişkisel cebirdeki equi-join'e eşdeğerdir.

clientNo	fName	lName	propertyNo	comment
CR56	Aline	Stewart	PG36	
CR56	Aline	Stewart	PA14	too small
CR56	Aline	Stewart	PG4	
CR62	Mary	Tregear	PA14	no dining room
CR76	John	Kay	PG4	too remote

ALTERNATIF BIRLEŞTIRME YAPILARI

- SQL, birleştirmeleri belirtmek için alternatif yollar sunar:

```
FROM Client c JOIN Viewing v ON c.clientNo = v.clientNo FROM
Client JOIN Viewing USING clientNo
FROM Müşteri NATURAL JOIN Görüntüleme
```

- Her durumda, FROM orijinal FROM ve WHERE'nin yerini alır. Ancak, ilki iki aynı clientNo sütununa sahip tablo üretir.

ÖRNEK 25 - BIRLEŞTIRME SIRALAMA

- Her şube için, mülkleri yöneten personelin ve yönettikleri mülklerin numaralarını ve adlarını listeleyin.

```
SELECT s.şubeNo, s.personelNo, fAdi, lAdi,
              mülkHayır
FROM Personel s, PropertyForRent p
WHERE s.personelNo = p.personelNo
ORDER BY s.şubeNo, s.personelNo, mülkNo;
```


ÖRNEK 25 - BIRLEŞTIRME SIRALAMA

branchNo	staffNo	fName	lName	propertyNo
B003	SG14	David	Ford	PG16
B003	SG37	Ann	Beech	PG21
B003	SG37	Ann	Beech	PG36
B005	SL41	Julie	Lee	PL94
B007	SA9	Mary	Howe	PA14

ÖRNEK 26 - ÜÇ TABLO BİRLEŞTİRME

- Her şube için, şubenin bulunduğu şehir ve yönettikleri mülkler de dahil olmak üzere mülkleri yöneten personeli listeleyin.

```
SELECT b.şubeNo, b.şehir, s.personelNo, fAdi, lAdi, mülkNo
FROM Şube b, Personel s, EmlakKirası p WHERE
b.şubeNo = s.şubeNo AND
       s.personelNo = p.personelNo
ORDER BY b.şubeNo, s.personelNo, mülkNo;
```

ÖRNEK 26 - ÜÇ TABLO BİRLEŞTİRME

branchNo	city	staffNo	fName	lName	propertyNo
B003	Glasgow	SG14	David	Ford	PG16
B003	Glasgow	SG37	Ann	Beech	PG21
B003	Glasgow	SG37	Ann	Beech	PG36
B005	London	SL41	Julie	Lee	PL94
B007	Aberdeen	SA9	Mary	Howe	PA14

- FROM ve WHERE için alternatif formülasyon:
- FROM (Şube b JOIN Personel s USING şubeNo) AS
- bs JOIN PropertyForRent p USING staffNo

ÖRNEK 27 - ÇOKLU GRUPLAMA SÜTUNLARI

- Her personel tarafından ele alınan mülk sayısını bulun.

```
SELECT s.branchNo, s.staffNo, COUNT(*) AS myCount
FROM Staff s, PropertyForRent p
WHERE s.staffNo = p.staffNo GROUP
BY s.branchNo, s.staffNo ORDER BY
.branchNo, .staffNo;
```


ÖRNEK 27 - ÇOKLU GRUPLAMA SÜTUNLARI

branchNo	staffNo	myCount
B003	SG14	1
B003	SG37	2
B005	SL41	1
B007	SA9	1

BIRLEŞTIRME HESAPLAMA

- Birleştirmenin sonuçlarını oluşturma prosedürü şunlardır:

1. FROM cümlesinde adı geçen tabloların Kartezyen çarpımını oluşturur.
2. WHERE cümlesi varsa, arama koşulunu ürün tablosunun her satırına uygulayın ve koşulu karşılayan satırları saklayın.
3. Kalan her satır için, sonuç tablosunda tek bir satır oluşturmak üzere SELECT listesindeki her öğenin değerini belirleyin.
4. DISTINCT belirtilmişse, sonuç tablosundaki tüm yinelenen satırları ortadan kaldırın.
5. ORDER BY cümlesi varsa, sonuç tablosunu gerektiği gibi sıralayın.
6. SQL, Kartezyen çarpım için özel SELECT biçimi sağlar:

```
SEÇİNİZ     [DISTINCT | ALL]     {*| columnList}
FROM Tablo1 CROSS JOIN Tablo2
```


OUTER JOIN

- Birleştirilmiş bir tablonun bir satırı eşleşmezse, satır sonuç tablosundan çıkarılır.
- Dış birleştirme işlemleri, birleştirme koşulunu karşılamayan satırları tutar.
- Aşağıdaki tabloları dikkate alın:

Branch1

branchNo	bCity
B003	Glasgow
B004	Bristol
B002	London

PropertyForRent1

propertyNo	pCity
PA14	Aberdeen
PL94	London
PG4	Glasgow

OUTER JOIN

- Bu iki tablonun (iç) birleşimi:

```
SELECT b.*, p.*
FROM Branch1 b, PropertyForRent1 p WHERE
b.bCity = p.pCity;
```

branchNo	bCity	propertyNo	pCity
B003	Glasgow	PG4	Glasgow
B002	London	PL94	London

OUTER JOIN

- Sonuç tablosunda şehirlerin aynı olduğu iki satır vardır.
- Bristol ve Aberdeen'deki şubelere karşılık gelen herhangi bir satır bulunmamaktadır.
- Eşleşmeyen satırları sonuç tablosuna dahil etmek için bir Outer join kullanın.

ÖRNEK 28 - SOL DIŞ BIRLEŞTIRME

- Aynı şehirde bulunan şubeleri ve mülkleri, eşleşmeyen şubelerle birlikte .

```
SELECT b.*, p.*
FROM Şube1 b LEFT JOIN
     PropertyForRent1 p ON b.bCity = p.pCity;
```

- Birinci (sol) tablonun ikinci (sağ) tablodaki satırlarla eşleşmeyen satırlarını içerir.
- İkinci tablodaki sütunlar NULL'lar ile doldurulur.

ÖRNEK 28 - SOL DIŞ BIRLEŞTIRME

branchNo	bCity	propertyNo	pCity
B003	Glasgow	PG4	Glasgow
B004	Bristol	NULL	NULL
B002	London	PL94	London

ÖRNEK 29 - SAĞ DIŞ BIRLEŞTIRME

- Aynı şehirdeki şubeleri ve mülkleri ve eşleşmeyen mülkleri listeleyin.

```
SELECT b.*, p.*
FROM Şube1 b SAĞ JOIN
      PropertyForRent1 p ON b.bCity=  p.pCity;
```

- Sağ Dış birleştirme, birinci (sol) tablodaki satırlarla eşleşmeyen ikinci (sağ) tablodaki satırları içerir.
- İlk tablodaki sütunlar NULL'lar ile doldurulur.

ÖRNEK 29 - SAĞ DIŞ BIRLEŞTIRME

branchNo	bCity	propertyNo	pCity
NULL	NULL	PA14	Aberdeen
B003	Glasgow	PG4	Glasgow
B002	London	PL94	London

ÖRNEK 30 - TAM DIŞ BIRLEŞTIRME

- Aynı şehirdeki şubeleri ve mülkleri ve eşleşmeyen şubeleri veya mülkleri listeleyin.

```
SELECT b.*, p.*
FROM Şube1 b FULL JOIN
      PropertyForRent1 p ON b.bCity = p.pCity;
```

- Her iki tabloda da eşleşmeyen satırları içerir.
- Eşleşmeyen sütunlar NULL'lar ile doldurulur.

ÖRNEK 30 - TAM DIŞ BIRLEŞTIRME

branchNo	bCity	propertyNo	pCity
NULL	NULL	PA14	Aberdeen
B003	Glasgow	PG4	Glasgow
B004	Bristol	NULL	NULL
B002	London	PL94	London

VAR VE YOK

- EXISTS ve NOT EXISTS yalnızca alt sorgularla kullanım içindir.
- Basit bir doğru/yanlış sonucu üretin.
- Yalnızca ve yalnızca alt sorgu tarafından döndürülen sonuç tablosunda en az bir satır varsa doğrudur.
- Alt sorgu boş bir sonuç tablosu döndürürse false.
- NOT EXISTS, EXISTS'in zıttıdır.
- (NOT) EXISTS yalnızca alt sorgu sonuç tablosundaki satırların varlığını veya yokluğunu kontrol ettiğinden, alt sorgu herhangi bir sayıda sütun içerebilir.
- (NOT) EXISTS'i takip eden alt sorguların formda olması yaygındır:
- `(SELECT * ...)`

ÖRNEK 31 - EXISTS KULLANARAK SORGU

- Londra şubesinde çalışan tüm personeli bulun.

```
SELECT staffNo, fName, lName, position
FROM Personel s
WHERE EXISTS
      (SELECT *
       FROM Şube b
       WHERE s.branchNo= b.branchNo AND city =
                 'London');
```

staffNo	fName	lName	position
SL21	John	White	Manager
SL41	Julie	Lee	Assistant

ÖRNEK 31 - EXISTS KULLANARAK SORGU

- Not: s.branchNo= b.branchNo arama koşulu, her personel için doğru şube kaydını dikkate almak için gereklidir.
- Atlanırsa, alt sorgu nedeniyle tüm personel kayıtları listelenir:

```
SELECT * FROM Şube WHERE şehir='Londra'
```

- her zaman doğru olur ve sorgu :

```
SELECT staffNo, fName, lName, position FROM Staff WHERE
true;
```


ÖRNEK 31 - EXISTS KULLANARAK SORGU

- Bu sorguyu birleştirme yapısını kullanarak da yazabilirsiniz: `SELECT personelNo, fAdi, lAdi, pozisyon FROM Personel s, Şube b WHERE s.şubeNo = b.şubeNo AND şehir= 'Londra';`

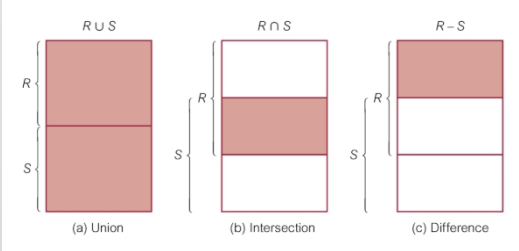
BIRLEŞIM, KESIŞIM VE FARK (HARIÇ)

- İki veya daha fazla sorgunun sonuçlarını tek bir sonuç tablosunda birleştirmek için normal küme işlemleri olan Union, Intersection ve Difference'ı kullanabilir.
- A ve B olmak üzere iki tablonun birleşimi, A veya B'deki ya da her ikisindeki tüm satırları içeren tablodur.
- Kesişim, hem A hem de B için ortak olan tüm satırları içeren tablodur.
- Fark, A'daki tüm satırları içeren ancak B'de bulunmayan tablodur.
- İki tablo birlik uyumlu olmalıdır.

BIRLEŞIM, KESIŞIM VE FARK (HARIÇ)

- Her durumda set operatörü cümlesinin formatı şöyledir:
- `op [ALL] [CORRESPONDING BY {column1 [, ...]}]`
- CORRESPONDING BY belirtilmişse, set işlemi adı geçen sütun(lar) üzerinde gerçekleştirilir.
- CORRESPONDING belirtilmiş ancak BY cümlesi kullanılmamışsa, işlem ortak sütunlar üzerinde gerçekleştirilir.
- ALL belirtilirse, sonuç yinelenen satırları içerebilir.

BIRLEŞIM, KESIŞIM VE FARK (HARIÇ)



ÖRNEK 32 - BİRLİK KULLANIMI

- Bir şubenin ya da mülkün bulunduğu tüm şehirleri listeleyin.

```
(SELECT şehir
FROM Şube
WHERE city IS NOT NULL) UNION
(ŞEHİR SEÇİNİZ
FROM PropertyForRent WHERE
city IS NOT NULL);
```


ÖRNEK 32 - BİRLİK KULLANIMI

- Ya da

```
(SELECT *
Şube'den
WHERE city IS NOT NULL) UNION
CORRESPONDING BY city (SELECT
*
FROM PropertyForRent WHERE
city IS NOT NULL);
```


ÖRNEK 32 - BİRLİK KULLANIMI

- Her iki sorgudan sonuç tabloları üretir ve her iki birleştirir.

city
London
Glasgow
Aberdeen
Bristol

ÖRNEK 33 - INTERSECT KULLANIMI

- Hem bir şubenin hem de bir mülkün bulunduğu tüm şehirleri listeleyin.

```
(SELECT şehir FROM Şube) INTERSECT
(SELECT city FROM PropertyForRent);
```


ÖRNEK 33 - INTERSECT KULLANIMI

- Ya da

```
(SELECT * FROM Şube)
Şehir bazında INTERSECT yazışmaları
(SELECT * FROM PropertyForRent);
```

city
Aberdeen
Glasgow
London

ÖRNEK 33 - INTERSECT KULLANIMI

- Bu sorgu INTERSECT operatörü olmadan yeniden yazılabilir:

```
SELECT b.şehir
FROM Şube b PropertyForRent p WHERE
b.city = p.city;
```

- Ya da:

```
SELECT DISTINCT city FROM Branch b
WHERE EXISTS
(SELECT * FROM PropertyForRent p
WHERE p.city = b.city);
```


ÖRNEK 34 - EXCEPT KULLANIMI

- Şube ofisi bulunan ancak mülk bulunmayan tüm şehirlerin listesi.

```
(SELECT city FROM Branch)
EXCEPT
(SELECT city FROM PropertyForRent);
```

- Ya da

```
(SELECT * FROM Şube)
Şehir tarafından YAZIŞMA HARİÇ
(SELECT * FROM PropertyForRent);
```

city
Bristol

ÖRNEK 34 - EXCEPT KULLANIMI

- Bu sorguyu EXCEPT olmadan yeniden yazabilir:

```
SELECT DISTINCT city FROM Branch
WHERE city NOT IN
(SELECT city FROM PropertyForRent);
```

- Ya da

```
SELECT DISTINCT city FROM Branch b
WHERE NOT EXISTS
(SELECT * FROM PropertyForRent p WHERE
p.city = b.city);
```


INSERT

```
INSERT INTO TableName [ (columnList) ] VALUES
(dataValueList)
```

- columnList isteğe bağlıdır; atlanırsa SQL, tüm orijinal CREATE TABLE sırasına göre bir listesini varsayar.
- Sütun oluşturulurken DEFAULT belirtilmediği sürece, atlanan sütunların tablo oluşturulurken NULL olarak bildirilmiş olması gerekir.
- dataValueList aşağıdaki gibi columnList ile eşleşmelidir:
  - Her listedeki öğe sayısı aynı olmalıdır;
  - iki listedeki öğelerin konumlarında doğrudan bir karşılık olmalıdır;
  - dataValueList'teki her bir öğenin veri türü, ilgili sütunun veri türüyle uyumlu olmalıdır.

INSERT

```
INSERT INTO TableName [ (columnList) ] VALUES
(dataValueList)
```

- columnList isteğe bağlıdır; atlanırsa SQL, tüm sütunların orijinal CREATE TABLE sırasına göre bir listesini varsayar.
- Sütun oluşturulurken DEFAULT belirtilmediği sürece, atlanan sütunların tablo oluşturulurken NULL olarak bildirilmiş olması gerekir.

ÖRNEK 35 - INSERT INTO ... VALUES

- Personel tablosuna tüm sütunlar için veri sağlayan yeni bir satır ekleyin

```
INSERT INTO Personel
VALUES ('SG16', 'Alan', 'Brown', 'Assistant', 'M',
Date'1957-05-25', 8300, 'B003');
```


ÖRNEK 36 - VARSAYILANLARI KULLANARAK EKLEME

- Personel tablosuna tüm zorunlu sütunlar için veri sağlayan yeni bir satır ekleyin.

```
INSERT INTO Personel (personelNo, fAdi, lAdi,
                      position, salary, branchNo) VALUES
('SG44', 'Anne', 'Jones',
                      'Asistan', 8100, 'B003');
```

- Ya da

```
INSERT INTO Personel
VALUES ('SG44', 'Anne', 'Jones', 'Asistan', NULL,
                        NULL, 8100, 'B003');
```


EKLE... SEÇ

- INSERT'in ikinci biçimi, birden fazla satırın bir veya daha fazla tablodan diğerine kopyalanmasını sağlar:

```
INSERT INTO TabloAdı [ (sütunListesi) ]
SELECT ...
```


ÖRNEK 37 - EKLE ... SEÇ

- Personelin adlarını ve yönettikleri mülk sayısını içeren bir StaffPropCount tablosu olduğunu varsayın:
- `StaffPropCount(staffNo, fName, lName, propCnt)`
- Staff ve PropertyForRent tablolarını kullanarak StaffPropCount'u doldurun.

ÖRNEK 37 - EKLE ... SEÇ

```
INSERT INTO StaffPropCount
(SELECT s.staffNo, fName, lName, COUNT(*) FROM
Staff s, PropertyForRent p
WHERE s.personelNo= p.personelNo
GROUP BY s.staffNo, fName, lName) UNION
(SELECT staffNo, fName, lName, 0 FROM
Personel
WHERE personelNo NOT IN
(SELECT DISTINCT personelNo
     FROM PropertyForRent));
```


ÖRNEK 37 - EKLE ... SEÇ

staffNo	fName	lName	propCount
SG14	David	Ford	1
SL21	John	White	0
SG37	Ann	Beech	2
SA9	Mary	Howe	1
SG5	Susan	Brand	0
SL41	Julie	Lee	1

- UNION'ın ikinci kısmı atlanırsa, şu anda herhangi bir mülkü yönetmeyen personel hariç tutulur.

GÜNCELLEME

```
UPDATE TabloAdı
SET columnName1 = dataValue1
          [, columnName2= dataValue2...] [WHERE
searchCondition]
```

- TableName bir temel tablonun veya güncellenebilir bir görünümün adı olabilir.
- SET cümlesi, güncellenecek bir veya daha fazla sütunun adını belirtir.

GÜNCELLEME

- WHERE cümlesi isteğe bağlıdır:
  - atlanırsa, adlandırılmış sütunlar tablodaki tüm satırlar için güncellenir;
  - belirtilirse, yalnızca searchCondition öğesini karşılayan satırlar güncellenir.
- Yeni dataValue(lar) ilgili sütunun veri tipiyle uyumlu olmalıdır.

ÖRNEK 38/39 - GÜNCELLEME

- Tüm personele %3 maaş zammı verin.

```
GÜNCEL Personel
SET maaş = maaş * 1,03;
```

- Tüm Müdürlere %5 maaş artışı verin.

```
GÜNCEL Personel
SET salary = salary * 1.05
WHERE position= 'Manager';
```


ÖRNEK 40 - GÜNCELLEME

- David Ford'u (staffNo='SG14') Müdür olarak terfi ettirin ve maaşını 18.000 £ olarak değiştirin.

```
GÜNCEL Personel
SET pozisyon= 'Müdür', maaş= 18000 WHERE
personelNo = 'SG14';
```


SİL

```
DELETE FROM TabloAdı [WHERE
searchCondition]
```

- TableName bir temel tablonun veya güncellenebilir bir görünümün adı olabilir.
- searchCondition isteğe bağlıdır; atlanırsa, tablodan tüm satırlar silinir. Bu işlem tabloyu silmez. search_condition belirtilirse, yalnızca koşulu karşılayan satırlar silinir.

ÖRNEK 41/42 - SİLİN

- PG4 mülkü ile ilgili tüm görüntülemeleri silin.

```
Görüntüleme'den DELETE
WHERE propertyNo = 'PG4';
```

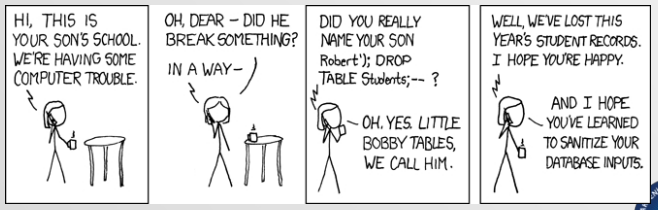
- Görüntüleme tablosundaki tüm kayıtları silin.

```
Görüntülemeden Sil;
```


CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ İLERİ SQL

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

GÜNÜN SÖZÜ



ANA HATLAR

- SQL standardı tarafından desteklenen veri türleri.
- SQL'in bütünlük geliştirme özelliğinin amacı.
- SQL kullanarak bütünlük kısıtlamaları nasıl tanımlanır.
- CREATE ve ALTER TABLE deyimlerinde bütünlük geliştirme özelliği nasıl kullanılır?
- Görünümler
  - Görüşlerin amacı.
  - SQL kullanarak görünümler nasıl oluşturulur ve silinir.
  - DBMS'nin görünümler üzerinde nasıl işlem yaptığı.
  - Görünümlerin hangi koşullar altında güncellenebilir olduğu.
  - Görüşlerin avantajları ve dezavantajları.
- ISO işlem modeli nasıl çalışır?
- GRANT ve REVOKE deyimlerinin bir güvenlik seviyesi olarak nasıl kullanılacağı.

ISO SQL VERİ TÜRLERİ

DATA TYPE	DECLARATIONS				
boolean	BOOLEAN				
character	CHAR	VARCHAR			
bit†	BIT	BIT VARYING			
exact numeric	NUMERIC	DECIMAL	INTEGER	SMALLINT	BIGINT
approximate numeric	FLOAT	REAL	DOUBLE PRECISION		
datetime	DATE	TIME	TIMESTAMP		
interval	INTERVAL				
large objects	CHARACTER LARGE OBJECT		BINARY LARGE OBJECT		

†BIT and BIT VARYING have been removed from the SQL:2003 standard.

BÜTÜNLÜK GELIŞTIRME ÖZELLIĞI

- Beş tür bütünlük kısıtlaması düşünün:
  - gerekli veriler
  - alan kısıtlamaları
  - varlık bütünlüğü
  - referans bütünlüğü
  - genel kısıtlamalar.

BÜTÜNLÜK GELIŞTIRME ÖZELLIĞI

- Gerekli Veriler

```
pozisyon VARCHAR(10)        NOT NULL
```

- Etki Alanı Kısıtlamaları
- KONTROL ET

```
sex CHAR NOT NULL
        CHECK (cinsiyet IN ('M', 'F'))
```


BÜTÜNLÜK GELIŞTIRME ÖZELLIĞI

- DOMAİN OLUŞTUR

```
CREATE DOMAIN DomainName [AS] dataType
    [DEFAULT defaultOption]
    [CHECK (searchCondition)]
```

- Örneğin:

```
CREATE DOMAIN SexType AS CHAR CHECK
    (VALUE IN ('M', 'F'));
sex SexType      NOT NULL
```


BÜTÜNLÜK GELIŞTIRME ÖZELLIĞI

- searchCondition bir tablo araması içerebilir:

```
CREATE DOMAIN BranchNo AS CHAR(4)
CHECK (VALUE IN (SELECT branchNo
                 FROM Şube));
```

- Etki alanları DROP DOMAIN kullanılarak kaldırılabilir:

```
DROP DOMAIN DomainName [RESTRICT|
    CASCADE]
```


IEF - VARLIK BÜTÜNLÜĞÜ

- Bir tablonun birincil anahtarı, her satır için benzersiz, boş olmayan bir değer içermelidir.
- ISO standardı CREATE ve ALTER TABLE deyimlerinde FOREIGN KEY cümlesini destekler:

```
PRIMARY KEY(personelNo)
PRIMARY KEY(clientNo, propertyNo)
```

- Tablo başına yalnızca bir PRIMARY KEY cümlesi olabilir. UNIQUE kullanarak alternatif anahtarlar için benzersizliği hala sağlayabilir:

```
UNIQUE(telNo)
```


IEF - REFERANS BÜTÜNLÜĞÜ

- FK, yabancı FK içeren alt tablodaki her satırı eşleşen PK içeren üst tablonun satırına bağlayan sütun veya sütun kümesidir.
- Referans bütünlüğü, FK bir değer içeriyorsa, bu üst tablodaki mevcut satıra atıfta bulunması gerektiği anlamına gelir.
- ISO standardı, CREATE ve ALTER TABLE'da FOREIGN KEY cümlesi ile FK'lerin tanımlanmasını destekler:

```
FOREIGN KEY(şubeNo) REFERENCES Şube
```


IEF - REFERANS BÜTÜNLÜĞÜ

- Üst tablodaki CK değeriyle eşleşmeden alt tabloda FK değeri oluşturmaya çalışan herhangi bir INSERT/UPDATE işlemi reddedilir.
- Üst tablodaki bir CK değerini alt tabloda eşleşen satırlarla güncellemeye/silmeye çalışırken gerçekleştirilen eylem, ON UPDATE ve ON DELETE alt cümleleri kullanılarak belirtilen referans eylemine bağlıdır:

```
CASCADE              - NULL AYARLA
VARSAYILAN AYARLA    - EYLEM YOK
```


IEF - REFERANS BÜTÜNLÜĞÜ

- **CASCADE**: Üstteki satırı silin ve alttaki eşleşen satırları silin ve bu şekilde basamaklı olarak devam edin.
- **SET NULL**: Üstteki satırı silin ve alttaki FK sütun(lar)ını NULL olarak ayarlayın. Yalnızca FK sütunları NULL DEĞİL ise geçerlidir.
- **SET DEFAULT**: Üstteki satırı siler ve alttaki her FK bileşenini belirtilen varsayılana ayarlar. Yalnızca FK sütunları için DEFAULT belirtilmişse geçerlidir.
- **EYLEM YOK**: Ebeveynden silmeyi reddedin. Varsayılan değer.

```
FOREIGN KEY (staffNo) REFERENCES Personel
                                          SI
LINDIĞINDE NULL OLARAK AYARLANIR
FOREIGN KEY (sahipNo) REFERENCES Sahip        AÇIK
                                              GÜNCELLEME
CASCADE
```

IEF - GENEL KISITLAMALAR

- CREATE ve ALTER TABLE'da CHECK/UNIQUE kullanabilir.
- CHECK cümlesine benzer şekilde, ayrıca:

```
CREATE ASSERTION AssertionName
CHECK (searchCondition)

CREATE ASSERTION StaffNotHandlingTooMuch
CHECK (NOT EXISTS      (SELECT personelNo
FROM PropertyForRent
GROUP BY staffNo HAVING
COUNT(*)> 100))
```


VERİ TANIMI

- SQL DDL, şemalar, etki alanları, tablolar, görünümler ve dizinler gibi veritabanı nesnelerinin oluşturulmasını ve yok edilmesini sağlar.
- Ana SQL DDL deyimleri şunlardır: ŞEMA

```
OLUŞTUR ŞEMAYI BIRAK ETKI ALANI
OLUŞTUR/DEĞIŞTIR ETKI ALANINI
BIRAK TABLO OLUŞTUR/DEĞIŞTIR
TABLOYU BIRAK GÖRÜNÜM OLUŞTUR
  DAMLA GÖRÜNÜM
```

- Birçok DBMS ayrıca şunları da sağlar:

```
DİZİN OLUŞTUR      DÜŞME ENDEKSİ
```


VERİ TANIMI

- İlişkiler ve diğer veritabanı nesneleri bir ortamda bulunur.
- Her ortam bir veya daha fazla katalog içerir ve her bir dizi şemadan oluşur.
- Şema, ilgili veritabanı nesnelerinin koleksiyonu olarak adlandırılır.
- Bir şemadaki nesneler tablolar, görünümler, etki alanları, onaylar, harmanlamalar, çeviriler ve karakter kümeleri olabilir. Hepsinin sahibi aynıdır.

ŞEMA OLUŞTUR

```
CREATE SCHEMA [|
          AUTHORIZATION CreatorId ] DROP
SCHEMA Name [RESTRICT| CASCADE ]
```

- RESTRICT (varsayılan) ile şema boş olmalıdır, aksi takdirde işlem başarısız olur.
- CASCADE ile, şema ile ilişkili tüm nesneleri yukarıda tanımlanan sırayla bırakmak için işlem basamaklandırılır. Bu işlemlerden herhangi biri başarısız olursa DROP SCHEMA başarısız olur.

TABLO OLUŞTUR

```
CREATE TABLO TableName
{(colName dataType [NOT NULL] [UNIQUE] [DEFAULT
defaultOption]
[CHECK searchCondition] [,...]}
[PRIMARY KEY (listOfColumns),]
{[UNIQUE (listOfColumns),] [...,]}
{[FOREIGN KEY (listOfFKColumns)
  REFERENCES ParentTableName [(listOfCKColumns)], [ON
  UPDATE referentialAction]
  [ON DELETE referentialAction ]] [,...]}
 {[CHECK (searchCondition)] [,...] })
```


TABLO OLUŞTUR

- Belirtilen veri türünde bir veya daha fazla sütuna sahip bir tablo oluşturur.
- NOT NULL ile, sistem sütuna bir null ekleme girişimini reddeder.
- Sütun için bir DEFAULT değeri belirtebilir.
- Birincil anahtarlar her zaman NOT NULL olarak belirtilmelidir.
- FOREIGN KEY cümlesi, referans eylemiyle birlikte FK'yi belirtir.

ÖRNEK 1 - TABLO OLUŞTUR

```
CREATE DOMAIN OwnerNumber AS VARCHAR(5)
CHECK (VALUE IN (SELECT ownerNo FROM PrivateOwner));
CREATE DOMAIN StaffNumber AS VARCHAR(5)
CHECK (VALUE IN (SELECT staffNo FROM Staff)); CREATE
DOMAIN PNumber AS VARCHAR(5);
CREATE DOMAIN PRooms AS SMALLINT;
CHECK(VALUE BETWEEN 1 AND 15); CREATE
DOMAIN PRent AS DECIMAL(6,2)
CHECK(VALUE BETWEEN 0 AND 9999.99);
```


ÖRNEK 1 - TABLO OLUŞTUR

```
CREATE TABLE PropertyForRent (
propertyNo       PNumarası  NULL DEĞİL,
....
odalar          Odalar      NOT NULL DEFAULT 4,
kira           PRent NOT NULL,         DEFAULT 600,
sahipHayır  SahipNumarası    NOT NULL,
personelNo PersonelNumarası
Constraint StaffNotHandlingTooMuch ....
branchNo BranchNumber      NOT NULL,
PRIMARY KEY (propertyNo),
FOREIGN KEY (staffNo) REFERENCES Personel
                SILME IŞLEMINDE NULL KÜMESI GÜNCELLEME
                IŞLEMINDE KADEMELI....);
```


ALTER TABLOSU

- Tabloya yeni bir sütun ekleyin.
- Tablodan bir sütun düşürün.
- Yeni bir tablo kısıtlaması ekleyin.
- Bir tablo kısıtlamasını bırakın.
- Bir sütun için varsayılan ayarlayın.
- Bir sütun için varsayılanı bırakın.

ÖRNEK 2 - TABLOYU DEĞIŞTIR

- Pozisyon sütunu için 'Asistan' varsayılanını kaldırarak ve cinsiyet sütunu için varsayılanı kadın ('F') olarak ayarlayarak Personel tablosunu değiştirin.

```
ALTER TABLO Personel
      ALTER position DROP DEFAULT;
ALTER TABLE Personel
      ALTER sex SET DEFAULT 'F';
```


ÖRNEK 2 - TABLOYU DEĞIŞTIR

- PropertyForRent'ten personelin bir seferde 100'den fazla mülkle ilgilenmesine izin verilmemesine ilişkin kısıtlamayı kaldırın. Müşteri tablosuna yeni sütun ekleyin.

```
ALTER TABLOSU PropertyForRent
      DROP CONSTRAINT StaffNotHandlingTooMuch; ALTER
TABLO Müşteri
      ADD prefNoRooms PRooms;
```


TABLO DÜŞÜR

```
DROP TABLO TableName [RESTRICT | CASCADE]
      örneğin DROP TABLE PropertyForRent;
```

- Adlandırılmış tabloyu ve tüm satırları kaldırır.
- RESTRICT ile, başka herhangi bir nesnenin varlığı bu tablonun varlığının devamına , SQL talebe izin vermez.
- CASCADE ile SQL tüm bağımlı nesneleri (ve bu nesnelere bağımlı nesneleri) düşürür.

GÖRÜNÜMLER

- Görünüm
  - Başka bir ilişki üretmek için temel üzerinde çalışan bir veya daha fazla ilişkisel işlemin dinamik sonucu.
- Veritabanında gerçekte var olmayan ancak istek üzerine, istek anında üretilen sanal ilişki.

GÖRÜNÜMLER

- Bir görünümün içeriği, bir veya daha fazla temel ilişki üzerinde bir sorgu olarak tanımlanır.
- Görünüm çözümleme ile, görünüm üzerindeki tüm işlemler otomatik olarak türetildiği ilişkiler üzerindeki işlemlere çevrilir.
- Görünüm somutlaştırma ile, görünüm geçici bir tablo olarak depolanır ve bu tablo, altta yatan temel tablolar güncellendikçe korunur.

GÖRÜNÜMLER OLUŞTURMAK IÇIN SQL

```
CREATE VIEW ViewName [ (newColumnName [,...]) ]
AS subselect
[BASAMAKLI | YEREL] KONTROL SEÇENEĞI ILE]
```

- Görünümdeki her sütuna bir ad atayabilir.
- Sütun adları listesi belirtilirse, subselect tarafından üretilen sütun sayısı ile aynı öğeye sahip olmalıdır.
- Atlanırsa, her sütun alt seçimdeki ilgili sütunun adını alır.

GÖRÜNÜMLER OLUŞTURMAK IÇIN SQL

- Bir sütun adında herhangi bir belirsizlik varsa liste belirtilmelidir.
- Alt seçim, tanımlayıcı sorgu olarak bilinir.
- WITH CHECK OPSİYONU, bir satırın tanımlama sorgusunun WHERE cümlesini karşılayamaması durumunda, altta yatan temel tabloya eklenmemesini sağlar.
- Alt seçimde başvurulan tüm tablolarda SELECT ayrıcalığı ve başvurulan sütunlarda kullanılan tüm alanlarda USAGE ayrıcalığı gerekir.

ÖRNEK 3 - YATAY GÖRÜNÜM OLUŞTURUN

- B003 şubesindeki yöneticinin yalnızca kendi ofisinde çalışan personelin ayrıntılarını görebilmesi için görünüm oluşturun.

```
CREATE VIEW Manager3Staff
AS    SELECT *
      Personelden
      WHERE şubeNo = 'B003';
```

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000.00	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000.00	B003
SG5	Susan	Brand	Manager	F	3-Jun-40	24000.00	B003

ÖRNEK 4 - DİKEY GÖRÜNÜM OLUŞTURUN

- B003 şubesinde maaşlar hariç personel ayrıntılarının görünümünü oluşturun.

```
CREATE VIEW Personel3
AS SELECT staffNo, fName, lName, position, sex FROM
      Personel
      WHERE branchNo=  'B003';
```

staffNo	fName	lName	position	sex
SG37	Ann	Beech	Assistant	F
SG14	David	Ford	Supervisor	M
SG5	Susan	Brand	Manager	F

ÖRNEK 5 - GRUPLANDIRILMIŞ VE BİRLEŞTİRİLMİŞ GÖRÜNÜMLER

- Çalıştıkları şube numarası, personel sayısı ve yönettikleri mülk sayısı dahil olmak üzere kiralık yöneten personelin görünümünü oluşturun.

```
CREATE VIEW StaffPropCnt (şubeNo, personelNo, cnt) AS
SELECT s.şubeNo, s.personelNo, COUNT(*)
      FROM Staff s, PropertyForRent p
      WHERE s.staffNo = p.staffNo GROUP
      BY .branchNo, s.staffNo;
```

branchNo	staffNo	cnt
B003	SG14	1
B003	SG37	2
B005	SL41	1
B007	SA9	1

DÜŞEN GÖRÜNÜMLER

```
DROP VIEW ViewName [RESTRICT|  CASCADE]
```

- Görünüm tanımının veritabanından silinmesine neden olur.
- Örneğin:

```
GÖRÜNÜMÜ DÜŞÜR Müdür3Personel;
```


DÜŞEN GÖRÜNÜMLER

- CASCADE ile, ilgili tüm bağımlı nesneler silinir; yani, görünüm üzerinde tanımlanan görünümler düşürülür.
- RESTRICT (varsayılan) ile, başka herhangi bir nesnenin varlığı düşürülen görünümün varlığının devamına bağlıysa, komut reddedilir.

ÇÖZÜNÜRLÜĞÜ GÖRÜNTÜLE

- B003 şubesinde her bir üye tarafından yönetilen mülk sayısını sayın.

```
SELECT staffNo, cnt FROM
StaffPropCnt
WHERE şubeNo= 'B003' ORDER BY
personelNo;
```


ÇÖZÜNÜRLÜĞÜ GÖRÜNTÜLE

- SELECT listesindeki görünüm sütun adları, tanımlama sorgusundaki karşılık gelen sütun adlarına çevrilir:

```
SELECT s.personelNo As personelNo, COUNT(*) As cnt
```

- FROM içindeki görünüm adları, tanımlama sorgusunun karşılık gelen FROM listeleriyle değiştirilir:

```
FROM Personel s, PropertyForRent p
```


ÇÖZÜNÜRLÜĞÜ GÖRÜNTÜLE

- Kullanıcı sorgusundaki WHERE, AND kullanılarak tanımlama sorgusunun WHERE'si ile birleştirilir:

```
WHERE s.personelNo = p.personelNo VE şubeNo = 'B003'
```

- GROUP BY ve HAVING cümleleri tanımlayıcı sorgudan kopyalanmıştır:

```
GROUP BY s.şubeNo, s.personelNo
```

- ORDER BY, sorgu sütun adını tanımlayan sorgu sütun adına çevrilmiş görünüm sütun adıyla sorgudan kopyalandı

```
ORDER BY s.personelNo
```


ÇÖZÜNÜRLÜĞÜ GÖRÜNTÜLE

- Son birleştirilmiş sorgu şimdi sonucu üretmek için yürütülür: SELECT

```
s.staffNo AS staffNo, COUNT(*) AS cnt FROM Staff
s, PropertyForRent p
WHERE s.personelNo = p.personelNo AND
                    şubeNo= 'B003'
GROUP BY s.şubeNo, s.personelNo ORDER
BY s.personelNo;
```


KISITLAMALARI GÖRÜNTÜLE

SQL, görünümlerin oluşturulması ve kullanılması konusunda çeşitli kısıtlamalar getirir.

- Görünümdeki sütun bir toplama işlevine dayanıyorsa:
  - Sütun, yalnızca görünüme erişen sorguların SELECT ve ORDER BY cümlelerinde görünebilir.
  - Sütun WHERE içinde kullanılamaz veya görünüme dayalı herhangi bir sorguda bir toplama işlevinin bağımsız değişkeni olamaz.

KISITLAMALARI GÖRÜNTÜLE

- Örneğin, aşağıdaki sorgu başarısız olur:

```
SELECT COUNT(cnt)
FROM StaffPropCnt;
```

- Benzer şekilde, aşağıdaki sorgu da başarısız olacaktır:

```
SELECT *
FROM StaffPropCnt WHERE
cnt > 2;
```

KISITLAMALARI GÖRÜNTÜLE

- Gruplandırılmış görünüm hiçbir zaman bir temel tablo veya görünüm ile birleştirilemez.
  - Örneğin, StaffPropCnt görünümü gruplandırılmış bir görünümdür, bu nedenle bu görünümü başka bir tablo veya görünümle birleştirme girişimi başarısız olur.

GÜNCELLENEBILIRLIĞI GÖRÜNTÜLE

- Temel tabloda yapılan tüm güncellemeler, temel kapsayan tüm görünümlere yansıtılır.
- Benzer şekilde, görünüm güncellenirse temel tablo(lar)ın değişikliği yansıtması beklenebilir.
- Ancak, StaffPropCnt görünümünü tekrar ele alın.
- Eğer B003 şubesinde SG5'in 2 mülk yönettiğini gösteren bir kayıt eklemeye çalışsaydık:

```
INSERT INTO StaffPropCnt VALUES
('B003', 'SG5', 2);
```

- SG5'in hangi mülkleri yönettiğini gösteren PropertyForRent'e 2 kayıt eklemek zorundayım. Ancak, bunların hangi mülkler olduğunu bilmiyoruz; yani birincil anahtarları bilmiyoruz!

GÜNCELLENEBILIRLIĞI GÖRÜNTÜLE

- Görünüm tanımını değiştirirseniz ve sayımı gerçek mülk numaralarıyla değiştirirseniz:

```
CREATE VIEW StaffPropList (şubeNo,
                        personelNo, mülkNo)
AS SELECT s.branchNo, s.staffNo, p.propertyNo FROM
            Staff s, PropertyForRent p WHERE
            s.staffNo = p.staffNo;
```


GÜNCELLENEBILIRLIĞI GÖRÜNTÜLE

- Şimdi kaydı yerleştirmeyi deneyin:

```
INSERT INTO StaffPropList VALUES
('B003', 'SG5', 'PG19');
```

- Hala sorun var, çünkü PropertyForRent'te posta kodu/personelNo hariç tüm sütunlarda null'lara izin verilmiyor.
- Ancak, kalan boş olmayan sütunlara değer vermenin bir yolu yoktur.

GÜNCELLENEBILIRLIĞI GÖRÜNTÜLE

- ISO, bir görünümün yalnızca ve yalnızca şu durumlarda güncellenebilir olduğunu belirtir:
  - DISTINCT belirtilmemiştir.
  - Tanımlama sorgusunun SELECT listesindeki her öğe bir sütun adıdır ve hiçbir sütun birden fazla kez görünmez.
  - FROM cümlesi, , birleştirme, kesişim veya farka dayalı görünümleri hariç tutarak yalnızca bir tablo belirtir.
  - Dış tabloya başvuran iç içe SELECT yok.
  - GROUP BY veya HAVING cümlesi yok.
  - Ayrıca, görünüm aracılığıyla eklenen her satır, temel tablonun bütünlük kısıtlamalarını ihlal etmemelidir.

GÜNCELLENEBİLİR GÖRÜNÜM

- Görünümün güncellenebilir olması için, DBMS'nin herhangi bir satırı veya sütunu kaynak tablodaki satırına veya sütununa kadar izleyebilmesi gerekir.

ÇEK SEÇENEĞI ILE

- Satırlar, sorguyu tanımlayan WHERE koşulunu karşıladıkları için bir görünümde bulunur.
- Bir satır değişir ve artık koşulu karşılamazsa, görünümden kaybolur.
- Görünümde ekleme/güncelleme WHERE koşulunu karşılamalarına neden olduğunda görünümde yeni satırlar görünür.
- Bir görünüme giren veya görünümden çıkan satırlara geçiş yapan satırlar denir.
- WITH CHECK SEÇENEĞİ, bir satırın görünümün dışına taşınmasını yasaklar.

ÇEK SEÇENEĞI ILE

- LOCAL/CASCADED görünüm hiyerarşileri için geçerlidir.
- LOCAL ile, görünümdeki herhangi bir satır ekleme/güncelleme ve bu görünüm üzerinde doğrudan veya dolaylı olarak tanımlanan herhangi bir görünüm, satır türetilmiş görünümden/tablodan da kaybolmadığı sürece satırın görünümden kaybolmasına neden olmamalıdır.
- CASCADED (varsayılan) ile, bu görünümde ve bu görünüm üzerinde doğrudan veya dolaylı olarak tanımlanan herhangi bir görünümde herhangi bir satır ekleme/güncelleme, satırın görünümden kaybolmasına neden .

ÖRNEK 6 - ÇEK SEÇENEĞI ILE

```
CREATE VIEW Manager3Staff
AS    SELECT *
Personelden
WHERE branchNo= 'B003' WITH
CHECK OPTION;
```

- Satırın görünümden taşınmasına neden olacağından B003 satırının şube numarası B002 olarak güncellenemiyor.
- Ayrıca, B003'e eşit olmayan bir şube numarasına sahip bir satırı görünüme ekleyemezsiniz.

ÖRNEK 6 - ÇEK SEÇENEĞI ILE

- Şimdi aşağıdakileri düşünün:

```
CREATE VIEW LowSalary
AS SELECT * FROM Staff WHERE salary> 9000; CREATE
VIEW HighSalary
AS SELECT * FROM LowSalary WHERE
salary > 10000
YEREL KONTROL SEÇENEĞI ILE;
CREATE VIEW Manager3Staff
AS SELECT * FROM HighSalary WHERE
branchNo = 'B003';
```


ÖRNEK 6 - ÇEK SEÇENEĞI ILE

```
UPDATE Manager3Staff SET
salary = 9500
WHERE personelNo=  'SG37';
```

- Bu güncelleme başarısız olur: güncelleme satırın HighSalary'den kaybolmasına neden olsa da, satır LowSalary'den kaybolmaz.
- Ancak, güncelleme maaşı 8000 olarak ayarlamaya çalışırsa, satır artık LowSalary'nin bir parçası olmayacağı için güncelleme başarılı olur.

ÖRNEK 6 - ÇEK SEÇENEĞI ILE

- HighSalary WITH CASCADED CHECK OPTION belirtmiş olsaydı, maaşın 9500 veya 8000 olarak ayarlanması çünkü satır HighSalary'den kaybolurdu.
- Bunun gibi anormallikleri önlemek için, her görünüm KASKOLU KONTROL SEÇENEĞİ kullanılarak oluşturulmalıdır.

GÖRÜŞ AVANTAJLARI

- Veri bağımsızlığı
- Para Birimi
- Geliştirilmiş güvenlik
- Azaltılmış karmaşıklık
- Kolaylık
- Özelleştirme
- Veri bütünlüğü

GÖRÜNÜMLERIN DEZAVANTAJLARI

- Güncelleme kısıtlaması
- Yapı kısıtlaması
- Performans

SOMUTLAŞTIRMAYI GÖRÜNTÜLE

- Görünüm çözümleme mekanizması, özellikle de görünüme sık erişiliyorsa yavaş olabilir.
- Görünüm materyalleştirme, görünüm ilk kez sorgulandığında görünümü geçici tablo olarak depolar.
- Bundan sonra, materyalleştirilmiş görünüme dayalı sorgular, görünümü her seferinde yeniden hesaplamaktan daha hızlı olabilir.
- Zorluk, temel tablo(lar) güncellenirken görünümün güncelliğini korumaktır.

BAKIMI GÖRÜNTÜLE

- Görünüm bakımı, yalnızca görünümü güncel tutmak için gerekli olan değişiklikleri uygulamayı amaçlar.
- Aşağıdaki görüşü dikkate alın:

```
CREATE VIEW StaffPropRent(staffNo) AS
      SELECT DISTINCT personelNo
            FROM PropertyForRent
            WHERE şubeNo= 'B003' VE kira >
                  400;
```

staffNo
SG37
SG14

SOMUTLAŞTIRMAYI GÖRÜNTÜLE

- PropertyForRent içine kira<= 400 ile satır eklenirse görünüm değişmez.
- B003 şubesinde PG24 mülkü için personelNo = SG19 ve kira= 550 ile satır eklenirse, satır somutlaştırılmış görünümde görünecektir.
- B003 şubesindeki PG54 mülkü için personelNo= SG37 ve kira = 450 ile satır eklenirse, materyalleştirilmiş görünüme yeni satır eklenmesine gerek kalmaz.
- Eğer delete özelliği PG24 ise, satır materyalize görünümden silinmelidir.
- PG54 özelliği silinirse, PG37 için satır silinmemelidir (mevcut PG21 özelliği nedeniyle).

İŞLEMLER

- SQL, COMMIT ve ROLLBACK'e dayalı işlem modelini tanımlar.
- Transaction, kurtarma açısından atomik olması garanti edilen bir veya daha fazla SQL deyimi içeren mantıksal çalışma birimidir.
- Bir SQL işlemi otomatik olarak işlemi başlatan bir SQL deyimiyle (örn. SELECT, INSERT) başlar.
- İşlem tarafından yapılan değişiklikler, işlem tamamlanana kadar eşzamanlı olarak yürütülen diğer işlemler tarafından görülmez.

İŞLEMLER

- İşlem dört yoldan biriyle tamamlanabilir:
  - COMMIT işlemi başarıyla sonlandırarak değişiklikleri kalıcı hale getirir.
  - ROLLBACK işlemi iptal eder, işlem tarafından yapılan tüm değişiklikleri geri alır.
  - Programatik SQL için, başarılı program sonlandırması, COMMIT yürütülmemiş olsa bile son işlemi başarıyla sonlandırır.
  - Programatik SQL için, anormal program sonu işlemi iptal eder.

İŞLEMLER

- Yeni işlem, bir sonraki işlemi başlatan ifade ile başlar.
- SQL işlemleri iç içe geçemez.
- SET TRANSACTION işlemi yapılandırır:

```
IŞLEM AYARLA
[YALNIZCA OKUNUR | OKUNUR YAZILIR] |
[IZOLASYON SEVIYESI OKUNUR |
OKUNDU TAAHHÜT EDILDI|TEKRARLANABILIR OKUNDU
|SERIALIZABLE ]
```


ANINDA VE ERTELENMIŞ BÜTÜNLÜK KISITLAMALARI

- Kısıtlamaların her zaman hemen kontrol edilmesini istemeyin, bunun yerine işlem işlendiğinde kontrol edilmesini isteyin.
- Kısıtlama, kısıtlamanın her işlemin başlangıcında üstlendiği modu belirten INITIALLY IMMEDIATE veya INITIALLY DEFERRED olarak tanımlanabilir.
- Önceki durumda, [NOT] DEFERRABLE niteleyicisini kullanarak modun sonradan değiştirilip değiştirilemeyeceğini belirtmek de mümkündür.
- Varsayılan mod BAŞLANGIÇTA HEMEN'dir.

ANINDA VE ERTELENMIŞ BÜTÜNLÜK KISITLAMALARI

- SET CONSTRAINTS deyimi, geçerli işlem için belirtilen kısıtlamaların modunu ayarlamak için kullanılır:

```
KISITLAMALARI AYARLA
{ALL | constraintName [, . . ]}
    {ERTELENMIŞ ¦ HEMEN}
```


ERIŞIM KONTROLÜ - YETKILENDIRME TANIMLAYICILARI VE SAHIPLIK

- Yetkilendirme tanımlayıcısı, bir kullanıcının kimliğini belirlemek için kullanılan normal SQL tanımlayıcısıdır. Genellikle ilişkili bir parolası vardır.
- Kullanıcının hangi nesnelere başvurabileceğini ve bu nesneler üzerinde hangi işlemlerin gerçekleştirilebileceğini belirlemek için kullanılır.
- SQL'de oluşturulan her nesnenin, nesnenin ait olduğu şemanın AUTHORIZATION cümlesinde tanımlandığı gibi bir sahibi vardır.
- Bunu bilebilecek tek kişi mal sahibidir.

AYRICALIKLAR

- Kullanıcının verilen temel tablo veya görünüm üzerinde

gerçekleştirmesine izin verilen eylemler: SELECT Bir tablodan veri alın.

INSERT Bir tabloya yeni satırlar ekleyin.

GÜNCELLE Bir tablodaki veri satırlarını değiştirin.

DELETE Bir tablodan veri satırlarını silin.

REFERENCES Bütünlük kısıtlamalarında adlandırılmış tablonun sütunlarına referans verir.

KULLANIM Etki alanlarını, harmanlamaları, karakter kümelerini ve çevirileri kullanın.

AYRICALIKLAR

- INSERT/UPDATE/REFERENCES işlemlerini adlandırılmış sütunlarla kısıtlayabilir.
- Tablo sahibi GRANT deyimini kullanarak diğer kullanıcılara gerekli ayrıcalıkları vermelidir.
- Görünüm oluşturmak için, kullanıcının görünümü tüm tablolar üzerinde SELECT ayrıcalığına ve adlandırılmış sütunlar üzerinde REFERENCES ayrıcalığına sahip olması gerekir.

HİBE

```
HİBE        {PrivilegeList| TÜM AYRICALIKLAR}
AÇIK  ObjectName
K     {AuthorizationIdList| PUBLIC}
[HIBE SEÇENEĞI ILE]
```

- PrivilegeList, virgülle ayrılmış yukarıdaki ayrıcalıklardan bir veya daha fazlasından oluşur.
- ALL PRIVILEGES (TÜM AYRICALIKLAR) bir kullanıcıya tüm ayrıcalıkları verir.

HİBE

- PUBLIC, mevcut ve gelecekteki tüm yetkili kullanıcılara erişim izni verilmesini sağlar.
- ObjectName bir temel tablo, görünüm, etki alanı, karakter kümesi, harmanlama veya çeviri olabilir.
- WITH GRANT SEÇENEĞİ ayrıcalıkların izin verir.

ÖRNEK 7 & 8 - HİBE

- Manager'a Staff tablosu için tam ayrıcalıklar verin.

```
TÜM AYRICALIKLARI VERIN
AÇIK Personel
HİBE SEÇENEĞİ İLE YÖNETİCİYE;
```

- Personel ve Müdür kullanıcılarına Personel maaşı sütununda SELECT ve UPDATE verin.

```
GRANT SELECT, UPDATE (salary)
AÇIK Personel
Personele, Direktör;
```


ÖRNEK 9 - KAMUYA ÖZEL AYRICALIKLAR VERİN

- Tüm kullanıcılara Branch tablosunda SELECT verin.

```
GRANT SELECT
halka açik
şubede;
```


İPTAL

- REVOKE, GRANT ile verilen ayrıcalıkları geri alır.

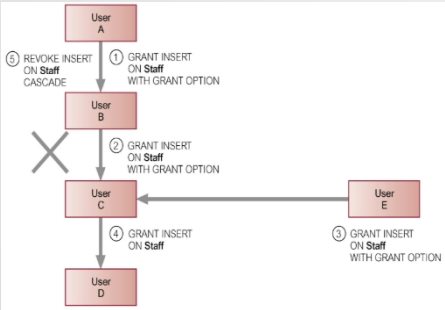
```
IPTAL ETMEK [IÇIN OPSIYON VERMEK]
    {PrivilegeList| TÜM AYRICALIKLAR} ON
ObjectName
FROM {AuthorizationIdList| PUBLIC}
    [RESTRICT | CASCADE]
```

- TÜM AYRICALIKLAR, kullanıcının ayrıcalıkları iptal etmesiyle bir kullanıcıya verilen tüm ayrıcalıkları ifade eder.

İPTAL

- GRANT OPTION FOR, GRANT'ın WITH GRANT OPTION'ı aracılığıyla aktarılan ayrıcalıkların, ayrıcalıkların kendisinden ayrı olarak iptal edilmesini sağlar.
- REVOKE, CASCADE anahtar sözcüğü belirtilmediği sürece, görünüm gibi terk edilmiş bir nesneyle sonuçlanırsa başarısız olur.
- Diğer kullanıcılar tarafından bu kullanıcıya verilen ayrıcalıklar etkilenmez.

İPTAL



ÖRNEK 10 & 11 - BELİRLİ AYRICALIKLARIN İPTALİ

- Tüm kullanıcılardan Branch tablosunda SELECT ayrıcalığını iptal edin.

```
SEÇİMİ İPTAL ET
AÇIK Şube FROM
PUBLIC;
```

- Personel masasında Direktöre verilen tüm ayrıcalıkları iptal edin.

```
TÜM AYRICALIKLARI IPTAL ETMEK
AÇIK Personel
Yönetmen'den;
```


CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ VERİTABANI GELİŞTİRME

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- Bir bilgi sisteminin ana bileşenleri.
- Veritabanı sistemi geliştirme yaşam döngüsünün ana aşamaları.
- Veritabanı tasarımının ana aşamaları: kavramsal, mantıksal ve fiziksel tasarım.
- CASE araçlarının faydaları.
- Bir VTYS nasıl değerlendirilir ve seçilir.
- Veri yönetimi ve veritabanı yönetimi arasındaki ayrım.
- Veri yönetimi ve veritabanı yönetimi ile ilgili amaç ve görevler.

YAZILIM DEPRESYONU

- Son birkaç on yılda, birçoğu sürekli bakım gerektiren yazılım uygulamalarının çoğalmasına tanık olduk:
  - hataların düzeltilmesi,
  - yeni kullanıcı gereksinimlerinin uygulanması,
  - Yazılımın yeni veya yükseltilmiş platformlarda çalışacak şekilde değiştirilmesi.
- Bakım için harcanan çaba, kaynakları endişe verici bir oranda emmeye başladı.
- Sonuç olarak, birçok büyük yazılım projesi
  - Geç oldu,
  - bütçenin üzerinde,
  - güvenilmez,
  - bakımı zor,
  - kötü performans gösterdi.
- 1960'ların sonlarında, günümüzde "yazılım bunalımı" olarak adlandırılan "yazılım krizine" açmıştır.

YAZILIM DEPRESYONU

- Yazılım projelerinin başarısızlığının başlıca nedenleri :
  - eksiksiz bir gereksinim spesifikasyonunun olmaması;
  - uygun geliştirme metodolojisinin olmaması;
  - Tasarımın yönetilebilir bileşenlere zayıf bir şekilde ayrıştırılması.
- Bilgi Sistemleri Yaşam Döngüsü (ISLC) adı verilen yapılandırılmış bir geliştirme yaklaşımı önerilmiştir.

BİLGİ SİSTEMİ

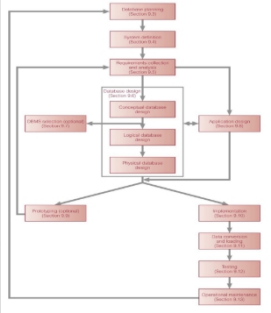
Bir kuruluş genelinde bilginin toplanmasını, yönetilmesini, kontrol edilmesini ve yayılmasını sağlayan kaynaklar.

- Veritabanı, IS'nin temel ve geliştirilmesi/kullanımı, kurumun daha geniş gereksinimleri perspektifinden görülmelidir.

VERITABANI SISTEMI GELIŞTIRME YAŞAM DÖNGÜSÜ

- Veritabanı planlaması
- Sistem tanımı
- Gereksinimlerin toplanması ve analizi
- Veritabanı tasarımı
- DBMS seçimi (isteğe bağlı)
- Uygulama tasarımı
- Prototipleme (isteğe bağlı)
- Uygulama
- Veri dönüştürme ve yükleme
- Test
- Operasyonel bakım

VERITABANI SISTEMI GELIŞTIRME YAŞAM DÖNGÜSÜNÜN AŞAMALARI



VERITABANI PLANLAMASI

- Veritabanı sistemi geliştirme yaşam döngüsü aşamalarının mümkün olduğunca verimli ve etkin bir şekilde gerçekleştirilmesini sağlayan yönetim faaliyetleri.
- Kurumun genel IS stratejisi ile entegre edilmelidir.

VERİTABANI PLANLAMASI - GÖREV BİLDİRİMİ

- Veritabanı projesi için *misyon beyanı*, veritabanı uygulamasının temel amaçlarını tanımlar.
- Veritabanı projesini yürütenler normalde misyon bildirimini tanımlar.
- Misyon bildirimi, veritabanı projesinin amacını netleştirmeye yardımcı olur ve gerekli veritabanı sisteminin verimli ve etkili bir şekilde oluşturulmasına yönelik daha net bir yol sağlar.

VERİTABANI PLANLAMASI - GÖREV HEDEFLERİ

- Misyon ifadesi tanımlandıktan sonra, misyon *hedefleri* tanımlanır.
- Her hedef, veritabanının desteklemesi gereken belirli bir görevi tanımlamalıdır.
- Yapılacak işi, işi yapmak için gerekli kaynakları ve tüm bunları ödemek için gerekli parayı belirten bazı ek bilgilerle birlikte verilebilir.

VERITABANI PLANLAMASI

- Veritabanı planlaması, yöneten standartların geliştirilmesini de içermelidir:
  - verilerin nasıl toplanacağı,
  - formatın nasıl belirtilmesi gerektiği,
  - hangi gerekli belgelere ihtiyaç duyulacağı,
  - tasarım ve uygulamanın nasıl ilerlemesi gerektiği.

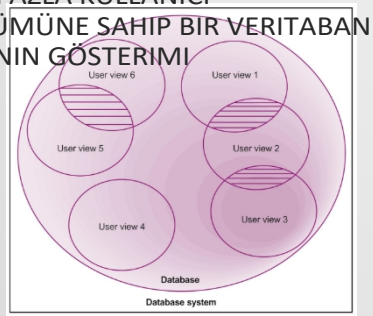
SİSTEM TANIMI

- Veritabanı sisteminin kapsamını ve sınırlarını ve ana kullanıcı görünümlerini açıklar.
- Kullanıcı görüşü, bir veritabanı kullanıcı açısından ne istendiğini tanımlar:
  - belirli bir iş rolü (Müdür veya Süpervizör gibi) veya
  - kurumsal uygulama alanı (pazarlama, personel veya stok kontrolü gibi).

SİSTEM TANIMI

- Veritabanı uygulaması bir veya daha fazla kullanıcı görünümüne sahip olabilir.
- Kullanıcı görüşlerinin belirlenmesi, yeni sistem için gereksinimler geliştirilirken veritabanının önemli kullanıcılarının unutulmamasını sağlamaya yardımcı olur.
- Kullanıcı görüşleri ayrıca karmaşık veritabanı sistemlerinin geliştirilmesine yardımcı olarak gereksinimlerin yönetilebilir parçalara ayrılmasını sağlar.

BIRDEN FAZLA KULLANICI GÖRÜNÜMÜNE SAHIP BIR VERITABANI SISTEMININ GÖSTERIMI



GEREKSINIMLERIN TOPLANMASI VE ANALIZI

- Veritabanı sistemi tarafından desteklenecek organizasyon bölümü hakkında bilgi toplama ve analiz etme ve bu bilgileri kullanıcıların yeni sistem gereksinimlerini belirlemek için kullanma süreci.
- Aşağıdakiler de dahil olmak üzere her bir ana kullanıcı görünümü için bilgi toplanır:
  - Kullanılan veya üretilen verilerin bir açıklaması;
  - Verilerin nasıl kullanılacağına/üretileceğine ilişkin ayrıntılar;
  - Yeni veritabanı sistemi için herhangi bir ek gereksinim.
- Yeni veritabanı sistemine dahil edilecek gereksinimleri belirlemek için bilgiler analiz edilir. Gereksinim spesifikasyonunda açıklanır.

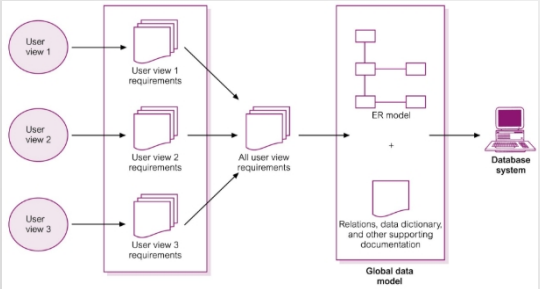
GEREKSINIMLERIN TOPLANMASI VE ANALIZI

- Bir diğer önemli faaliyet, birden fazla kullanıcı görünümüne sahip bir veritabanı sistemi için gereksinimlerin nasıl yönetileceğine karar vermektir.
- Üç ana yaklaşım:
  - merkezi yaklaşım;
  - entegrasyon yaklaşımını görüntüleyin;
  - her iki yaklaşımın kombinasyonu.

GEREKSINIMLERIN TOPLANMASI VE ANALIZI

- Merkezi yaklaşım
  - Her bir kullanıcı görünümü için gereksinimler tek bir kümesinde birleştirilir.
  - Veritabanı tasarımı aşamasında tüm kullanıcı görünümlerini temsil eden bir veri modeli oluşturulur.

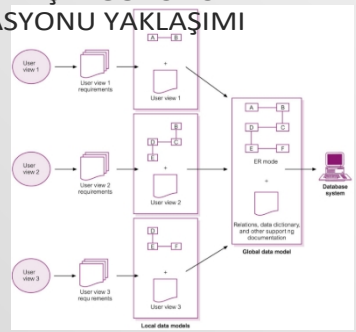
ÇOKLU KULLANICI GÖRÜNÜMLERINI YÖNETMEK IÇIN MERKEZI YAKLAŞIM



GEREKSINIMLERIN TOPLANMASI VE ANALIZI

- Entegrasyon yaklaşımını görüntüleyin
  - Her kullanıcı görünümü için gereksinimler ayrı listeler olarak kalır.
  - Her bir kullanıcı görünümünü temsil eden veri modelleri oluşturulur ve daha sonra veritabanı tasarımı aşamasında birleştirilir.
- Tek bir kullanıcı görünümünü (veya tüm kullanıcı bir alt kümesini) temsil eden veri modeline yerel veri modeli denir.
- Her model, veritabanının tüm kullanıcı görünümleri için olmasa da bir veya daha fazlası için gereksinimleri açıklayan diyagramlar ve belgeler içerir.
- Yerel veri modelleri, tasarımı sırasında daha sonraki bir aşamada birleştirilerek veritabanı için tüm kullanıcı görünümlerini temsil eden global bir veri modeli oluşturulur.

ÇOKLU KULLANICI GÖRÜNÜMLERINI YÖNETMEK IÇIN GÖRÜNÜM ENTEGRASYONU YAKLAŞIMI



VERITABANI TASARIMI

- Gerekli veritabanı sistemi için işletmenin misyon bildirimini ve misyon hedeflerini destekleyecek veritabanı tasarımı oluşturma süreci.
- Ana yaklaşımlar şunlardır:
  - Yukarıdan aşağıya
  - Aşağıdan yukarıya
  - İçten dışa
  - Karışık

VERITABANI TASARIMI

- Veri modellemenin temel amaçları şunlardır:
  - Verilerin anlamını (semantiğini) anlamaya yardımcı olmak için;
  - bilgi gereksinimleri hakkında iletişimi kolaylaştırmak için.
- Veri modeli oluşturmak, varlıklar, ilişkiler ve öznitelikler hakkındaki soruların yanıtlanmasını gerektirir.
- Bir veri modeli anlamamızı sağlar:
  - her kullanıcının verilere bakış açısı;
  - fiziksel temsillerinden bağımsız olarak verinin kendisinin doğası;
  - kullanıcı görünümleri arasında veri kullanımı.

OPTIMUM VERI MODELI ÜRETMEK IÇIN KRITERLER

Structural validity	Consistency with the way the enterprise defines and organizes information.
Simplicity	Ease of understanding by IS professionals and non-technical users.
Expressibility	Ability to distinguish between different data, relationships between data, and constraints.
Nonredundancy	Exclusion of extraneous information; in particular, the representation of any one piece of information exactly once.
Shareability	Not specific to any particular application or technology and thereby usable by many.
Extensibility	Ability to evolve to support new requirements with minimal effect on existing users.
Integrity	Consistency with the way the enterprise uses and manages information.
Diagrammatic representation	Ability to represent a model using an easily understood diagrammatic notation.

VERITABANI TASARIMI

- Veritabanı tasarımının üç aşaması:
  - Kavramsal veritabanı tasarımı
  - Mantıksal veritabanı tasarımı
  - Fiziksel veritabanı tasarımı.

KAVRAMSAL VERİTABANI TASARIMI

- Tüm fiziksel hususlardan bağımsız olarak bir kuruluşta kullanılan verilerin bir modelini oluşturma süreci.
- Veri modeli, kullanıcıların gereksinim spesifikasyonundaki bilgiler kullanılarak oluşturulur.
- Kavramsal veri modeli, mantıksal tasarım aşaması için bilgi .

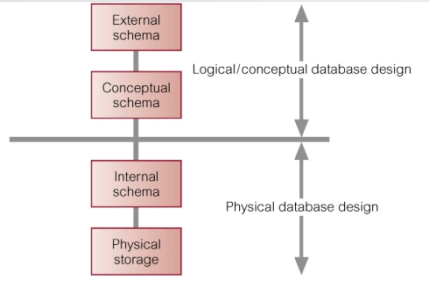
MANTIKSAL VERITABANI TASARIMI

- Belirli bir veri modeline (örneğin ilişkisel) dayalı, ancak belirli bir DBMS'den ve diğer fiziksel hususlardan bağımsız olarak bir kuruluşta kullanılan verilerin bir modelini oluşturma süreci.
- Kavramsal veri modeli rafine edilir ve mantıksal bir veri modeliyle eşleştirilir.

FIZIKSEL VERITABANI TASARIMI

- İkincil depolama üzerinde veritabanı uygulamasının bir tanımını üretme süreci.
- Verilere etkin erişim sağlamak için kullanılan temel ilişkileri, dosya organizasyonlarını ve dizinleri açıklar. Ayrıca ilişkili bütünlük kısıtlamalarını ve güvenlik önlemlerini de açıklar.
- Belirli bir DBMS sistemine göre uyarlanmıştır.

ÜÇ SEVİYELİ ANSI-SPARC MİMARİSİ VE VERİ TABANI TASARIM AŞAMALARI



DBMS SEÇİMİ

- Veritabanı sistemini desteklemek için uygun bir DBMS seçimi.
- Sistem gereksinimleri hakkında yeterli bilgiye sahip olunması koşuluyla mantıksal tasarımdan önce herhangi bir zamanda üstlenilebilir.
- Bir DBMS seçmenin ana adımları:
  - Çalışmanın İş Tanımı'nı tanımlayın;
  - iki veya üç ürünü kısa listeye alın;
  - ürünleri değerlendirin;
  - seçim önerisinde bulunmak ve rapor hazırlamak.

DBMS DEĞERLENDİRME ÖZELLİKLERİ

DATA DEFINITION	PHYSICAL DEFINITION
Primary key enforcement	File structures available
Foreign key specification	File structure maintenance
Data types available	Ease of reorganization
Data type extensibility	Indexing
Domain specification	Variable length fields/records
Ease of restructuring	Data compression
Integrity controls	Encryption routines
View mechanism	Memory requirements
Data dictionary	Storage requirements
Data independence	
Underlying data model	
Schema evolution	
ACCESSIBILITY	**TRANSACTION HANDLING**
Query language: SQL2/SQL:2011/ODMG compliant	Backup and recovery routines Checkpointing facility
Interfacing to 3GLs	Logging facility
Multi-user	Granularity of concurrency
Security	Deadlock resolution strategy
• Access controls	Advanced transaction models
• Authorization mechanism	Parallel query processing

DBMS DEĞERLENDİRME ÖZELLİKLERİ

UTILITIES	DEVELOPMENT
Performance measuring	4GL/SQL tools
Tuning	CASE tools
Load/unload facilities	Windows capabilities
User usage monitoring	Stored procedures, triggers, and rules
Database administration support	Web development tools
OTHER FEATURES	
Upgradability	Interoperability with other DBMSs and other systems
Vendor stability	Web integration
User base	Replication utilities
Training and user support	Distributed capabilities
Documentation	Portability
Operating system required	Hardware required
Cost	Network support
Online help	Object-oriented capabilities
Standards used	Architecture (2- or 3-tier client/server)
Version management	Performance
Extensible query optimization	Transaction throughput
Scalability	Maximum number of concurrent users
Support for reporting and analytical tools	XML and Web services support

ÖRNEK - DBMS ÜRÜNÜNÜN DEĞERLENDİRİLMESİ

DBMS: Sample product
Vendor: Sample vendor

Physical Definition Group

Features	Comments	Rating	Weighting	Score
File structures available	Choice of 4	8	0.15	1.2
File structure maintenance	NOT self-regulating	6	0.2	1.2
Ease of reorganization		4	0.25	1.0
Indexing		6	0.15	0.9
Variable length fields/records		6	0.15	0.9
Data compression	Specify with file structure	7	0.05	0.35
Encryption routines	Choice of 2	4	0.05	0.2
Memory requirements		0	0.00	0
Storage requirements		0	0.00	0
Totals		41	1.0	**5.75**
Physical definition group		5.75	0.25	**1.44**

UYGULAMA TASARIMI

- Veritabanını kullanan ve işleyen kullanıcı arayüzü ve uygulama programlarının tasarımı.
- Veritabanı tasarımı ve uygulama tasarımı paralel faaliyetlerdir.
- İki önemli faaliyet içerir:
  - işlem tasarımı;
  - kullanıcı arayüzü tasarımı.

UYGULAMA TASARIMI - İŞLEMLER

- Tek bir kullanıcı veya uygulama programı tarafından gerçekleştirilen ve veritabanının içeriğine erişen veya içeriğini değiştiren bir eylem veya eylem dizisi.
- Gerekli işlemlerin üst düzey özelliklerini tanımlamalı ve .

UYGULAMA TASARIMI - İŞLEMLER

- İşlemlerin önemli özellikleri:
  - işlem tarafından kullanılacak veriler;
  - işlemin işlevsel özellikleri;
  - işlemin çıktısı;
  - kullanıcılar için önemlidir;
  - beklenen kullanım oranı.
- Üç ana işlem türü: alma, güncelleme ve karma.

PROTOTİPLEME

- Bir veritabanı sisteminin çalışma oluşturma.
- Amaç
  - Bir sistemin iyi çalışan veya yetersiz olan özelliklerini belirlemek için;
  - iyileştirmeler ve hatta yeni özellikler önermek için;
  - Kullanıcıların gereksinimlerini netleştirmek için;
  - Belirli bir sistem tasarımının fizibilitesini değerlendirmek için.

UYGULAMA

- Veritabanı ve uygulama tasarımlarının fiziksel olarak gerçekleştirilmesi.
  - Veritabanı şemaları ve boş veritabanı dosyaları oluşturmak için DDL kullanın.
  - Belirtilen kullanıcı görünümlerini oluşturmak için DDL kullanın.
  - Uygulama programlarını oluşturmak için 3GL veya 4GL kullanın. Bu, muhtemelen bir ana programlama diline gömülü DML kullanılarak uygulanan veritabanı işlemlerini içerecektir.

VERİ DÖNÜŞTÜRME VE YÜKLEME

- Mevcut verilerin yeni aktarılması ve mevcut uygulamaların yeni veritabanında çalışacak şekilde dönüştürülmesi.
- Yalnızca yeni veritabanı sistemi eski bir sistemin yerini aldığında gereklidir.
  - DBMS normalde mevcut dosyaları yeni veritabanına yükleyen bir yardımcı programa sahiptir.
- Yeni sistem tarafından kullanılmak üzere eski sistemdeki uygulama programlarını dönüştürmek ve kullanmak mümkün olabilir.

TEST

- Hataları bulmak amacıyla veritabanı sistemini çalıştırma süreci.
- Dikkatlice planlanmış test stratejileri ve gerçekçi veriler kullanın.
- Test, hataların olmadığını gösteremez; yalnızca yazılım hatalarının mevcut olduğunu gösterebilir.
- Veritabanı ve uygulama programlarının gereksinimlere göre çalışıyor gibi göründüğünü gösterir.

TEST

- Sistemin kullanılabilirliğini de test etmelidir.
- Bir kullanılabilirlik spesifikasyonuna göre yapılan değerlendirme.
- Kriter örnekleri şunları içerir:
  - Öğrenilebilirlik;
  - Performans;
  - Sağlamlık;
  - Kurtarılabilirlik;
  - Uyarlanabilirlik.

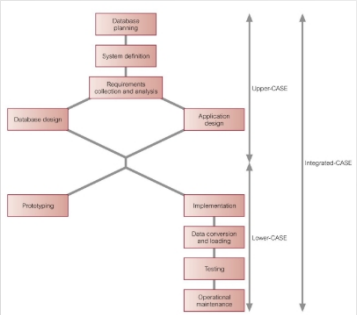
OPERASYONEL BAKIM

- Kurulumun ardından veritabanı sisteminin izlenmesi ve sürdürülmesi süreci.
- Sistem performansının izlenmesi.
- performans düşerse, veritabanının ayarlanması veya yeniden düzenlenmesi gerekebilir.
- Veritabanı uygulamasının bakımı ve yükseltilmesi (gerektiğinde).
- Yeni gereksinimleri veritabanı uygulamasına dahil etme.

VAKA ARAÇLARI

- CASE araçları tarafından sağlanan destek şunları içerir:
  - veritabanı sisteminin verileri hakkında bilgi depolamak için veri sözlüğü;
  - Veri analizini desteklemek için araçlar tasarlamak;
  - Kurumsal veri modeli ile kavramsal ve mantıksal veri modellerinin geliştirilmesine izin veren araçlar;
  - uygulamaların prototiplenmesini sağlayan araçlar.
- Aşağıdaki faydaları sağlayın:
  - Standartlar;
  - Entegrasyon;
  - Standart yöntemler için destek;
  - Tutarlılık;
  - Otomasyon .

VAKA ARAÇLARI VE VERITABANI SISTEMI GELIŞTIRME YAŞAM DÖNGÜSÜ



VERI YÖNETIMI VE VERITABANI YÖNETIMI

- Veri Yöneticisi (DA) ve Veritabanı Yöneticisi (DBA) sırasıyla kurumsal verilerin ve kurumsal veritabanının yönetiminden ve kontrolünden sorumludur.
- DA daha çok veritabanı sistemi geliştirme yaşam döngüsünün erken aşamalarıyla ilgilenirken, DBA daha çok sonraki ilgilenir.

VERI YÖNETIMI

- Aşağıdakileri içeren veri kaynağının yönetimi:
  - veritabanı planlaması,
  - standartların, politikaların ve prosedürlerin ve kavramsal ve mantıksal veritabanı tasarımının geliştirilmesi sürdürülmesi.

VERITABANI YÖNETIMI

- Aşağıdakileri içeren bir veritabanı sisteminin fiziksel olarak gerçekleştirilmesinin yönetimi:
  - fiziksel veritabanı tasarımı ve uygulaması,
  - güvenlik ve bütünlük kontrollerinin ayarlanması,
  - sistem performansının izlenmesi ve veritabanının yeniden düzenlenmesi.

CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ VERİTABANI ANALİZİ

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

GÜNÜN SÖZÜ

- Kavramların analizi, anlama için büyüteç ne ise görme için de odur.
  - Moses Mendelssohn

ANA HATLAR

- Veritabanı uygulama yaşam döngüsünde gerçek bulma teknikleri kullanıldığında.
- Veritabanı uygulama yaşam döngüsünün her aşamasında toplanan olgu türleri.
- Veritabanı uygulama yaşam döngüsünün her aşamasında üretilen dokümantasyon türleri.
- En yaygın kullanılan bilgi toplama teknikleri.
- Her bir bilgi toplama tekniğinin nasıl kullanılacağı ve her birinin avantaj ve dezavantajları.
- DreamHome adında bir emlak kiralama şirketi hakkında.
- Veritabanı uygulama yaşam döngüsünün ilk aşamalarında bilgi toplama tekniklerinin nasıl uygulanacağı.

GERÇEK BULMA TEKNIKLERI

- Gerekli veritabanı uygulamasını oluşturmak için gerekli gerçekleri yakalamak çok önemlidir.
- Bu gerçekler, gerçek bulma teknikleri kullanılarak yakalanır.
- Sistemler, gereksinimler ve tercihler hakkında bilgi toplamak için mülakat ve anket gibi tekniklerin kullanıldığı resmi süreç.

GERÇEK BULMA TEKNİKLERİ NE ZAMAN KULLANILIR?

- Veritabanı uygulama yaşam döngüsü boyunca kullanılan gerçek bulma. Veritabanı planlaması, sistem tanımı ve gereksinim toplama ve analiz aşamaları dahil olmak üzere ilk aşamalar için çok önemlidir.
- Geliştiricinin kurumun ve sistem kullanıcılarının terminolojisi, sorunları, fırsatları, kısıtlamaları, gereksinimleri ve öncelikleri hakkında bilgi edinmesini sağlar.

Stage of database system development lifecycle	Examples of data captured	Examples of documentation produced
Database planning	Aims and objectives of database project	Mission statement and objectives of database system
System definition	Description of major user views (includes job roles or business application areas)	Definition of scope and boundary of database application; definition of user views to be supported
Requirements collection and analysis	Requirements for user views; systems specifications, including performance and security requirements	Users' and system requirements specifications
Database design	Users' responses to checking the logical database design; functionality provided by target DBMS	Conceptual/logical database design (includes ER model(s), data dictionary, and relational schema); physical database design
Application design	Users' responses to checking interface design	Application design (includes description of programs and user interface)
DBMS selection	Functionality provided by target DBMS	DBMS evaluation and recommendations
Prototyping	Users' responses to prototype	Modified users' requirements and systems specifications
Implementation	Functionality provided by target DBMS	
Data conversion and loading	Format of current data; data import capabilities of target DBMS	
Testing	Test results	Testing strategies used; analysis of test results
Operational maintenance	Performance testing results; new or changing user and system requirements	User manual; analysis of performance results; modified users' requirements and systems specifications

VERITABANI UYGULAMASI YAŞAM DÖNGÜSÜ SIRASINDA YAKALANAN VERI VE ÜRETILEN DOKÜMANTASYON ÖRNEKLERI

GERÇEK BULMA TEKNIKLERI

- Bir veritabanı geliştiricisi normalde tek bir veritabanı projesi sırasında aşağıdakiler de dahil olmak üzere çeşitli bilgi bulma teknikleri kullanır:
  - belgelerin incelenmesi
  - görüşme
  - kuruluşun faaliyette olduğunu gözlemlemek
  - araştırma
  - anketler

BELGELERIN INCELENMESI

- Yararlı olabilir
  - bir veritabanına olan ihtiyacın nasıl ortaya çıktığına dair bir fikir edinmek için.
  - Kuruluşun sorunla ilişkili bölümünü tanımlamak için.
  - Mevcut sistemi anlamak.

INCELENMESI GEREKEN BELGE TÜRLERINE ÖRNEKLER

Purpose of documentation	Examples of useful sources
Describes problem and need for database	Internal memos, e-mails, and minutes of meetings Employee/customer complaints, and documents that describe the problem Performance reviews/reports
Describes the part of the enterprise affected by problem	Organizational chart, mission statement, and strategic plan of the enterprise Objectives for the part of the enterprise being studied Task/job descriptions Samples of completed manual forms and reports Samples of completed computerized forms and reports
Describes current system	Various types of flowcharts and diagrams Data dictionary Database system design Program documentation User/training manuals

RÖPORTAJ

- En yaygın kullanılan ve normalde en faydalı bilgi toplama tekniğidir. Bireylerden yüz yüze bilgi toplanmasını sağlar.
- Hedefler arasında gerçekleri bulmak, gerçekleri doğrulamak, gerçekleri netleştirmek, heyecan yaratmak, son kullanıcıyı dahil etmek, gereksinimleri belirlemek ve fikir ve görüşleri toplamak yer alır.

MÜLAKAT YAPMANIN AVANTAJLARI VE DEZAVANTAJLARI

Advantages	Disadvantages
Allows interviewee to respond freely and openly to questions	Very time-consuming and costly, and therefore may be impractical
Allows interviewee to feel part of project	Success is dependent on communication skills of interviewer
Allows interviewer to follow up on interesting comments made by interviewee	Success can be dependent on willingness of interviewees to participate in interviews
Allows interviewer to adapt or re-word questions during interview	
Allows interviewer to observe interviewee's body language	

RÖPORTAJ

- Yapılandırılmamış ve olmak üzere iki tür görüşme vardır.
- Açık uçlu sorular, görüşülen kişinin uygun görünen herhangi bir şekilde yanıt vermesine olanak tanır.
- Kapalı uçlu sorular, cevapları ya belirli seçeneklerle ya da kısa, doğrudan yanıtlarla sınırlandırır.

KURULUŞUN FAALIYETTE OLDUĞUNU GÖZLEMLEMEK

- Bir sistemi anlamak için etkili bir teknik.
- Bir kişinin sistem hakkında bilgi edinmek için gerçekleştirdiği faaliyetlere ya da bunları izlemek mümkündür.
- Toplanan verilerin geçerliliği söz konusu olduğunda veya sistemin belirli yönlerinin karmaşıklığı son kullanıcılar tarafından net bir açıklama yapılmasını engellediğinde faydalıdır.

GÖZLEM KULLANMANIN AVANTAJ VE DEZAVANTAJLARI

Advantages	Disadvantages
Allows the validity of facts and data to be checked	People may knowingly or unknowingly perform differently when being observed
Observer can see exactly what is being done	May miss observing tasks involving different levels of difficulty or volume normally experienced during that time period
Observer can also obtain data describing the physical environment of the task	Some tasks may not always be performed in the manner in which they are observed
Relatively inexpensive	May be impractical
Observer can do work measurements	

ARAŞTIRMA

- Uygulama ve problemi araştırmak için faydalıdır.
- Bilgisayar ticaret dergilerini, referans kitaplarını ve interneti (kullanıcı grupları ve ilan panoları dahil) kullanın.
- Başkalarının benzer sorunları nasıl çözdüğü ve sorunu çözecek ya da kısmen çözecek yazılım paketlerinin mevcut olup olmadığı hakkında bilgi verin.

ARAŞTIRMA KULLANMANIN AVANTAJLARI VE DEZAVANTAJLARI

Advantages	Disadvantages
Can save time if solution already exists	Requires access to appropriate sources of information
Researcher can see how others have solved similar problems or met similar requirements	May ultimately not help in solving problem because problem is not documented elsewhere
Keeps researcher up to date with current developments	

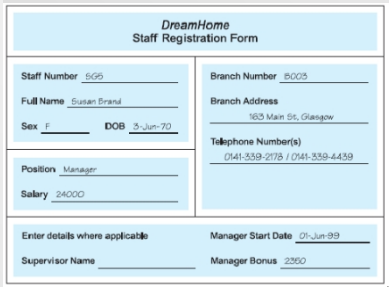
SORU KAĞITLARI

- Anketler, çok sayıda insandan gerçeklerin toplanmasına olanak tanıyan özel amaçlı belgeler olan anketler aracılığıyla gerçekleştirilir ve bu anketlerin yanıtları üzerinde bir miktar kontrol sağlanır.
- Serbest formatlı ve sabit formatlı olmak üzere iki tür soru vardır.

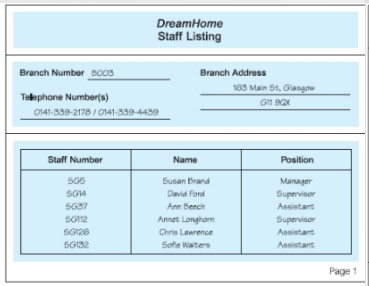
ANKET KULLANMANIN AVANTAJLARI VE DEZAVANTAJLARI

Advantages	Disadvantages
People can complete and return questionnaires at their convenience	Number of respondents can be low, possibly only 5% to 10%
Relatively inexpensive way to gather data from a large number of people	Questionnaires may be returned incomplete
People more likely to provide the real facts as responses can be kept confidential	May not provide an opportunity to adapt or re-word questions that have been misinterpreted
Responses can be tabulated and analyzed quickly	Cannot observe and analyze the respondent's body language

OLGU BULMA TEKNİKLERİNİN KULLANIMI - ÇALIŞILMIŞ BİR ÖRNEK


DreamHome
Staff Registration Form

Staff Number SG5
Full Name Susan Brand
Sex F DOB 3-Jun-70
Position Manager
Salary 24000
Branch Number B003
Branch Address 163 Main St, Glasgow
Telephone Number(s) 0141-339-2178 / 0141-339-4439
Enter details where applicable
Supervisor Name
Manager Start Date 01-Jun-99
Manager Bonus 2360

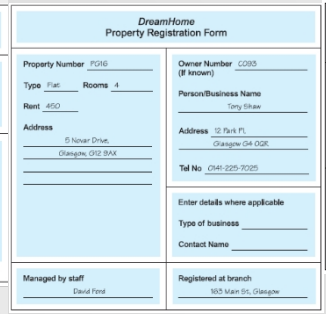

DreamHome
Staff Listing

Branch Number B003
Telephone Number(s) 0141-339-2178 / 0141-339-4439
Branch Address 163 Main St, Glasgow G11 9QX

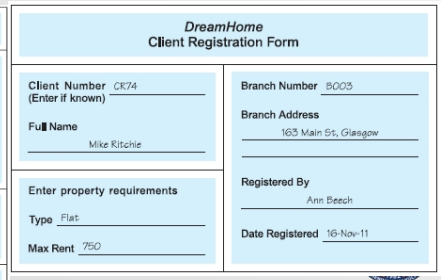
Staff Number	Name	Position
SG5	Susan Brand	Manager
SG14	David Ford	Supervisor
SG37	Ann Beech	Assistant
SG12	Annet Longhorn	Supervisor
SG26	Chris Lawrence	Assistant
SG32	Sofie Walters	Assistant

Page 1

OLGU BULMA TEKNİKLERİNİN KULLANIMI - ÇALIŞILMIŞ BİR ÖRNEK


DreamHome
Property Registration Form

Property Number PG16
Type Flat Rooms 4
Rent 450
Address 5 Novar Drive, Glasgow, G12 9AX
Owner Number CO93 (If known)
Person/Business Name Tony Shaw
Address 12 Park Pl, Glasgow G4 0QR
Tel No 0141-225-7025
Enter details where applicable
Type of business
Contact Name
Managed by staff David Ford
Registered at branch 163 Main St, Glasgow


DreamHome
Client Registration Form

Client Number CR74 (Enter if known)
Full Name Mike Ritchie
Enter property requirements
Type Flat
Max Rent 750
Branch Number B003
Branch Address 163 Main St, Glasgow
Registered By Ann Beech
Date Registered 16-Nov-11

OLGU BULMA TEKNİKLERİNİN KULLANIMI - ÇALIŞILMIŞ BİR ÖRNEK

DreamHome
Property Listing for Week beginning 01/06/13

If you are interested in viewing or renting any of the properties in this list, please contact our branch office as soon as possible.

Branch Address
163 Main St, Glasgow
G11 9QX

Telephone Number(s)
0141-339-2178 / 0141-339-4439

Property No	Address	Type	Rooms	Rent
PG4	6 Lawrence St, Glasgow	Flat	3	350
PG36	2 Manor Rd, Glasgow	Flat	3	375
PG21	18 Dale Road, Glasgow	House	5	600
PG16	5 Novar Drive, Glasgow	Flat	4	450
PG77	100A Apple Lane, Glasgow	House	6	560
PG81	781 Greentree Dr, Glasgow	Flat	4	440

Page 1

DreamHome
Property Viewing Report

Property Numner PG4
Type Flat
Rent 350

Property Address
6 Lawrence St, Glasgow

Client No	Name	Date	Comments
CR76	John Kay	20/04/13	Too remote.
CR56	Aline Stewart	26/05/13	
CR74	Mike Ritchie	11/11/13	
CR62	Mary Tregear	11/11/13	OK, but needs redecoration throughout.

Page 1

OLGU BULMA TEKNİKLERİNİN KULLANIMI - ÇALIŞILMIŞ BİR ÖRNEK

DreamHome Lease
Number 00345810

Client Number CR74
(Enter if known)
Full Name Mike Ritchie
(Please print)
Client Signature _____

Property Number PG16
Property Address
5 Novar Dr, Glasgow

Enter payment details
Monthly Rent 450
Payment Method Cheque
Deposit Paid (Y or N) Yes

Rent Start 01/06/12
Rent Finish 31/05/13
Duration 1 year

DREAMHOME İÇİN MİSYON BİLDİRİMİ VERITABANI SISTEMI

"The purpose of the *DreamHome* database system is to maintain the data that is used and generated to support the property rentals business for our clients and property owners and to facilitate the cooperation and sharing of information between branches."

IÇIN GÖREV HEDEFLERI *DREAMHOME* VERİ TABANI SİSTEMİ

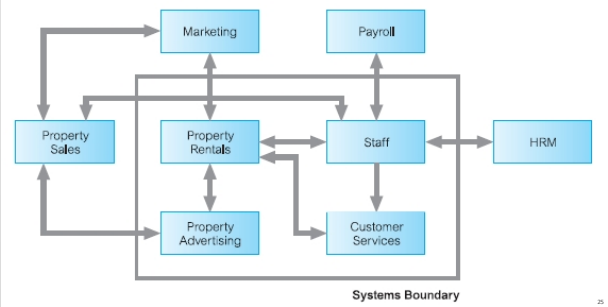
To maintain (enter, update, and delete) data on branches.
To maintain (enter, update, and delete) data on staff.
To maintain (enter, update, and delete) data on properties for rent.
To maintain (enter, update, and delete) data on property owners.
To maintain (enter, update, and delete) data on clients.
To maintain (enter, update, and delete) data on property viewings.
To maintain (enter, update, and delete) data on leases.
To maintain (enter, update, and delete) data on newspaper adverts.

To perform searches on branches.
To perform searches on staff.
To perform searches on properties for rent.
To perform searches on property owners.
To perform searches on clients.
To perform searches on property viewings.
To perform searches on leases.
To perform searches on newspaper adverts.

To track the status of property for rent.
To track the status of clients wishing to rent.
To track the status of leases.

To report on branches.
To report on staff.
To report on properties for rent.
To report on property owners.
To report on clients.
To report on property viewings.
To report on leases.
To report on newspaper adverts.

DREAMHOME IÇIN SISTEM SINIRI VERITABANI SISTEMI



IÇIN BAŞLICA KULLANICI GÖRÜŞLERI *DREAMHOME* VERİ TABANI SİSTEMİ

Data	Access Type	Director	Manager	Supervisor	Assistant
All Branches	Maintain				
	Query	X	X		
	Report	X	X		
Single Branch	Maintain		X		
	Query		X		
	Report		X		
All Staff	Maintain				
	Query	X	X		
	Report	X	X		
Branch Staff	Maintain		X		
	Query		X	X	
	Report		X	X	
All Property	Maintain				
	Query	X			
	Report	X	X		
Branch Property	Maintain		X	X	
	Query		X	X	X
	Report		X	X	X
All Owners	Maintain				
	Query	X			
	Report	X	X		
Branch Owners	Maintain		X	X	
	Query		X	X	X
	Report		X		
All Clients	Maintain				
	Query	X			
	Report	X	X		
Branch Clients	Maintain		X	X	
	Query		X	X	X
	Report		X		
All Viewings	Maintain				
	Query				
	Report				
Branch Viewings	Maintain			X	X
	Query			X	X
	Report			X	X
All Leases	Maintain				
	Query	X			
	Report	X	X		
Branch Leases	Maintain		X	X	
	Query		X	X	X
	Report		X	X	
All Newspapers	Maintain				
	Query	X			
	Report	X	X		
Branch Newspapers	Maintain		X		
	Query		X		
	Report		X		

HER BIRI TARAFINDAN KULLANILAN ANA VERI TÜRLERI ILE KULLANICI GÖRÜNÜMLERININ ÇAPRAZ REFERANSI

	Director	Manager	Supervisor	Assistant
branch	X	X		
staff	X	X	X	
property for rent	X	X	X	X
owner	X	X	X	X
client	X	X	X	X
property viewing			X	X
lease	X	X	X	X
newspaper	X	X		

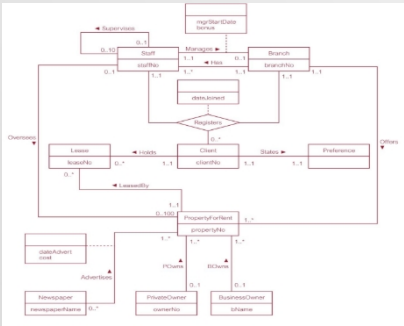
CSE 204 - VERİ TABANI SİSTEMLERİNE GİRİŞ E-R MODELLEME

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- Veritabanı tasarımında Varlık-İlişki (ER) modellemesi nasıl kullanılır?
- ER modeli ile ilgili temel kavramlar.
- Birleşik Modelleme Dili (UML) kullanarak ER modelini görüntülemek için diyagramatik teknik.
- Bağlantı tuzakları adı verilen ER modelleriyle ilgili sorunların nasıl belirleneceği ve çözüleceği.
- Bir gereksinim spesifikasyonundan ER modeli nasıl oluşturulur.

DREAMHOME'UN ŞUBE KULLANICI GÖRÜŞLERININ DIYAGRAMI



ER MODELİNİN KAVRAMLARI

- Varlık türleri
- İlişki türleri
- Nitelikler

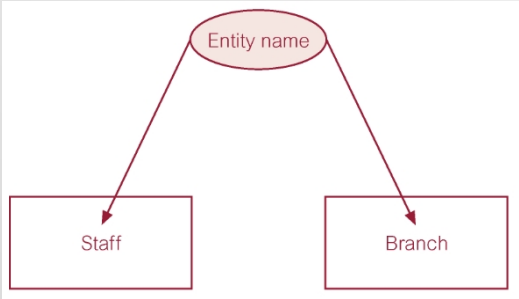
ENTITY TİP

- Varlık türü
  - İşletme tarafından bağımsız bir varlığa sahip olarak tanımlanan, aynı özelliklere sahip nesneler grubu.
- Varlık oluşumu
  - Bir varlık türünün benzersiz olarak tanımlanabilir nesnesi.

VARLIK TÜRLERINE ÖRNEKLER

Physical existence	
Staff	Part
Property	Supplier
Customer	Product
Conceptual existence	
Viewing	Sale
Inspection	Work experience

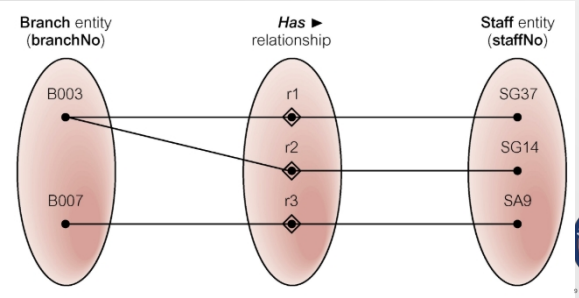
PERSONEL VE ŞUBE VARLIK TÜRLERININ DIYAGRAMI



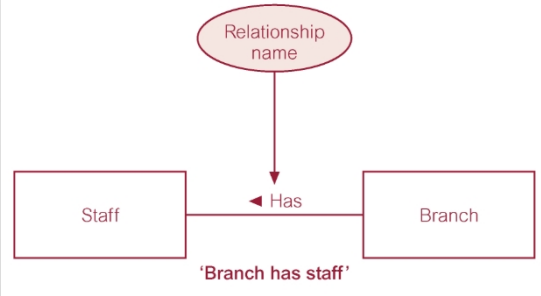
ILIŞKI TÜRLERI

- İlişki türü
  - Varlık türleri arasında anlamlı ilişkiler kümesi.
- İlişki oluşumu
  - Her bir katılımcı varlık türünden bir oluşum içeren benzersiz şekilde tanımlanabilir birliktelik.

HAS ILIŞKI TÜRÜNÜN ANLAMSAL AĞI



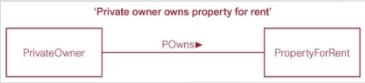
ŞUBE ŞEMASINDA PERSONEL ILIŞKISI *VARDIR*

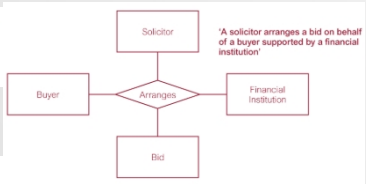


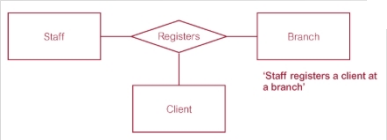
ILIŞKI TÜRLERI

- İlişki Derecesi
  - İlişkiye katılan kuruluşların sayısı.
- Derece ilişkisi :
  - iki ikilidir
  - üç üçlüdür
  - Dört, dördüncüldür.

İLİŞKİLER



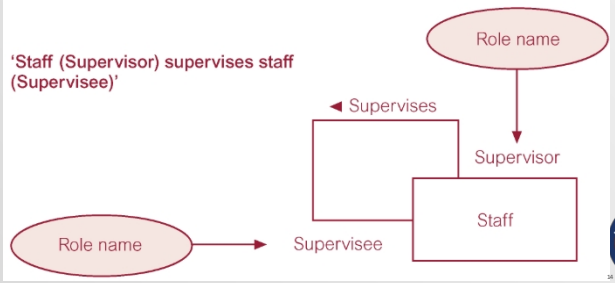




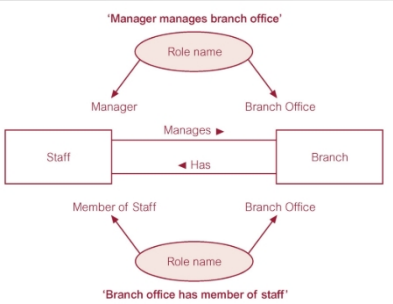
ILIŞKI TÜRLERI

- Yinelemeli İlişki
  - Aynı varlık türünün farklı rollerde birden fazla kez yer aldığı ilişki türü.
- İlişkilere, her bir katılımcı varlık türünün bir ilişkide oynadığı amacı belirtmek rol adları verilebilir.

OLARAK ADLANDIRILAN ÖZYINELEMELI ILIŞKI ROL ADLARI ILE *DENETLER*



ROL ADLARI ILE IKI FARKLI ILIŞKI ARACILIĞIYLA ILIŞKILENDIRILMIŞ VARLIKLAR



ÖZELLİKLER

- Öznitelik
  - Bir varlığın veya ilişki türünün özelliği.
- Öznitelik Etki Alanı
  - Bir veya daha fazla öznitelik için izin verilen değerler kümesi.
- Basit Nitelik
  - Bağımsız bir varlığa sahip tek bir bileşenden oluşan öznitelik.
- Bileşik Öznitelik
  - Her biri bağımsız bir varlığa sahip birden fazla bileşenden oluşan öznitelik.

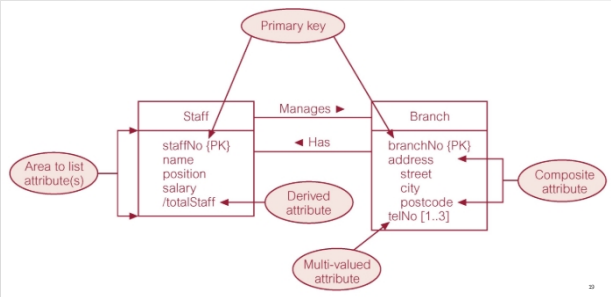
ÖZELLİKLER

- Tek Değerli Öznitelik
  - Bir varlık türünün her bir oluşumu için tek bir değer tutan öznitelik.
- Çok Değerli Öznitelik
  - Bir varlık türünün her bir oluşumu için birden fazla değer tutan öznitelik.
- Türetilmiş Öznitelik
  - Aynı varlık türünde olması gerekmeyen, ilgili bir veya öznitelikler kümesinin değerinden türetilebilen bir değeri temsil eden öznitelik.

ANAHTARLAR

- Aday Anahtarı
  - Bir varlık türünün her bir oluşumunu benzersiz bir şekilde tanımlayan minimum nitelikler kümesi.
- Birincil Anahtar
  - Bir varlık türünün her bir oluşumunu benzersiz bir şekilde tanımlamak için seçilen aday anahtar.
- Kompozit Anahtar
  - İki veya daha fazla öznitelikten oluşan bir aday anahtar.

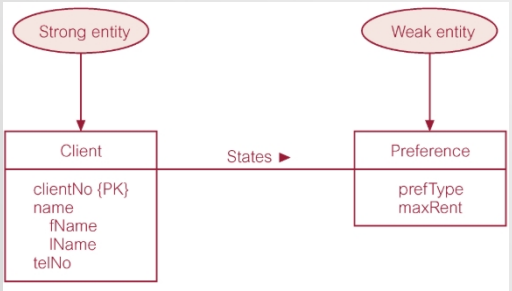
PERSONEL VE ŞUBE VARLIKLARININ VE NITELIKLERININ DIYAGRAMI



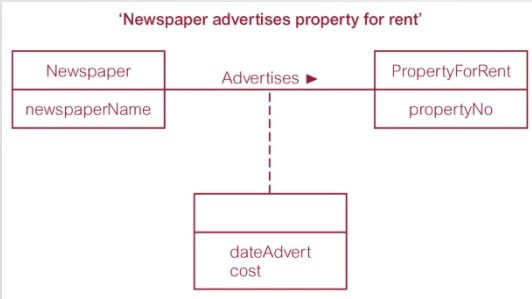
ENTITY TİP

- Güçlü Varlık Türü
  - Varlığı başka bir varlık türüne bağlı olmayan varlık türü.
- Zayıf Varlık Türü
  - Varlığı başka bir varlık türüne bağlı olan varlık türü.

MÜŞTERI OLARAK ADLANDIRILAN GÜÇLÜ VARLIK TÜRÜ VE TERCIH OLARAK ADLANDIRILAN ZAYIF VARLIK TÜRÜ



REKLAM OLARAK ADLANDIRILAN ILIŞKI NITELIKLERI ILE



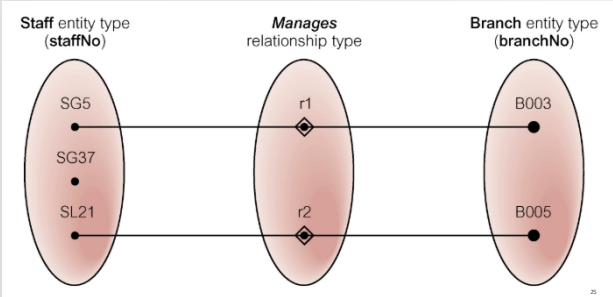
YAPISAL KISITLAMALAR

- İlişkiler üzerindeki ana kısıtlama türü ***çokluk*** olarak adlandırılır.
- Çokluk - belirli bir ilişki aracılığıyla ilişkili bir varlık türünün tek bir oluşumuyla ilişkili olabilecek bir varlık türünün olası oluşumlarının sayısı (veya aralığı).
- Kullanıcı veya şirket tarafından oluşturulan politikaları (***iş kuralları*** olarak adlandırılır) temsil eder.

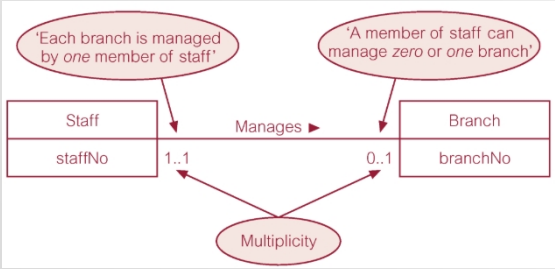
YAPISAL KISITLAMALAR

- İlişkiler için en yaygın derece ikilidir.
- İkili ilişkiler genellikle varlık olarak adlandırılır:
  - bire bir (1:1)
  - bire-çok (1:*)
  - çoktan çoğa (*:*)

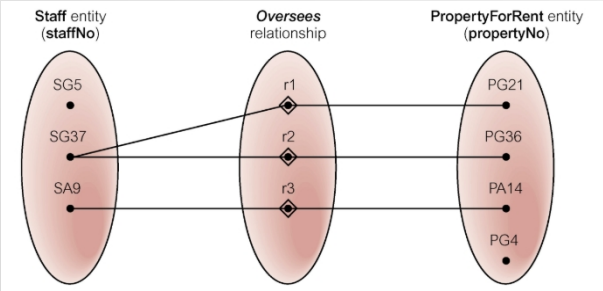
PERSONEL *YÖNETIMININ* SEMANTIK AĞI ŞUBE ILIŞKI TÜRÜ



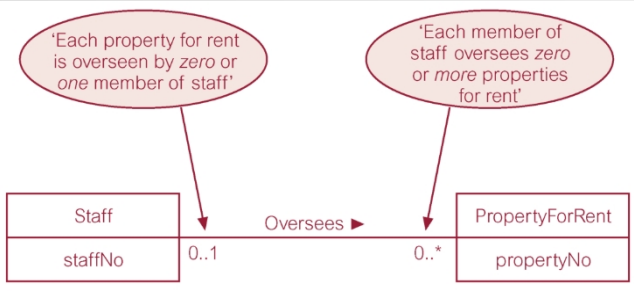
PERSONEL ÇOKLUĞU ŞUBEYI YÖNETIR (1:1) ILIŞKI



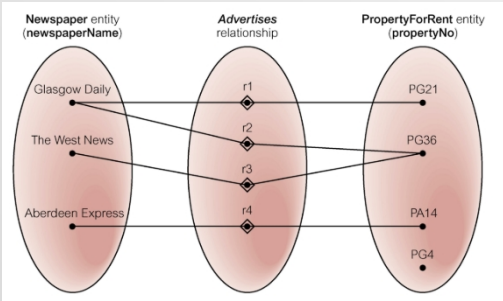
PERSONELIN SEMANTIK AĞI *DENETLER* PROPERTYFORRENT ILIŞKI TÜRÜ



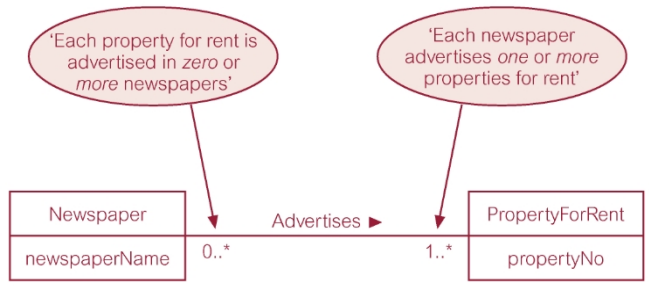
ÇOK SAYIDA PERSONEL *DENETLER* PROPERTYFORRENT (1:*) ILIŞKI TÜRÜ



GAZETE *REKLAMLARININ* ANLAMSAL AĞI PROPERTYFORRENT ILIŞKI TÜRÜ



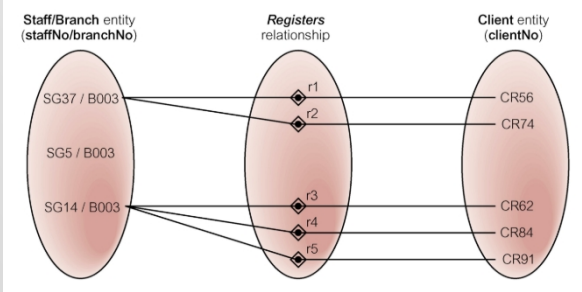
GAZETE *ILANLARININ* ÇOKLUĞU PROPERTYFORRENT (*:*) ILIŞKISI



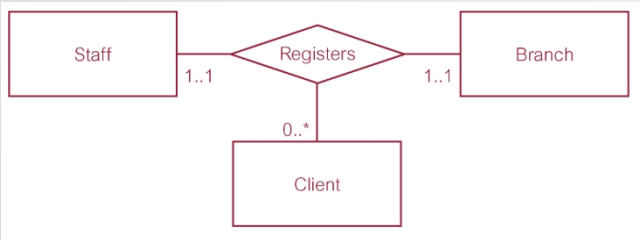
YAPISAL KISITLAMALAR

- Karmaşık İlişkiler için Çokluk
  - Diğer (n-1) değerler sabitken, n-ary ilişkide bir varlık türünün olası oluşumlarının sayısı (veya aralığı).

PERSONEL VE ŞUBE VARLIKLARI IÇIN SABIT DEĞERLERLE ÜÇLÜ *KAYIT* ILIŞKISININ ANLAMSAL AĞI



ÜÇLÜ *KAYITLARIN* ÇOKLUĞU İLİŞKİ



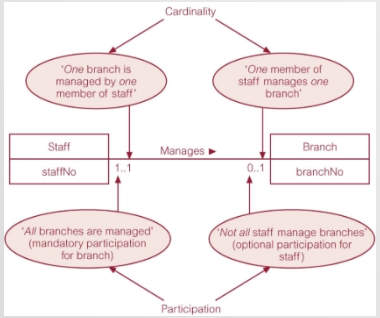
ÇOKLUK KISITLAMALARININ ÖZETI

Alternative ways to represent multiplicity constraints	Meaning
0..1	Zero or one entity occurrence
1..1 (or just 1)	Exactly one entity occurrence
0..* (or just *)	Zero or many entity occurrences
1..*	One or many entity occurrences
5..10	Minimum of 5 up to a maximum of 10 entity occurrences
0, 3, 6–8	Zero or three or six, seven, or eight entity occurrences

YAPISAL KISITLAMALAR

- Çokluk, ilişkiler üzerindeki iki tür kısıtlamadan oluşur: kardinalite ve katılım.
- Kardinalite
  - Belirli bir ilişki türüne katılan bir varlık için olası ilişki oluşumlarının maksimum sayısını tanımlar.
- Katılım
  - Bir ilişkiye tüm varlık oluşumlarının mı yoksa yalnızca bazı varlık oluşumlarının mı katılacağını belirler.

KARDINALITE VE KATILIM KISITLAMALARI OLARAK ÇOKLUK

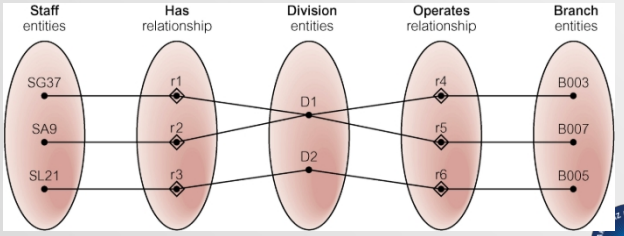


ER MODELLERİ İLE İLGİLİ SORUNLAR

- olarak adlandırılan kavramsal bir veri modeli tasarlarken sorunlar ortaya çıkabilir
  bağlantı tuzakları.
- Genellikle belirli ilişkilerin anlamının yanlış yorumlanmasından kaynaklanmaktadır.
- İki ana bağlantı tuzağı türü *fan* tuzakları ve *uçurum tuzakları* olarak adlandırılır.
- Fan Kapanı
  - Bir modelin varlık türleri arasında bir ilişkiyi temsil ettiği, ancak belirli varlık oluşumları yolun belirsiz olduğu durumlarda.
- Uçurum Tuzağı
  - Bir modelin varlık türleri arasında bir ilişki olduğunu öne sürdüğü, ancak belirli varlık oluşumları arasında bir yol bulunmadığı durumlar.

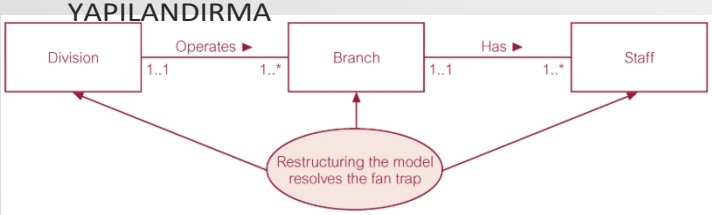
BIR FAN TUZAĞI ÖRNEĞI

FAN KAPANI ILE ER MODELININ ANLAMSAL AĞI

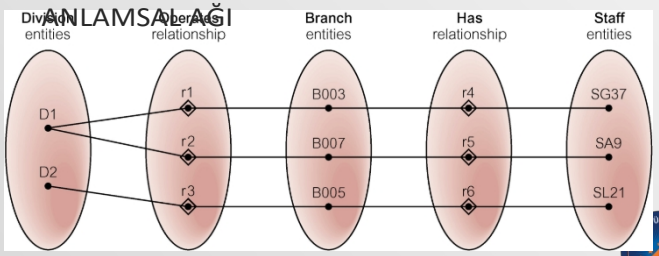


- SG37 numaralı personel hangi şubede çalışıyor?

FAN KAPAĞINI ÇIKARMAK IÇIN ER MODELINI YENIDEN YAPILANDIRMA

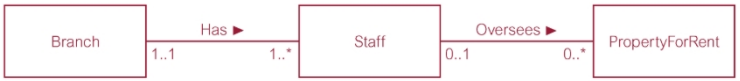


FAN TUZAĞI KALDIRILMIŞ YENIDEN YAPILANDIRILMIŞ ER MODELININN ANLAMSAL AĞI

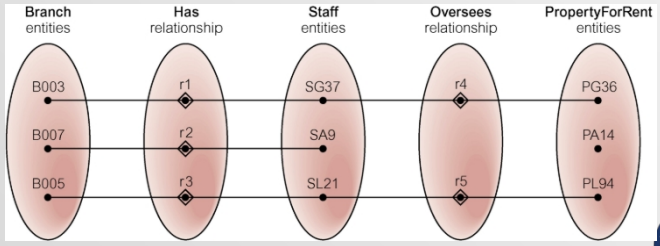


- SG37, B003 şubesinde çalışıyor.

BIR UÇURUM TUZAĞI ÖRNEĞI

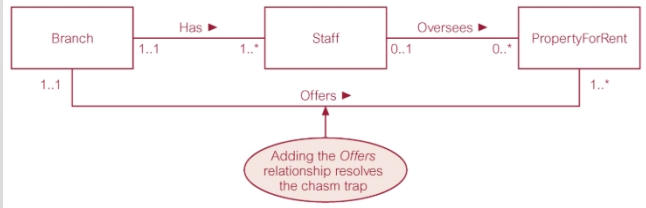


UÇURUM TUZAĞI ILE ER MODELININ ANLAMSAL AĞI

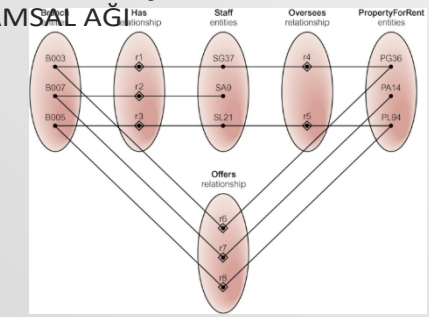


- PA14 mülkü hangi şubeden temin edilebilir?

ER MODELI UÇURUM TUZAĞINI ORTADAN KALDIRMAK IÇIN YENIDEN YAPILANDIRILDI



UÇURUM TUZAĞI KALDIRILMIŞ YENIDEN YAPILANDIRILMIŞ ER MODELININN ANLAMSAL AĞI



BÖLÜM 13 - ANA HATLAR

- ER modelinin temel kavramlarının sınırlamaları ve ek veri modelleme kavramları kullanarak daha karmaşık uygulamaları temsil etme gereksinimleri.
- Geliştirilmiş ER (EER) modelinin en kullanışlı ek veri modelleme kavramı özelleştirme/genelleştirme olarak adlandırılır.
- UML kullanarak bir EER diyagramında uzmanlaşma/genelleşmeyi göstermek için diyagramatik bir teknik.

GELIŞTIRILMIŞ VARLIK-ILIŞKI MODELI

- 1980'lerden bu yana, daha zorlu gereksinimleri olan yeni veritabanı uygulamalarının ortaya çıkışında bir artış olmuştur.
- ER modellemesinin temel kavramları, daha yeni ve karmaşık uygulamaların gereksinimlerini temsil etmek için yeterli değildir.
- Yanıt, ek 'semantik' modelleme kavramlarının geliştirilmesidir.
- Anlamsal kavramlar orijinal ER modeline dahil edilir ve Geliştirilmiş Varlık-İlişki (EER) modeli olarak adlandırılır.
- EER modelinin ek kavram örnekleri uzmanlaşma / genelleme olarak adlandırılır.

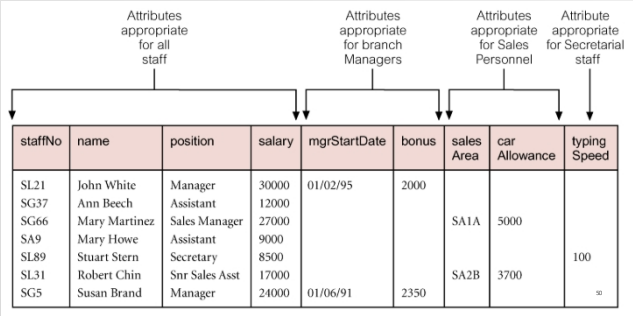
UZMANLAŞMA / GENELLEME

- Süper Sınıf
  - Oluşumlarının bir veya daha fazla farklı alt gruplamasını içeren bir varlık türü.
- Alt sınıf
  - Bir varlık türünün oluşumlarının farklı bir alt gruplaması.
- Üst sınıf/alt sınıf ilişkisi bire birdir (1:1).
- Üst sınıf, örtüşen veya farklı alt sınıflar içerebilir.
- Bir üst sınıfın tüm üyelerinin bir alt sınıfın üyesi olması gerekmez.

UZMANLAŞMA / GENELLEME

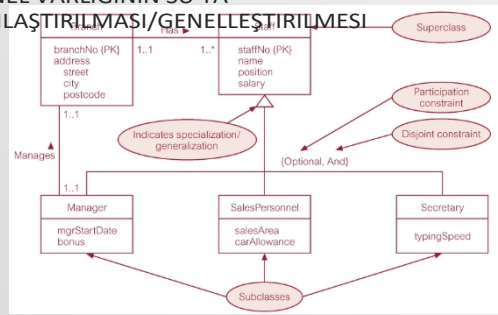
- Öznitelik Kalıtımı
  - Alt sınıftaki bir varlık, üst sınıftakiyle aynı 'gerçek dünya' nesnesini temsil eder ve üst sınıfla ilişkili olanların yanı sıra alt sınıfa özgü niteliklere de sahip olabilir.
- Uzmanlık
  - Bir varlığın üyeleri arasındaki farklılıkları, ayırt edici özelliklerini belirleyerek en üst düzeye çıkarma süreci.
- Genelleme
  - Ortak özelliklerini belirleyerek varlıklar arasındaki farklılıkları en aza indirme süreci.

ALLSTAFF ILIŞKISI TÜM PERSONELIN BILGILERINI TUTAR

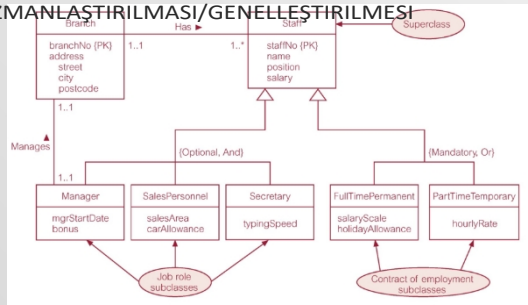


staffNo	name	position	salary	mgrStartDate	bonus	sales Area	car Allowance	typing Speed
SL21	John White	Manager	30000	01/02/95	2000			
SG37	Ann Beech	Assistant	12000					
SG66	Mary Martinez	Sales Manager	27000			SA1A	5000	
SA9	Mary Howe	Assistant	9000					
SL89	Stuart Stern	Secretary	8500					100
SL31	Robert Chin	Snr Sales Asst	17000			SA2B	3700	
SG5	Susan Brand	Manager	24000	01/06/91	2350			

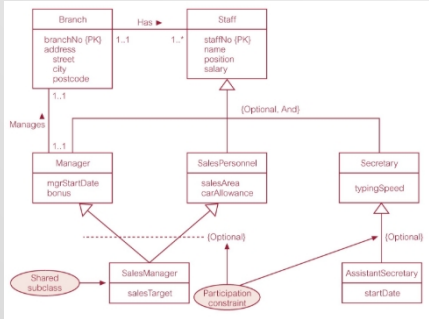
PERSONEL VARLIĞININ SU'YA UZMANLAŞTIRILMASI/GENELLEŞTIRILMESI



PERSONEL VARLIĞININ IŞ ROLLERINE VE IŞ SÖZLEŞMELERINE GÖRE UZMANLAŞTIRILMASI/GENELLEŞTIRILMESI



PAYLAŞILAN ALT SINIF VE KENDI ALT SINIFINA SAHIP ALT SINIF ILE EER DIYAGRAMI



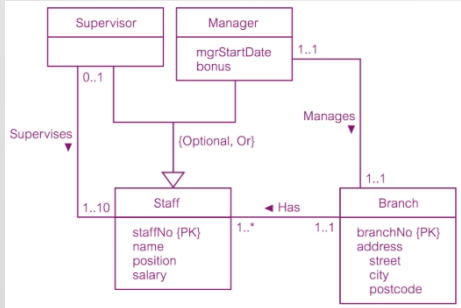
UZMANLAŞMA / GENELLEME ÜZERINDEKI KISITLAMALAR

- Bir uzmanlaştırma/genelleştirme için geçerli olabilecek iki kısıtlama:
  - katılım kısıtlamaları
  - ayrık kısıtlamalar.
- Katılım kısıtlaması
  - Üst sınıftaki her üyenin bir alt sınıfın üyesi olarak yer alması gerekip gerekmediğini belirler.
  - Zorunlu veya isteğe bağlı olabilir.

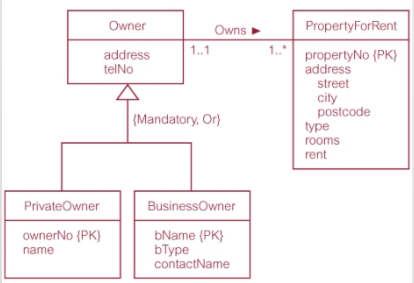
UZMANLAŞMA / GENELLEME ÜZERINDEKI KISITLAMALAR

- Ayrık kısıtlama
  - Alt sınıfların üyeleri arasındaki ilişkiyi tanımlar ve bir üst sınıfın üyesinin bir veya birden fazla alt sınıfın üyesi olup olamayacağını belirtir.
  - Ayrık veya ayrık olmayan olabilir.
- Uzmanlaşma ve genelleme kısıtlamalarının dört kategorisi vardır:
  - zorunlu ve ayrık
  - isteğe bağlı ve ayrık
  - zorunlu ve eşlenik olmayan
  - isteğe bağlı ve ayrık olmayan.

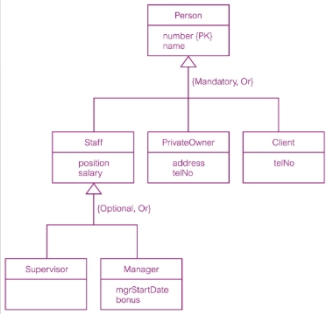
DREAMHOME ÇALIŞMA ÖRNEĞI - SUPERVISOR VE MANAGER ALT SINIFLARI ILE PERSONEL ÜST SINIFI



DREAMHOME ÇALIŞMA ÖRNEĞİ - PRIVATEOWNER VE BUSINESSOWNER ALT SINIFLARI İLE OWNER ÜST SINIFI



DREAMHOME ÇALIŞMA ÖRNEĞİ - PRIVATEOWNER VE BUSINESSOWNER ALT SINIFLARI İLE OWNER ÜST SINIFI



CSE 204 - VERİ TABANI SİSTEMLERİNE GİRİŞ NORMALLEŞME

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- Normalleştirmenin amacı.
- İlişkisel bir veritabanı tasarlarken normalleştirmenin nasıl kullanılabileceği.
- Temel ilişkilerdeki gereksiz verilerle ilişkili potansiyel sorunlar.
- Nitelikler arasındaki ilişkiyi tanımlayan işlevsel bağımlılık kavramı.
- Normalleştirmede kullanılan fonksiyonel bağımlılıkların özellikleri.
- Belirli bir ilişki için işlevsel bağımlılıkların nasıl belirleneceği.

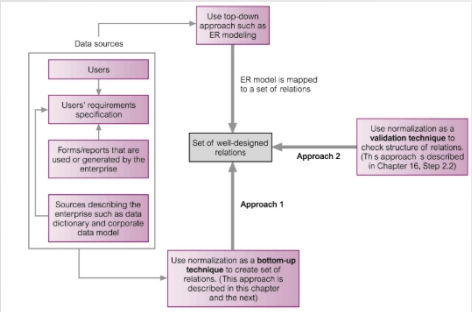
ANA HATLAR

- İşlevsel bağımlılıkların bir ilişki için birincil anahtarı nasıl belirlediği.
- Normalleşme süreci nasıl yürütülür?
- Normalleştirme, nitelikleri bilinen bir normal formda olan ilişkiler halinde gruplamak için işlevsel bağımlılıkları nasıl kullanır?
- En yaygın kullanılan normal formlar olan Birinci Form (1NF), İkinci Normal Form (2NF) ve Üçüncü Normal Form NF) nasıl tanımlanır.
- 1NF, 2NF veya 3NF kurallarını ihlal eden ilişkilerle ilgili sorunlar.
- Normalleştirme kullanılarak bir formda gösterilen özniteliklerin 3NF ilişkileri olarak nasıl gösterileceği.

NORMALLEŞTİRMENİN AMACI

- Normalleştirme, bir kuruluşun veri gereksinimlerini destekleyen bir dizi uygun ilişki üretmeye yönelik bir tekniktir.
- Uygun bir ilişki setinin özellikleri şunları içerir:
  - Kurumun veri gereksinimlerini desteklemek için gerekli minimum sayıda öznitelik;
  - Yakın mantıksal ilişkiye sahip nitelikler aynı ilişkide bulunur;
  - Yabancı anahtarların tamamını veya bir kısmını oluşturan öznitelikler hariç olmak üzere, her özniteliğin yalnızca bir kez temsil minimum fazlalık.
- Uygun ilişkiler kümesine sahip bir veritabanı kullanmanın faydaları, veritabanının
  - kullanıcının verilere erişmesini ve bunları muhafaza etmesini kolaylaştırır;
  - bilgisayarda minimum depolama alanı kaplar.

NORMALLEŞTIRME VERITABANI TASARIMINI NASIL DESTEKLER



VERI FAZLALIĞI VE GÜNCELLEME ANOMALILERI

- İlişkisel veritabanı tasarımının temel amacı, veri fazlalığını en aza indirmek için öznitelikleri ilişkiler halinde gruplamaktır.
- Uygulanan veritabanı için potansiyel faydalar şunlardır:
  - Veritabanında depolanan verilerdeki güncellemeler minimum sayıda işlemle gerçekleştirilir ve böylece veri tutarsızlığı fırsatları azaltılır.
  - Temel ilişkilerin gerektirdiği dosya depolama alanının azaltılması ve böylece maliyetlerin en aza indirilmesi.
- Veri fazlalığı ile ilgili sorunlar, Personel ve Şube ilişkilerinin PersonelŞube ilişkisi ile karşılaştırılmasıyla gösterilmiştir.

VERI FAZLALIĞI VE GÜNCELLEME ANOMALILERI

Staff

staffNo	sName	position	salary	branchNo
SL21	John White	Manager	30000	B005
SG37	Ann Beech	Assistant	12000	B003
SG14	David Ford	Supervisor	18000	B003
SA9	Mary Howe	Assistant	9000	B007
SG5	Susan Brand	Manager	24000	B003
SL41	Julie Lee	Assistant	9000	B005

Branch

branchNo	bAddress
B005	22 Deer Rd, London
B007	16 Argyll St, Aberdeen
B003	163 Main St, Glasgow

StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
SL21	John White	Manager	30000	B005	22 Deer Rd, London
SG37	Ann Beech	Assistant	12000	B003	163 Main St, Glasgow
SG14	David Ford	Supervisor	18000	B003	163 Main St, Glasgow
SA9	Mary Howe	Assistant	9000	B007	16 Argyll St, Aberdeen
SG5	Susan Brand	Manager	24000	B003	163 Main St, Glasgow
SL41	Julie Lee	Assistant	9000	B005	22 Deer Rd, London

VERI FAZLALIĞI VE GÜNCELLEME ANOMALILERI

- StaffBranch ilişkisi gereksiz verilere sahiptir; bir şubenin ayrıntıları her personel üyesi için tekrarlanır
- Buna karşılık, şube bilgileri Şube ilişkisinde her şube için yalnızca bir kez görünür ve Personel ilişkisinde her bir personel üyesinin bulunduğu yeri temsil etmek için yalnızca şube numarası (şubeNo) tekrarlanır.
- Gereksiz bilgi içeren ilişkiler potansiyel olarak güncelleme anomalilerinden muzdarip olabilir.
- Güncelleme anomalisi türleri şunları içerir
  - Yerleştirme
  - Silme
  - Değişiklik

KAYIPSIZ BIRLEŞTIRME VE BAĞIMLILIK KORUMA ÖZELLIKLERI

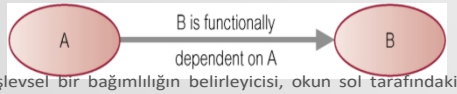
- Ayrıştırmanın iki önemli özelliği.
  - Kayıpsız birleştirme özelliği, daha küçük ilişkilerdeki karşılık gelen örneklerden orijinal ilişkinin herhangi bir örneğini bulmamızı sağlar.
  - Bağımlılık koruma özelliği, daha küçük ilişkilerin her biri üzerinde bazı kısıtlamalar uygulayarak orijinal ilişki üzerinde kısıtlama uygulamamızı sağlar.

IŞLEVSEL BAĞIMLILIKLAR

- Normalleştirme ile ilişkili önemli kavram.
- Fonksiyonel bağımlılık, nitelikler arasındaki ilişkiyi tanımlar.
- Örneğin, A ve B R ilişkisinin nitelikleri ise, R'deki her A değeri R'deki tam olarak bir B değeri ile ilişkiliyse, B A'işlevsel olarak bağımlıdır (A -> B olarak gösterilir).

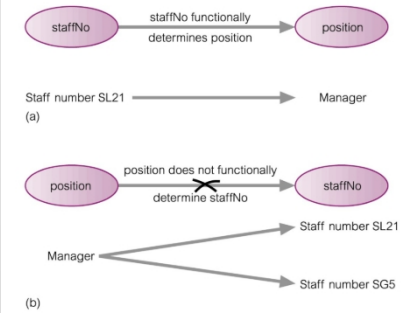
FONKSIYONEL BAĞIMLILIKLARIN ÖZELLIKLERI

- Bir ilişkideki niteliklerin anlam veya semantiğinin özelliği.
- Diyagramatik gösterim.



- İşlevsel bir bağımlılığın belirleyicisi, okun sol tarafındaki nitelik veya nitelikler grubunu ifade eder.

ÖRNEK BIR IŞLEVSEL BAĞIMLILIK



TÜM ZAMANLAR IÇIN GEÇERLI OLAN ÖRNEK IŞLEVSEL BAĞIMLILIK

- Staff ilişkisinin staffNo sName niteliklerinde gösterilen değerleri göz önünde bulundurun (slayt 7).
- Örnek verilere dayanarak, aşağıdaki işlevsel bağımlılıkların geçerli olduğu görülmektedir.
  - personelNo→ sAdı
  - sAdı→ personelNo
- Ancak, Staff ilişkisinin staffNo sName nitelikleri için olası tüm değerler için doğru kalan tek işlevsel bağımlılık şudur:
  - personelNo→ sAdı

FONKSIYONEL BAĞIMLILIKLARIN ÖZELLIKLERI

- Belirleyiciler, sağ taraftaki nitelik(ler) ile işlevsel bağımlılığı sürdürmek için gerekli en az sayıda sahip olmalıdır.
- Bu gereklilik tam işlevsel bağımlılık olarak adlandırılır.
- Tam işlevsel bağımlılık, A ve B bir ilişkinin nitelikleri ise, B işlevsel olarak A'ya bağımlıysa ancak A'nın herhangi bir uygun alt kümesine bağımlı değilse, B'nin işlevsel olarak A'ya tamamen bağımlı olduğunu gösterir.

ÖRNEK TAM IŞLEVSEL BAĞIMLILIK

- Personel ilişkisinde mevcuttur (slayt 7).
  - staffNo, sName→ branchNo
- True - her (staffNo, sName) değeri tek bir branchNo değeri ile ilişkilendirilir.
- Ancak, branchNo aynı zamanda işlevsel olarak (staffNo, sName) alt kümesine, yani staffNo'ya bağımlıdır. Yukarıdaki örnek kısmi bir bağımlılıktır.

FONKSIYONEL BAĞIMLILIKLARIN ÖZELLIKLERI

- Normalleştirmede kullanılan işlevsel bağımlılıkların temel özellikleri:
  - Fonksiyonel bağımlılığın sol tarafındaki (belirleyici) nitelik(ler) ile sağ tarafındaki nitelik(ler) arasında bire bir ilişki vardır.
  - Her zaman için tutar.
  - Belirleyici, sağ taraftaki nitelik(ler) ile bağımlılığı sürdürmek için gerekli minimum sayıda niteliğe sahiptir.

GEÇİŞLI BAĞIMLILIKLAR

- Geçişli bir bağımlılığı tanımak önemlidir çünkü bir ilişkideki varlığı potansiyel olarak güncelleme anomalilerine neden olabilir.
- Geçişli bağımlılık, A, B ve C'nin bir ilişkinin nitelikleri olduğu bir durumu tanımlar; öyle ki A → B ve B → C ise C, B üzerinden A'ya geçişli olarak bağımlıdır (A'nın B veya C'ye işlevsel olarak bağımlı olmaması şartıyla).

ÖRNEK GEÇIŞLI BAĞIMLILIK

- StaffBranch ilişkisindeki işlevsel bağımlılıkları göz önünde bulundurun (slayt 7)
  - personelNo→ sAdı, pozisyon, maaş, şubeNo, bAdres
  - şubeNo→ bAdres
- Geçişli bağımlılık, şubeNo→ bAdres, şubeNo aracılığıyla personelNo üzerinde mevcuttur.

NORMALLEŞME SÜRECI

- Bir ilişkinin birincil anahtarına ve bu ilişkinin nitelikleri arasındaki işlevsel bağımlılıklara dayalı olarak analiz edilmesine resmi teknik.
- Genellikle bir dizi adım olarak yürütülür. Her adım, bilinen özelliklere sahip belirli bir normal forma karşılık gelir.

FONKSIYONEL BAĞIMLILIKLARIN BELIRLENMESI

- Her bir niteliğin anlamı ve nitelikler arasındaki ilişkiler iyi anlaşılırsa, bir dizi nitelik arasındaki tüm işlevsel bağımlılıkları tanımlamak nispeten basittir.
- Bu bilgi, kullanıcılarla yapılan görüşmeler ve/veya kullanıcı gereksinimleri spesifikasyonu gibi dokümantasyon şeklinde işletme tarafından sağlanmalıdır.
- Ancak, kullanıcılar danışmak için müsait değilse ve/veya belgeler eksikse, veritabanı uygulamasına bağlı olarak, veritabanı tasarımcısının eksik bilgileri sağlamak için sağduyusunu ve/veya deneyimini kullanması gerekebilir.

ÖRNEK - STAFFBRANCH ILIŞKISI IÇIN BIR DIZI IŞLEVSEL BAĞIMLILIĞIN TANIMLANMASI

- StaffBranch ilişkisindeki niteliklerin semantiğini inceleyin (slayt 7). Sahip olunan pozisyonun ve şubenin bir personel üyesinin maaşını belirlediğini varsayınız.
- Yeterli bilgi mevcutsa, StaffBranch ilişkisi için işlevsel bağımlılıkları şu şekilde tanımlayın:
  - personelNo→ sAdı, pozisyon, maaş, şubeNo, bAdres
  - şubeNo→ bAdres
  - bAdres→ şubeNo
  - şubeNo, pozisyon→ maaş
  - bAdres, pozisyon→ maaş

ÖRNEK - IŞLEVSEL BAĞIMLILIKLARI BELIRLEMEK IÇIN ÖRNEK VERILERIN KULLANILMASI.

- Örnek ilişkisinde (slayt 23) A, B, C, D ve E gösterilen niteliklere ait verileri göz önünde bulundurun.
- İlişkide gösterilen örnek veri değerlerinin A, B, C, D ve E niteliklerinin sahip olabileceği tüm olası değerleri temsil ettiğini tespit etmek önemlidir.

GÜNÜN SÖZÜ

- Asla tahmin etmem. Bu, mantıksal zarar veren şok edici bir alışkanlıktır.
  - Arthur Conan Doyle

ÖRNEK - IŞLEVSEL BAĞIMLILIKLARI BELIRLEMEK IÇIN ÖRNEK VERILERIN KULLANILMASI.

Sample Relation

A	B	C	D	E
a	b	z	w	q
e	b	r	w	p
a	d	z	w	t
e	d	r	w	q
a	f	z	s	t
e	f	r	s	t

ÖRNEK - IŞLEVSEL BAĞIMLILIKLARI BELIRLEMEK IÇIN ÖRNEK VERILERIN KULLANILMASI.

- Örnek ilişkisinde A ile E nitelikleri arasındaki fonksiyon bağımlılıkları.
  - A→ C (fd1)
  - C→ A (fd2)
  - B→ D (fd3)
  - A, B→ E (fd4)

IŞLEVSEL BAĞIMLILIKLARI KULLANARAK BIR ILIŞKI IÇIN BIRINCIL ANAHTARI BELIRLEME

- Bir ilişki için bir dizi işlevsel bağımlılık belirlemenin temel amacı, bir ilişki üzerinde tutulması gereken bütünlük kısıtlamaları kümesini belirtmektir.
- İlk olarak dikkate alınması gereken önemli bir bütünlük kısıtlaması, aday anahtarların tanımlanması ve bunlardan birinin ilişki için birincil anahtar olarak

ÖRNEK - STAFFBRANCH ILIŞKISI IÇIN BIRINCIL ANAHTAR TANIMLAMA

- StaffBranch ilişkisinin beş işlevsel bağımlılığı vardır (slayt 21).
- Belirleyiciler personelNo, şubeNo, bAdres, (şubeNo, pozisyon) ve (bAdres, pozisyon) şeklindedir.
- Tüm aday anahtar(lar)ı tanımlamak için, bu ilişkideki her bir tuple'ı benzersiz bir şekilde tanımlayan özniteliği (veya öznitelik grubunu) tanımlayın.
- Bir aday anahtarın parçası olmayan tüm nitelikler işlevsel olarak anahtara bağımlı olmalıdır.
- StaffBranch ilişkisi için tek aday anahtar ve dolayısıyla birincil anahtar staffNo'dur, çünkü ilişkinin diğer tüm nitelikleri işlevsel olarak staffNo'ya bağlıdır.

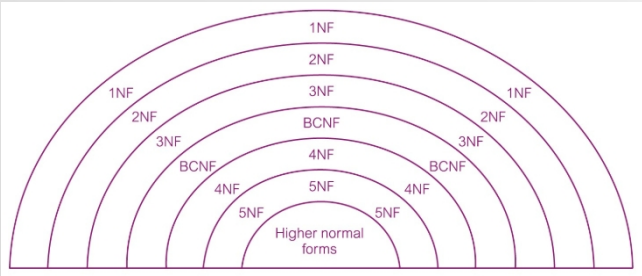
ÖRNEK - ÖRNEK ILIŞKI IÇIN BIRINCIL ANAHTARIN TANIMLANMASI

- Örnek ilişkinin dört işlevsel bağımlılığı vardır (slayt 21).
- Örnek ilişkisindeki belirleyiciler A, B, C ve (A, B)'dir. Ancak, ilişkinin diğer tüm niteliklerini işlevsel olarak belirleyen tek belirleyici (A, B)'dir.
- (A, B) bu ilişki için birincil anahtar olarak tanımlanır.

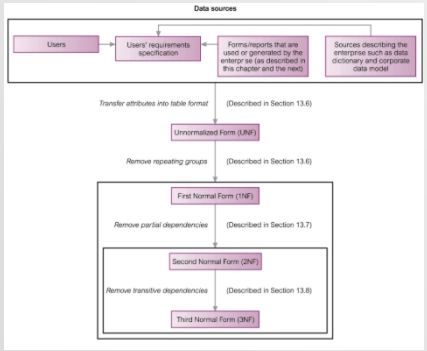
NORMALLEŞME SÜRECI

- Normalleştirme ilerledikçe, ilişkiler format olarak giderek daha kısıtlı (daha güçlü) hale gelir ve ayrıca güncelleme anormalliklerine karşı daha az savunmasız olur.

NORMALLEŞME SÜRECI



NORMALLEŞME SÜRECI



NORMALLEŞTIRILMEMIŞ FORM (UNF)

- Bir veya daha fazla tekrar eden grup içeren bir tablo.
- Normalleştirilmemiş bir tablo oluşturmak için
  - Bilgi kaynağındaki (örn. form) verileri sütun ve satırlarla tablo formatına dönüştürün.

BIRINCI NORMAL FORM (1NF)

- Her satır ve sütunun kesişiminin bir ve yalnızca bir değer içerdiği bir ilişki.

UNF'DEN 1NF'YE

- Normalleştirilmemiş tablo anahtar görevi görecek bir öznitelik veya öznitelik grubu atayın.
- Normalleştirilmemiş tabloda anahtar nitelik(ler) için tekrar grup(lar)ı belirleyiniz.
- Yinelenen grubu şu şekilde kaldırın
  - Yinelenen verileri içeren satırların boş sütunlarına uygun verilerin girilmesi ('tablonun düzleştirilmesi').
  - Ya da
  - Yinelenen verilerin orijinal anahtar öznitelik(ler)inin bir kopyasıyla birlikte ayrı bir ilişkiye yerleştirilmesi.

IKINCI NORMAL FORM (2NF)

- Tam işlevsel bağımlılık kavramına dayanır.
- Tam işlevsel bağımlılık, eğer
  - A ve B bir ilişkinin nitelikleridir,
  - B, A'ya işlevsel olarak bağımlıysa ancak A'nın herhangi bir uygun alt kümesine bağımlı değilse B, A'ya tamamen bağımlıdır.
- 1NF'de olan ve birincil anahtar olmayan her özniteliğin birincil anahtara işlevsel olarak tamamen bağımlı olduğu bir ilişki.

1NF'DEN 2NF'YE

- 1NF ilişkisi için birincil anahtarı tanımlayın.
- İlişkideki işlevsel bağımlılıkları tanımlayın.
- Birincil anahtar üzerinde kısmi bağımlılıklar varsa, bunları belirleyicilerinin bir kopyasıyla birlikte yeni bir ilişkiye yerleştirerek kaldırın.

ÜÇÜNCÜ NORMAL FORM (3NF)

- Geçişli bağımlılık kavramına dayanır.
- Geçişli Bağımlılık aşağıdaki durumlarda söz konusudur
  - A, B ve C bir ilişkinin nitelikleridir, öyle ki A → B ve B → C ise,
  - o zaman C, B aracılığıyla A'ya geçişli olarak bağımlıdır (A'nın işlevsel olarak B veya C'ye bağımlı olmaması koşuluyla).
- 1NF ve 2NF içinde olan ve birincil anahtar olmayan hiçbir niteliğin birincil anahtara geçişli olarak bağımlı olmadığı bir ilişki.

2NF'DEN 3NF'YE

- 2NF ilişkisindeki birincil anahtarı tanımlayın.
- İlişkideki işlevsel bağımlılıkları tanımlayın.
- Birincil anahtar üzerinde geçişli bağımlılıklar varsa, bunları baskınlarının bir kopyasıyla birlikte yeni bir ilişkiye yerleştirerek kaldırın.

2NF VE 3NF'NIN GENEL TANIMLARI

- İkinci normal form (2NF)
  - Birinci normal formda olan ve birincil anahtar olmayan her özniteliğin herhangi bir aday anahtara işlevsel olarak tamamen bağımlı olduğu bir ilişki.
- Üçüncü normal form (3NF)
  - Birinci ve ikinci normal formda olan ve hiçbir birincil anahtar olmayan özniteliğin herhangi bir aday anahtara geçişli olarak bağımlı olmadığı bir ilişki.

CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ GELIŞMIŞ NORMALLEŞTIRME

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- Çıkarım kurallarının bir ilişki için tüm işlevsel bağımlılıkların bir kümesini nasıl tanımlayabileceği.
- Armstrong aksiyomları olarak adlandırılan çıkarım kuralları, bir ilişki için tüm bağımlılıklar kümesinden minimum yararlı işlevsel bağımlılıklar kümesini nasıl belirleyebilir?
- Boyce-Codd Normal Formu (BCNF), Dördüncü Normal Form (4NF) ve Beşinci Normal Formu (5NF) içeren Üçüncü Normal Formun 3NF) ötesine geçen normal formlar.
- Boyce-Codd Normal Formu (BCNF) nasıl tanımlanır.
- Normalleştirme kullanılarak bir raporda BCNF ilişkileri olarak gösterilen öznitelikler nasıl temsil edilir.

ANA HATLAR

- Çok değerli bağımlılık kavramı ve Dördüncü Normal Form (4NF).
- 4NF kurallarını ihlal eden ilişkilerle ilgili sorunlar.
- 4NF kurallarını ihlal bir ilişkiden 4NF ilişkileri nasıl oluşturulur.
- Birleştirme bağımlılığı kavramı ve Beşinci Normal Form (5NF).
- 5NF kurallarını ihlal eden ilişkilerle ilgili sorunlar.
- 5NF kurallarını çiğneyen bir ilişkiden 5NF ilişkileri nasıl oluşturulur.

IŞLEVSEL BAĞIMLILIKLAR HAKKINDA DAHA FAZLA BILGI

- Belirli bir ilişki için işlevsel bağımlılıkların tam kümesi çok büyük olabilir.
- Kümeyi yönetilebilir bir boyuta indirebilecek bir yaklaşım bulmak önemlidir.

IŞLEVSEL BAĞIMLILIKLAR IÇIN ÇIKARIM KURALLARI

- Bir ilişki için, o ilişkinin tüm bağımlılıklar kümesinden (Y olarak temsil edilir) daha küçük olan ve Y'deki her işlevsel bağımlılığın X'teki işlevsel bağımlılıklar tarafından ima edildiği özelliğe sahip bir işlevsel bağımlılıklar kümesi (X olarak temsil edilir) tanımlamanız gerekir.
- Belirli bir X işlevsel bağımlılıklar kümesi tarafından ima edilen tüm işlevsel bağımlılıklar kümesine X'in kapanışı denir ve X+ olarak yazılır.
- Armstrong'un aksiyomları olarak adlandırılan bir dizi çıkarım kuralı, yeni işlevsel bağımlılıkların verilenlerden nasıl çıkarılabileceğini belirtir.

IŞLEVSEL BAĞIMLILIKLAR IÇIN ÇIKARIM KURALLARI

- A, B ve C, R ilişkisinin niteliklerinin alt kümeleri olsun. Armstrong'un aksiyomları aşağıdaki gibidir:

1. Refleksivite
   - B, A'nın bir alt kümesi ise, A→ B
2. Büyütme
   - Eğer A→ B ise, A,C→ B,C
3. Geçişkenlik
   - Eğer A → B ve B → C ise, A → C

IŞLEVSEL BAĞIMLILIKLAR IÇIN ÇIKARIM KURALLARI

- İlk üç kuraldan, X+'nın hesaplanmasına ilişkin pratik görevi basitleştiren başka kurallar da türetilebilir. D, R bağıntısının niteliklerinin başka bir alt kümesi olsun:

4. Kendi Kaderini Tayin
   - A→ A
5. Ayrıştırma
   - Eğer A→ B,C ise, o zaman A→ B ve A→ C
6. Birlik
   - Eğer A→ B ve A→ C ise, o zaman A→ B,C
7. Kompozisyon
   - Eğer A→ B ve C→ D ise A,C→ B,D

MINIMAL FONKSIYONEL BAĞIMLILIK KÜMELERI

- Bir Y işlevsel bağımlılıklar kümesi, Y'deki her işlevsel bağımlılık X+'da da varsa, bir X işlevsel bağımlılıklar kümesi tarafından kapsanır; Y'deki her bağımlılık 'ten çıkarılabilir.
- Bir işlevsel bağımlılıklar kümesi X, aşağıdaki koşulları sağlıyorsa minimaldir:
  - X'teki her bağımlılığın sağ tarafında tek bir nitelik vardır.
- X'teki herhangi bir A → B bağımlılığını, C'nin A'nın uygun bir alt kümesi olduğu C → B bağımlılığı ile değiştiremeyiz ve yine de X'e eşdeğer bir bağımlılık kümesine sahip olamayız.
- X'ten herhangi bir bağımlılığı kaldıramayız ve yine de X'e eşdeğer bir kümesine sahip oluruz.

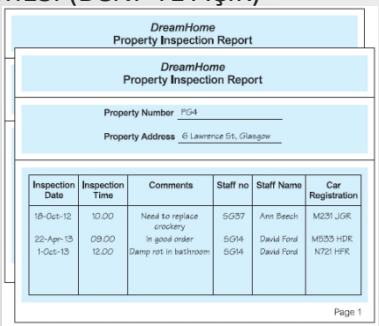
BOYCE-CODD NORMAL FORMU (BCNF)

- Bir ilişkideki tüm aday anahtarları dikkate alan işlevsel bağımlılıklara dayanan BCNF, 3NF'nin genel tanımına kıyasla ek kısıtlamalara da sahiptir.
- Boyce-Codd normal formu (BCNF)
  - Bir bağıntı, ancak ve ancak her belirleyici bir aday anahtar ise BCNF'dedir.
- 3NF ve BCNF arasındaki fark, A -> B işlevsel bağımlılığı için, B birincil anahtar niteliği ise ve A bir aday anahtar değilse, 3NF bu bağımlılığa ilişkide izin verir. Oysa BCNF, bu bağımlılığın bir ilişkide kalması için A'nın bir aday anahtar olması ısrar eder.
  - BCNF'deki her bağıntı aynı zamanda 3NF'dedir. Ancak, 3NF'deki bir bağıntı BCNF'de olmak zorunda değildir.

BOYCE-CODD NORMAL FORMU (BCNF)

- BCNF'nin ihlali oldukça nadirdir.
- BCNF'yi ihlal etme potansiyeli şu ilişkilerde ortaya çıkabilir:
  - iki (veya daha fazla) bileşik aday anahtar içerir;
  - aday anahtarların örtüşmesi, yani en az bir ortak özelliğe sahip olması.

NORMALLEŞMENİN GÖZDEN GEÇİRİLMESİ (BCNF'YE AÇIK)



DreamHome
Property Inspection Report

DreamHome
Property Inspection Report

Property Number PG4

Property Address 6 Lawrence St, Glasgow

Inspection Date	Inspection Time	Comments	Staff no	Staff Name	Car Registration
18-Oct-12	10.00	Need to replace crockery	SG37	Ann Beech	M231 JGR
22-Apr-13	09.00	In good order	SG14	David Ford	M533 HDR
1-Oct-13	12.00	Damp rot in bathroom	SG14	David Ford	N721 HFR

Page 1

NORMALLEŞMENİN GÖZDEN GEÇİRİLMESİ (BCNF'YE AÇIK)

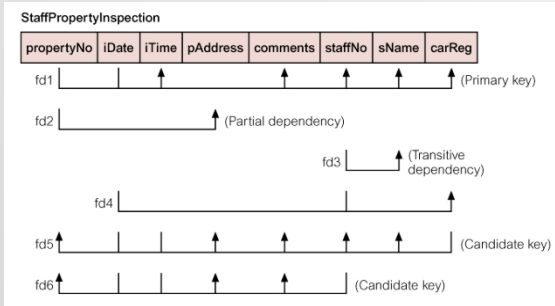
StaffPropertyInspection

propertyNo	pAddress	iDate	iTime	comments	staffNo	sName	carReg
PG4	6 Lawrence St, Glasgow	18-Oct-12 22-Apr-13 1-Oct-13	10.00 09.00 12.00	Need to replace crockery In good order Damp rot in bathroom	SG37 SG14 SG14	Ann Beech David Ford David Ford	M231 JGR M533 HDR N721 HFR
PG16	5 Novar Dr, Glasgow	22-Apr-13 24-Oct-13	13.00 14.00	Replace living room carpet Good condition	SG14 SG37	David Ford Ann Beech	M533 HDR N721 HFR

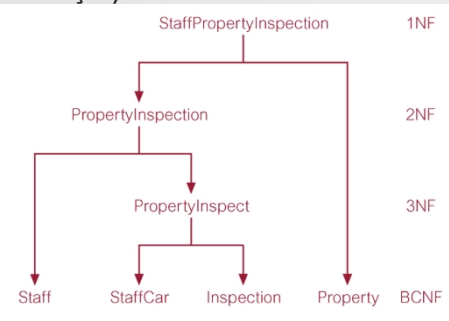
StaffPropertyInspection

propertyNo	iDate	iTime	pAddress	comments	staffNo	sName	carReg
PG4	18-Oct-12	10.00	6 Lawrence St, Glasgow	Need to replace crockery	SG37	Ann Beech	M231 JGR
PG4	22-Apr-13	09.00	6 Lawrence St, Glasgow	In good order	SG14	David Ford	M533 HDR
PG4	1-Oct-13	12.00	6 Lawrence St, Glasgow	Damp rot in bathroom	SG14	David Ford	N721 HFR
PG16	22-Apr-13	13.00	5 Novar Dr, Glasgow	Replace living room carpet	SG14	David Ford	M533 HDR
PG16	24-Oct-13	14.00	5 Novar Dr, Glasgow	Good condition	SG37	Ann Beech	N721 HFR

NORMALLEŞMENİN GÖZDEN GEÇİRİLMESİ (BCNF'YE AÇIK)



NORMALLEŞMENİN GÖZDEN GEÇİRİLMESİ (BCNF'YE AÇIK)



DÖRDÜNCÜ NORMAL FORM (4NF)

- BCNF işlevsel bağımlılıklardan kaynaklanan anormallikleri ortadan kaldırsa da, çok değerli bağımlılık (MVD) adı verilen başka bir bağımlılık türü de veri fazlalığına neden olabilir.
- Bir ilişkide çok değerli bağımlılıkların olası varlığı 1NF'den kaynaklanır ve veri fazlalığına neden olabilir.
- Çok Değerli Bağımlılık (MVD)
  - Bir ilişkideki nitelikler (örneğin, A, B ve C) arasındaki bağımlılık, öyle ki A'nın her değeri için B için bir değerler kümesi ve C için bir değerler kümesi vardır.

DÖRDÜNCÜ NORMAL FORM (4NF)

- Aşağıdaki gösterimi kullanarak bir ilişkideki A, B ve C nitelikleri arasındaki MVD:
  - A–>> B
  - A–>> C
- Çok değerli bir bağımlılık ayrıca önemsiz veya önemsiz olmayan olarak tanımlanabilir.

  R bağıntısında bir A ->> B MVD'si, (a) B, A'nın bir alt kümesi ise veya (b) A U B = R ise önemsiz olarak tanımlanır.

  Bir MVD, ne (a) ne de (b) şartlarının sağlanmaması durumunda önemsiz olarak tanımlanır.

  Önemsiz bir MVD, bir ilişki üzerinde bir kısıtlama , önemsiz bir MVD bir kısıtlama belirtir.
- Boyce-Codd Normal Formunda olan ve önemsiz olmayan çok değerli bağımlılıklar içermbir ilişki olarak tanımlanır.

4NF - ÖRNEK

BranchStaffOwner

branchNo	sName	oName
B003	Ann Beech	Carol Farrel
B003	David Ford	Carol Farrel
B003	Ann Beech	Tina Murphy
B003	David Ford	Tina Murphy

BranchStaff

branchNo	sName
B003	Ann Beech
B003	David Ford

BranchOwner

branchNo	oName
B003	Carol Farrel
B003	Tina Murphy

BEŞINCI NORMAL FORM (5NF)

- İki ilişkiye ayrıştırılan bir ilişkinin kayıpsız birleştirme özelliğine sahip olması gerekir; bu özellik, ilişkiler doğal bir birleştirme işlemiyle yeniden birleştirildiğinde sahte tuple'ların üretilmemesini sağlar.
- Ancak, bir ilişkiyi ikiden fazla ilişkiye ayrıştırmak için gereksinimler vardır. Nadiren de olsa, bu durumlar birleştirme bağımlılığı ve beşinci normal form (5NF) ile yönetilir.
- Birleştirme bağımlılığı olmayan bir ilişki olarak tanımlanır.

5NF - ÖRNEK

(a) **PropertyItemSupplier** (Illegal state)

propertyNo	itemDescription	supplierNo
PG4	Bed	S1
PG4	Chair	S2
PG16	Bed	S2

When this tuple is added to relation.

(b) **PropertyItemSupplier** (Legal state)

propertyNo	itemDescription	supplierNo
PG4	Bed	S1
PG4	Chair	S2
PG16	Bed	S2
PG4	Bed	S2

This new tuple must also be added to exist in any legal state of the relation.

5NF - ÖRNEK

PropertyItem

propertyNo	itemDescription
PG4	Bed
PG4	Chair
PG16	Bed

ItemSupplier

itemDescription	supplierNo
Bed	S1
Chair	S2
Bed	S2

PropertySupplier

propertyNo	supplierNo
PG4	S1
PG4	S2
PG16	S2

CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ KAVRAMSAL VERİTABANI TASARIMI

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- Bir tasarım metodolojisinin amacı.
- Veritabanı tasarımının üç ana aşaması vardır: kavramsal, mantıksal ve fiziksel tasarım.
- Tasarım kapsamının işletmenin belirli görünümlerine nasıl ayrıştırılacağı.
- Bir kuruluşun veri gereksinimlerine dayalı kavramsal bir veri modeli oluşturmak için Varlık-İlişki (ER) modellemesinin nasıl kullanılacağı.
- Veri gereksinimleri kuruluşunun gerçek ve doğru bir temsili olduğundan emin olmak için ortaya çıkan kavramsal modelin nasıl doğrulanacağı.
- Kavramsal veritabanı tasarım süreci nasıl belgelenir?
- Son kullanıcılar, kavramsal veritabanı tasarımı süreci boyunca ayrılmaz bir rol oynar.

TASARIM METODOLOJİSİ

- Tasarım sürecini desteklemek ve kolaylaştırmak için prosedürleri, teknikleri, araçları ve dokümantasyon yardımcılarını kullanan yapılandırılmış bir yaklaşım.

VERITABANI TASARIM METODOLOJISI

- Üç ana aşama
  - Kavramsal veritabanı tasarımı
  - Mantıksal veritabanı tasarımı
  - Fiziksel veritabanı tasarımı

KAVRAMSAL VERİTABANI TASARIMI

- Tüm fiziksel hususlardan bağımsız olarak bir kuruluşta kullanılan verilerin bir modelini oluşturma süreci.

MANTIKSAL VERITABANI TASARIMI

- Belirli bir veri modeline (örneğin ilişkisel) dayalı, ancak belirli bir DBMS'den ve diğer fiziksel hususlardan bağımsız olarak bir kuruluşta kullanılan verilerin bir modelini oluşturma süreci.

FIZIKSEL VERITABANI TASARIMI

- Veritabanının ikincil depolama üzerinde uygulanmasının bir tanımını üretme süreci; verilere verimli erişim sağlamak için kullanılan temel ilişkileri, dosya organizasyonlarını ve dizin tasarımını ve ilgili bütünlük kısıtlamalarını ve güvenlik önlemlerini açıklar.

VERİ TABANI TASARIMINDA KRİTİK BAŞARI FAKTÖRLERİ

- Kullanıcılarla mümkün olduğunca etkileşimli çalışın.
- Veri modelleme süreci boyunca yapılandırılmış bir metodoloji izleyin.
- Veri odaklı bir yaklaşım kullanın.
- Yapısal ve bütünlükle ilgili hususları veri modellerine dahil edin.
- Kavramsallaştırma, normalleştirme ve işlem doğrulama tekniklerini veri modelleme metodolojisinde birleştirin.

VERİ TABANI TASARIMINDA KRİTİK BAŞARI FAKTÖRLERİ

- Veri modellerinin mümkün olduğunca çoğunu temsil etmek için diyagramlar kullanın.
- Ek veri semantiğini temsil etmek için bir Veritabanı Tasarım Dili (DBDL) kullanın.
- Veri modeli diyagramlarını desteklemek için bir veri sözlüğü oluşturun.
- Adımları tekrarlamaya istekli olun.

GENEL BAKIŞ VERİTABANI TASARIM METODOLOJİSİ

Kavramsal veritabanı tasarımı

- Adım 1 Kavramsal veri modeli oluşturun
  - Adım 1.1 Varlık türlerini belirleyin
  - Adım 1.2 İlişki türlerini belirleyin
  - Adım 1.3 Öznitelikleri varlık veya ilişki türleriyle tanımlama ve ilişkilendirme
  - Adım 1.4 Öznitelik alanlarını belirleme
  - Adım 1.5 Aday, birincil ve alternatif anahtar öznitelikleri belirleyin
  - Adım 1.6 Geliştirilmiş modelleme kavramlarının kullanımını değerlendirin (isteğe bağlı adım)
  - Adım 1.7 Modeli yedeklilik açısından kontrol edin
  - Adım 1.8 Kavramsal modeli kullanıcı işlemlerine göre doğrulayın
  - Adım 1.9 Kavramsal veri modelini kullanıcı ile gözden geçirin

GENEL BAKIŞ VERİTABANI TASARIM METODOLOJİSİ

Mantıksal veritabanı tasarımı

- Adım 2 Mantıksal veri modeli oluşturma ve doğrulama
  - Adım 2.1 Mantıksal veri modeli için ilişkilerin türetilmesi
  - Adım 2.2 Normalleştirme kullanarak ilişkileri doğrulayın
  - Adım 2.3 Kullanıcı işlemlerine karşı ilişkileri doğrulama
  - Adım 2.4 Bütünlük kısıtlamalarını tanımlayın
  - Adım 2.5 Mantıksal veri modelini kullanıcı ile gözden geçirin
  - Adım 2.6 Mantıksal veri modellerini global modelle birleştirme (isteğe bağlı adım)
  - Adım 2.7 Gelecekteki büyümeyi kontrol edin

GENEL BAKIŞ VERİTABANI TASARIM METODOLOJİSİ

Fiziksel veritabanı tasarımı

- Adım 3 Hedef DBMS için mantıksal veri modelini çevirme
  - Adım 3.1 Temel ilişkileri tasarlayın
  - Adım 3.2 Türetilmiş verilerin tasarım temsili
  - Adım 3.3 Genel kısıtlamaları tasarlayın
- Adım 4 Dosya organizasyonlarını ve dizinleri tasarlama
  - Adım 4.1 İşlemleri analiz edin
  - Adım 4.2 Dosya organizasyonunu seçin
  - Adım 4.3 Dizinleri seçin
  - Adım 4.4 Disk alanı gereksinimlerini tahmin edin

GENEL BAKIŞ VERİTABANI TASARIM METODOLOJİSİ

- Adım 5 Kullanıcı görünümlerini tasarlayın
- Adım 6 Güvenlik mekanizmaları tasarlayın
- Adım 7 Kontrollü yedeklilik uygulamasını değerlendirin
- Adım 8 Operasyonel sistemi izleyin ve ayarlayın

ADIM 1 KAVRAMSAL VERI OLUŞTURMA

- Kurumun veri gereksinimlerine ilişkin kavramsal bir veri modeli oluşturmak.
  - Model; varlık türleri, ilişki türleri, öznitelikler ve öznitelik alanları, birincil ve alternatif anahtarlar ve bütünlük kısıtlamalarından oluşur.
- Adım 1.1 Varlık türlerini belirleyin
  - Gerekli varlık türlerini tanımlamak için.
- Adım 1.2 İlişki türlerini belirleyin
  - Varlık türleri arasında var olan önemli ilişkileri belirlemek.

ADIM 1 KAVRAMSAL VERI OLUŞTURMA

- Adım 1.3 Öznitelikleri varlık veya ilişki türleriyle tanımlama ve ilişkilendirme
  - Öznitelikleri uygun varlık veya ilişki türleriyle ilişkilendirmek ve her bir özniteliğin ayrıntılarını .
- Adım 1.4 Öznitelik alanlarını belirleme
  - Veri modelindeki öznitelikler için etki alanları belirlemek ve her bir etki alanının ayrıntılarını belgelemek.
- Adım 1.5 Aday, birincil ve alternatif anahtar öznitelikleri belirleyin
  - Her bir varlık için aday anahtar(ları) belirlemek ve birden fazla aday anahtar varsa, birini birincil anahtar ve diğerlerini alternatif anahtar olarak seçmek.
- Adım 1.6 Geliştirilmiş modelleme kavramlarının kullanımını değerlendirin (isteğe bağlı adım)
  - Özelleştirme / genelleştirme, toplama ve kompozisyon gibi gelişmiş modelleme kavramlarının kullanımını değerlendirmek.

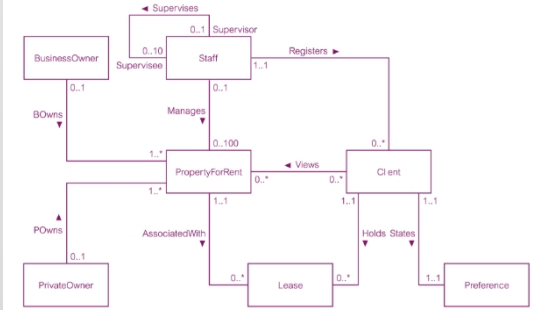
ADIM 1 KAVRAMSAL VERI OLUŞTURMA

- Adım 1.7 Modeli yedeklilik açısından kontrol edin
  - Modelde herhangi bir fazlalık olup olmadığını kontrol etmek ve var olanları kaldırmak.
  - Adım 1.8 Kavramsal modeli kullanıcı işlemlerine göre doğrulayın
    - Kavramsal modelin gerekli işlemleri desteklediğinden emin olmak.
- Adım1.9 Kavramsal veri modelini kullanıcı ile gözden geçirin
  - Modelin, işletmenin veri gereksinimlerinin 'gerçek' bir temsili olduğundan emin olmak için kavramsal veri modelini kullanıcıyla birlikte gözden geçirmek.

DREAMHOME'UN PERSONEL KULLANICI GÖRÜNÜMLERI IÇIN VERI SÖZLÜĞÜNDEN VARLIKLARIN AÇIKLAMASINI GÖSTEREN ALINTI

Entity name	*Description*	*Aliases*	*Occurrence*
Staff	General term descr bing al staff employed by *DreamHome*.	Employee	Each member of staff works at one particular branch.
PropertyForRent	General term descr bing all property for rent.	Property	Each property has a s ngle owner and is available at one specific branch, where the property is managed by one member of staff. A property is viewed by many clients and rented by a single client, at any one time.

DREAMHOME'UN PERSONEL KULLANICI GÖRÜŞLERI İÇIN İLK KESİM ŞEMASI



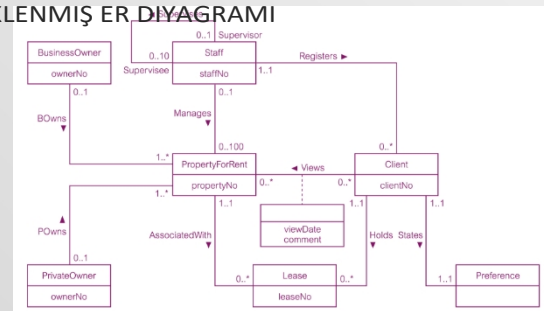
DREAMHOME'UN PERSONEL KULLANICI GÖRÜNÜMLERI IÇIN VERI SÖZLÜĞÜNDEN ILIŞKILERIN AÇIKLAMASINI GÖSTEREN ALINTI

Entity name	*Multiplicity*	*Relationship*	*Multiplicity*	*Entity name*
Staff	0..1 0..1	*Manages* *Supervises*	0..100 0..10	**PropertyForRent** **Staff**
PropertyForRent	1..1	*AssociatedWith*	0..*	**Lease**

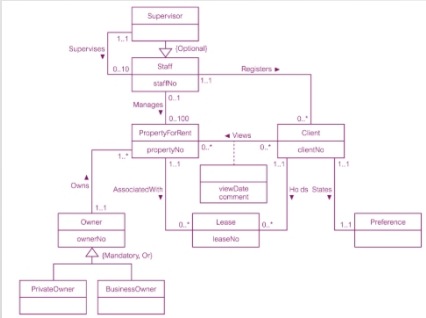
ÖZNITELIKLERIN AÇIKLAMASINI GÖSTEREN DREAMHOME'UN PERSONEL KULLANICI GÖRÜNÜMLERI IÇIN VERI SÖZLÜĞÜNDEN ALINTI

Entity name	*Attributes*	*Description*	*Data Type & Length*	*Nulls*	*Multi-valued*	...
Staff	**staffNo**	Unique y identifies a member of staff	5 variable characters	No	No	
	name					
	fName	First name of staff	15 variable characters	No	No	
	lName	Last name of staff	15 variable characters	No	No	
	position	Job title of member of staff	10 variable characters	No	No	
	sex	Gender of member of staff	1 character (M or F)	Yes	No	
	DOB	Date of birth of member of staff	Date	Yes	No	
PropertyForRent	**propertyNo**	Unique y identifies a property for rent	5 variable characters	No	No	

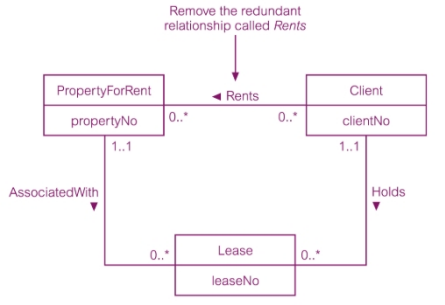
DREAMHOME'UN PERSONEL KULLANICI GÖRÜNÜMLERI IÇIN BIRINCIL ANAHTARLAR EKLENMIŞ ER DIYAGRAMI



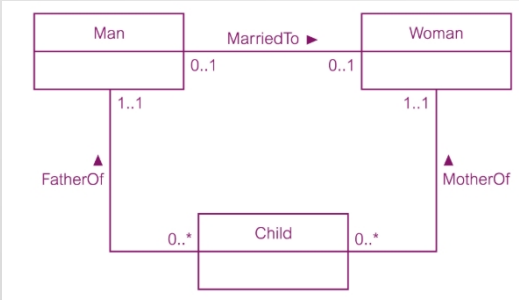
DREAMHOME'UN PERSONEL KULLANICI GÖRÜŞLERİ İÇIN UZMANLAŞMA / GENELLEŞTİRME İLE REVİZE EDİLMİŞ ER DİYAGRAMI



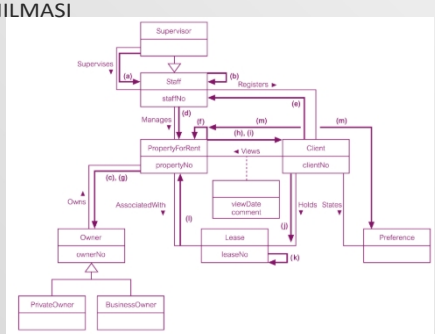
KIRA ADI VERILEN GEREKSIZ BIR ILIŞKININ KALDIRILMASI ÖRNEĞI



YEDEKLI OLMAYAN BIR ILIŞKI BABASININ ÖRNEĞI



KAVRAMSAL MODELIN KULLANICI IŞLEMLERINI DESTEKLEDIĞINI KONTROL ETMEK IÇIN YOLLARIN KULLANILMASI



CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ MANTIKSAL VERİTABANI TASARIMI

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

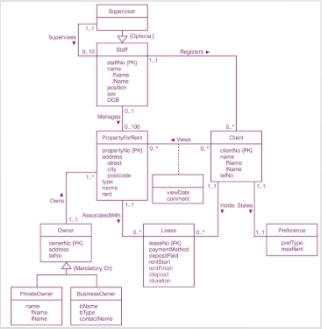
ANA HATLAR

- Kavramsal bir veri modelinden bir dizi ilişkinin nasıl türetileceği.
- Normalleştirme tekniği kullanılarak bu ilişkilerin nasıl doğrulanacağı.
- Gerekli işlemleri desteklediğinden emin olmak için mantıksal bir veri modelinin nasıl doğrulanacağı.
- Bir veya daha fazla kullanıcı görünümüne dayalı yerel mantıksal veri modellerinin tüm kullanıcı görünümlerini temsil eden global bir mantıksal veri modelinde nasıl birleştirileceği.
- Nihai mantıksal veri modelinin, kurumun veri gereksinimlerinin gerçek ve doğru bir temsili olduğundan nasıl emin olunur?

ADIM 2 MANTIKSAL VERI MODELI OLUŞTURMA VE DOĞRULAMA

- Kavramsal veri modelini mantıksal bir veri dönüştürmek ve daha sonra bu modeli normalleştirme kullanarak yapısal olarak doğru olduğunu ve gerekli işlemleri desteklediğini kontrol etmek için doğrulamak.
- Adım 2.1 Mantıksal veri modeli için ilişkilerin türetilmesi
  - Tanımlanan varlıkları, ilişkileri ve öznitelikleri temsil etmek üzere mantıksal veri modeli için ilişkiler oluşturmak.

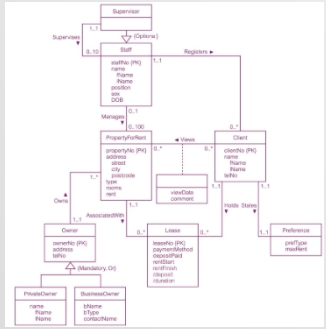
TÜM ÖZNITELIKLERI GÖSTEREN PERSONEL GÖRÜNÜMÜ IÇIN KAVRAMSAL VERI MODELI



ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

1. Güçlü varlık türleri
   - Veri modelindeki her güçlü varlık için, o varlığın tüm basit niteliklerini içeren bir ilişki oluşturun. Bileşik öznitelikler için, yalnızca bileşen basit öznitelikleri dahil edin.
2. Zayıf varlık türleri
   - Veri modelindeki her bir zayıf varlık için, o tüm basit niteliklerini içeren bir ilişki oluşturun. Zayıf bir varlığın birincil anahtarı kısmen veya tamamen her sahip varlıktan türetilir ve bu nedenle, sahip varlıklarla olan tüm ilişkiler eşlenene kadar zayıf bir varlığın birincil anahtarının tanımlanması yapılamaz.
3. Birden çoğa (1:*) ikili ilişki türleri
   - Her 1:* ikili ilişki için, ilişkinin 'bir tarafındaki' varlık üst varlık olarak ve 'çok tarafındaki' varlık alt olarak belirlenir. Bu ilişkiyi temsil etmek için, ana varlığın birincil anahtar öznitelik(ler)inin bir kopyasını yabancı anahtar olarak hareket etmek üzere alt varlığı temsil eden ilişkiye gönderin.

TÜM ÖZNITELIKLERI GÖSTEREN PERSONEL GÖRÜNÜMÜ IÇIN KAVRAMSAL VERI MODELI



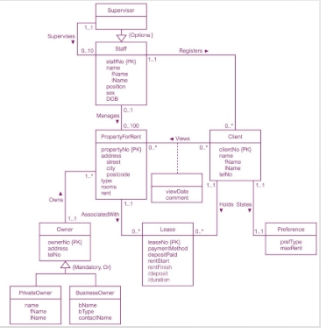
ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

4. Bire bir (1:1) ikili ilişki türleri
   - Bir ilişkideki üst ve alt varlıkları tanımlamak için kardinalite kullanılamayacağından, 1:1 ilişkiyi temsil etmek için ilişkiler oluşturmak daha karmaşıktır. Bunun yerine, katılım kısıtlamaları, ilgili varlıkları tek bir ilişkide birleştirerek veya iki ilişki oluşturup bir ilişkiden diğerine birincil anahtarın bir kopyasını göndererek ilişkiyi temsil en iyisi olup olmadığına karar vermek için kullanılır.
   - Aşağıdakileri göz önünde bulundurun
     - 1:1 ilişkinin her iki tarafında da zorunlu katılım;
     - 1:1 ilişkinin bir tarafında zorunlu katılım;
     - 1:1 ilişkinin her iki tarafında da isteğe bağlı katılım.

ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

- 1:1 ilişkinin her iki tarafında da zorunlu katılım
  - İlgili varlıkları tek bir ilişkide birleştirin ve orijinal varlıkların birincil anahtarlarından birini yeni ilişkinin birincil anahtarı olarak seçin, diğeri ise (varsa) alternatif anahtar olarak kullanılır.
- 1:1 ilişkinin bir tarafında zorunlu katılım
  - Katılım kısıtlamalarını kullanarak üst ve alt varlıkları tanımlayın. İlişkiye katılımı isteğe bağlı olan varlık ana varlık, katılımı zorunlu olan varlık ise alt varlık olarak belirlenir. Ana varlığın birincil anahtarının bir kopyası alt varlığı temsil eden ilişkiye yerleştirilir. İlişkinin bir veya daha fazla özniteliği varsa, bu öznitelikler birincil anahtarın alt gönderilmesini takip etmelidir.
- 1:1 ilişkinin her iki tarafına da isteğe bağlı katılım
  - Bu durumda, şu ya da bu şekilde bir karar verilmesine yardımcı olabilecek ilişki hakkında daha fazla bilgi sürece, ebeveyn ve çocuk varlıkların belirlenmesi keyfidir.

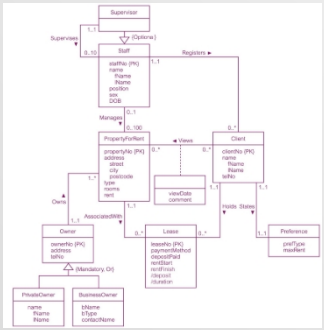
TÜM ÖZNITELIKLERI GÖSTEREN PERSONEL GÖRÜNÜMÜ IÇIN KAVRAMSAL VERI MODELI



ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

5. Bire bir (1:1) özyinelemeli ilişkiler
   - 1:1 özyinelemeli ilişki için, yukarıda 1:1 ilişki için açıklanan katılım kurallarını izleyin.
     - Her iki tarafta da zorunlu katılım, özyinelemeli ilişkiyi birincil anahtarın iki kopyası ile tek bir ilişki olarak temsil eder.
     - Yalnızca bir tarafta zorunlu katılım, birincil anahtarın iki kopyasıyla tek bir ilişki oluşturma veya ilişkiyi temsil etmek için yeni bir ilişki oluşturma seçeneği. Yeni ilişki, her ikisi de birincil anahtarın kopyası yalnızca iki özniteliğe sahip olacaktır. Daha önce olduğu gibi, birincil anahtarların kopyaları yabancı anahtarlar olarak işlev görür ve her birinin ilişkideki amacını belirtmek için yeniden adlandırılması gerekir.
     - her iki tarafta da isteğe bağlı katılım, yine yukarıda açıklandığı gibi yeni bir ilişki oluşturun.

TÜM ÖZNITELIKLERI GÖSTEREN PERSONEL GÖRÜNÜMÜ IÇIN KAVRAMSAL VERI MODELI



ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

6. Üst sınıf/alt sınıf ilişki türleri
   - Üst sınıf varlığını ana varlık olarak ve alt sınıf varlığını alt olarak tanımlayın. Böyle bir ilişkinin bir veya daha fazla ilişki olarak nasıl temsil edileceğine dair çeşitli seçenekler vardır.
   - En uygun seçeneğin seçimi, üst sınıf/alt sınıf ilişkisi üzerindeki ayrıklık ve katılım kısıtlamaları, alt sınıfların farklı ilişkilere dahil olup olmadığı ve üst sınıf/alt sınıf ilişkisindeki katılımcı sayısı gibi bir dizi faktöre bağlıdır.

ÜST SINIF / ALT SINIF İLİŞKİSİNİN TEMSİLİ İÇİN YÖNERGELER

Participation constraint	Disjoint constraint	Relations required
Mandatory	Nondisjoint {And}	Single relation (with one or more discriminators to distinguish the type of each tuple)
Optional	Nondisjoint {And}	Two relations: one relation for superclass and one relation for all subclasses (with one or more discriminators to distinguish the type of each tuple)
Mandatory	Disjoint {Or}	Many relations: one relation for each combined superclass/subclass
Optional	Disjoint {Or}	Many relations: one relation for superclass and one for each subclass

ÜST SINIF / ALT SINIF ILIŞKISININ KATILIM VE AYRIKLIĞA DAYALI TEMSILI

Option 1 – Mandatory, nondisjoint

AllOwner (ownerNo, address, telNo, fName, lName, bName, bType, contactName, pOwnerFlag, bOwnerFlag)
Primary Key ownerNo

Option 2 – Optional, nondisjoint

Owner (ownerNo, address, telNo)
Primary Key ownerNo

OwnerDetails (ownerNo, fName lName, bName, bType, contactName, pOwnerFlag, bOwnerFlag)
Primary Key ownerNo
Foreign Key ownerNo **references** Owner(ownerNo)

Option 3 – Mandatory, disjoint

PrivateOwner (ownerNo, fName, Name, address, telNo)
Primary Key ownerNo

BusinessOwner (ownerNo, bName, bType, contactName, address, telNo)
Primary Key ownerNo

Option 4 – Optional, disjoint

Owner (ownerNo, address, telNo)
Primary Key ownerNo

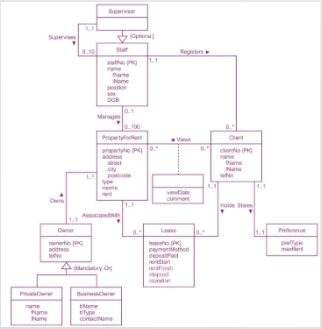
PrivateOwner (ownerNo, fName, Name)
Primary Key ownerNo
Foreign Key ownerNo **references** Owner(ownerNo)

BusinessOwner (ownerNo, bName, bType, contactName)
Primary Key ownerNo
Foreign Key ownerNo **references** Owner(ownerNo)

ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

7. Çoktan çoğa (*:*) ikili ilişki türleri
   - İlişkiyi temsil etmek için bir ilişki oluşturun ve ilişkinin parçası olan tüm öznitelikleri ekleyin. İlişkiye katılan varlıkların birincil anahtar öznitelik(ler)inin bir kopyasını, yabancı olarak hareket etmek üzere yeni ilişkiye göndeririz. Bu yabancı anahtarlar, muhtemelen ilişkinin bazı özniteliklerivle birlikte yeni birincil anahtarını da oluşturacaktır.
8. Karmaşık ilişki türleri
   - İlişkiyi temsil etmek için bir ilişki oluşturun ve ilişkinin parçası olan tüm öznitelikleri ekleyin. Karmaşık ilişkiye katılan varlıkların birincil anahtar nitelik(ler)inin bir kopyasını, yabancı anahtarlar olarak hareket etmek üzere yeni ilişkiye gönderin. Bir 'çoklu' ilişkiyi temsil eden yabancı anahtarlar (örneğin, 1..*, 0..*) genellikle, muhtemelen ilişkinin bazı özniteliklerivle birlikte bu yeni ilişkinin birincil anahtarını da oluşturacaktır.

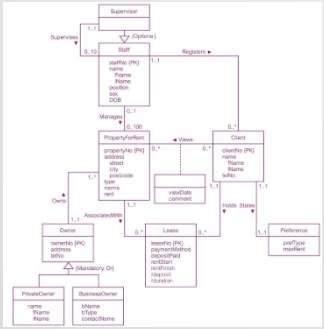
TÜM ÖZNITELIKLERI GÖSTEREN PERSONEL GÖRÜNÜMÜ IÇIN KAVRAMSAL VERI MODELI



ADIM 2.1 MANTIKSAL VERI MODELI IÇIN ILIŞKILERIN TÜRETILMESI

9. Çok değerli nitelikler
   - Çok değerli özniteliği temsil etmek için yeni bir ilişki oluşturun ve yabancı anahtar olarak hareket etmesi için varlığın birincil anahtarını yeni ilişkiye dahil edin. Çok değerli özniteliğin kendisi varlığın alternatif anahtarı olmadığı sürece, yeni ilişkinin birincil anahtarı çok değerli öznitelik ile varlığın birincil anahtarının birleşimidir.

TÜM ÖZNITELIKLERI GÖSTEREN PERSONEL GÖRÜNÜMÜ IÇIN KAVRAMSAL VERI MODELI



VARLIKLARIN VE ILIŞKILERIN ILIŞKILERLE NASIL EŞLEŞTIRILECEĞININ ÖZETI

Entity/Relationship	Mapping
Strong entity	Create relation that includes all simple attributes.
Weak entity	Create relation that includes all simple attributes (primary key still has to be identified after the relationship with each owner entity has been mapped).
1:* binary relationship	Post primary key of entity on 'one' side to act as foreign key in relation representing entity on 'many' side. Any attributes of relationship are also posted to 'many' side.
1:1 binary relationship:	
(a) Mandatory participation on both sides	Combine entities into one relation.
(b) Mandatory participation on one side	Post primary key of entity on 'optional' side to act as foreign key in relation representing entity on 'mandatory' side.
(c) Optional participation on both sides	Arbitrary without further information.
Superclass/subclass relationship	See Table 16.1.
: binary relationship, complex relationship	Create a relation to represent the relationship and include any attributes of the relationship. Post a copy of the primary keys from each of the owner entities into the new relation to act as foreign keys.
Multi-valued attribute	Create a relation to represent the multi-valued attribute and post a copy of the primary key of the owner entity into the new relation to act as a foreign key.

DREAMHOME'UN PERSONEL KULLANICI GÖRÜŞLERI İÇIN İLIŞKILER

Staff (staffNo, fName, lName, position, sex, DOB, supervisorStaffNo) **Primary Key** staffNo **Foreign Key** supervisorStaffNo **references** Staff(staffNo)	**PrivateOwner** (ownerNo, fName, lName, address, telNo) **Primary Key** ownerNo
BusinessOwner (ownerNo, bName, bType, contactName, address, telNo) **Primary Key** ownerNo **Alternate Key** bName **Alternate Key** telNo	**Client** (clientNo, fName, lName, telNo, prefType, maxRent, staffNo) **Primary Key** clientNo **Foreign Key** staffNo **references** Staff(staffNo)
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo) **Primary Key** propertyNo **Foreign Key** ownerNo **references** PrivateOwner(ownerNo) and BusinessOwner(ownerNo) **Foreign Key** staffNo **references** Staff(staffNo)	**Viewing** (clientNo, propertyNo, dateView, comment) **Primary Key** clientNo, propertyNo **Foreign Key** clientNo **references** Client(clientNo) **Foreign Key** propertyNo **references** PropertyForRent(propertyNo)
Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo) **Primary Key** leaseNo **Alternate Key** propertyNo, rentStart **Alternate Key** clientNo, rentStart **Foreign Key** clientNo **references** Client(clientNo) **Foreign Key** propertyNo **references** PropertyForRent(propertyNo) **Derived** deposit (PropertyForRent.rent*2) **Derived** duration (rentFinish – rentStart)	

ADIM 2.2 NORMALLEŞTIRME KULLANARAK ILIŞKILERI DOĞRULAMA

- Normalleştirme kullanarak mantıksal veri modelindeki ilişkileri doğrulamak.

ADIM 2.3 KULLANICI IŞLEMLERINE KARŞI ILIŞKILERI DOĞRULAYIN

- Mantıksal veri modelindeki ilişkilerin gerekli işlemleri desteklediğinden emin .

ADIM 2.4 BÜTÜNLÜK KISITLAMALARINI KONTROL EDIN

- Bütünlük kısıtlamalarının mantıksal veri modelinde temsil edildiğini kontrol etmek. Bu tanımlamayı içerir:
  - Gerekli veriler
  - Öznitelik etki alanı kısıtlamaları
  - Çokluk
  - Varlık bütünlüğü
  - Referans bütünlüğü
  - Genel kısıtlamalar

DREAMHOME'UN PERSONEL KULLANICI GÖRÜNÜMLERINDEKI İLIŞKILER İÇIN REFERANS BÜTÜNLÜĞÜ KISITLAMALARI

Staff (staffNo, fName, lName, position, sex, DOB, supervisorStaffNo)
Primary Key staffNo
Foreign Key supervisorStaffNo **references** Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL

Client (clientNo, fName, lName, telNo, prefType, maxRent, staffNo)
Primary Key clientNo
Foreign Key staffNo **references** Staff(staffNo) ON UPDATE CASCADE ON DELETE NO ACTION

PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo)
Primary Key propertyNo
Foreign Key ownerNo **references** PrivateOwner(ownerNo) and BusinessOwner(ownerNo)
ON UPDATE CASCADE ON DELETE NO ACTION

Foreign Key staffNo **references** Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL

Viewing (clientNo, propertyNo, dateView, comment)
Primary Key clientNo, propertyNo
Foreign Key clientNo **references** Client(clientNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key propertyNo **references** PropertyForRent(propertyNo)
ON UPDATE CASCADE ON DELETE CASCADE

Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo)
Primary Key leaseNo
Alternate Key propertyNo, rentStart
Alternate Key clientNo, rentStart
Foreign Key clientNo **references** Client(clientNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key propertyNo **references** PropertyForRent(propertyNo)
ON UPDATE CASCADE ON DELETE NO ACTION

ADIM 2.5 KULLANICI ILE MANTIKSAL VERI MODELINI GÖZDEN GEÇIRIN

- Mantıksal veri modelini kullanıcılarla birlikte gözden geçirerek, modelin kurumun veri gereksinimlerinin gerçek bir temsili olduğunu düşünmelerini sağlamak.

ADIM 2.6 MANTIKSAL VERI MODELLERINI KÜRESEL MODELLE BIRLEŞTIRME (ISTEĞE BAĞLI ADIM)

- Mantıksal veri modellerini, bir veritabanının tüm kullanıcı görünümlerini temsil eden tek bir global mantıksal veri modelinde birleştirmek için.

ADIM 2.6.1 YEREL MANTIKSAL VERI MODELLERINI KÜRESEL MODELLE BIRLEŞTIRME

- Yerel mantıksal veri modelini tek bir global mantıksal veri modelinde birleştirmek için.
- Bu adımdaki faaliyetler şunları içerir:
  - Adım 2.6.1 Yerel mantıksal veri modellerini global modelle birleştirme
  - Adım 2.6.2 Global mantıksal veri modelini doğrulayın
  - Adım 2.6.3 Global mantıksal veri modelini kullanıcılarla birlikte gözden geçirin.

ADIM 2.6.1 MANTIKSAL VERI MODELLERINI KÜRESEL BIR MODELDE BIRLEŞTIRME

- Görevler tipik olarak şunları içerir:
  1. Varlıkların/ilişkilerin adlarını ve içeriklerini ve aday anahtarlarını gözden geçirin.
  2. İlişkilerin/yabancı anahtarların adlarını ve içeriklerini gözden geçirin.
  3. Yerel veri modellerinden varlıkları/ilişkileri birleştirme
  4. Her yerel veri modeline özgü varlıkları/ilişkileri dahil edin (birleştirmeden)
  5. Yerel veri modellerindeki ilişkileri/yabancı anahtarları birleştirin.
  6. Her bir yerel veri modeline özgü ilişkileri/yabancı anahtarları dahil edin (birleştirmeden).
  7. Eksik varlıklar/ilişkiler ve ilişkiler/yabancı anahtarlar olup olmadığını kontrol edin.
  8. Yabancı anahtarları kontrol edin.
  9. Check Integrity Constraints.
  10. Küresel ER/ilişki diyagramını çizin
  11. Belgeleri güncelleyin.

ADIM 2.6.2 GLOBAL MANTIKSAL VERI MODELINI DOĞRULAYIN

- Global mantıksal veri modelinden oluşturulan ilişkileri normalizasyon tekniğini kullanarak doğrulamak ve gerekirse gerekli işlemleri desteklemelerini sağlamak.

ADIM 2.6.3 GLOBAL MANTIKSAL VERI MODELINI KULLANICILARLA GÖZDEN GEÇIRIN

- Global mantıksal veri modelini kullanıcılarla birlikte gözden , modelin bir kuruluşun veri gereksinimlerinin gerçek bir temsili olduğunu düşünmelerini sağlamak.

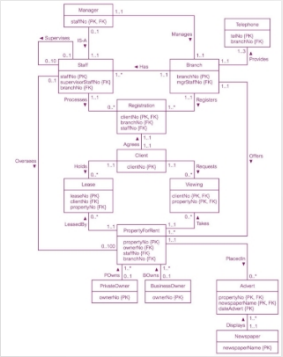
DREAMHOME'UN ŞUBE KULLANICI GÖRÜŞLERI IÇIN ILIŞKILER

Branch (branchNo, street, c ty, postcode, mgrStaffNo) **Primary Key** branchNo **Alternate Key** postcode **Foreign Key** mgrStaffNo **references** Manager(staffNo)	**Telephone** (telNo, branchNo) **Primary Key** telNo **Foreign Key** branchNo **references** Branch(branchNo)
Staff (staffNo, name, position, salary, supervisorStaffNo, branchNo) **Primary Key** staffNo **Foreign Key** supervisorStaffNo **references** Staff(staffNo) **Foreign Key** branchNo **references** Branch(branchNo)	**Manager** (staffNo, mgrStartDate, bonus) **Primary Key** staffNo **Foreign Key** staffNo **references** Staff(staffNo)
PrivateOwner (ownerNo, name, address, telNo) **Primary Key** ownerNo	**BusinessOwner** (bName, bType, contactName, address, telNo) **Primary Key** bName **Alternate Key** telNo
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo, bName, branchNo) **Primary Key** propertyNo **Foreign Key** ownerNo **references** PrivateOwner(ownerNo) **Foreign Key** bName **references** BusinessOwner(bName) **Foreign Key** staffNo **references** Staff(staffNo) **Foreign Key** branchNo **references** Branch(branchNo)	**Client** (clientNo, name, telNo, prefType, maxRent) **Primary Key** clientNo
Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo) **Primary Key** leaseNo **Alternate Key** propertyNo, rentStart **Alternate Key** clientNo, rentStart **Foreign Key** clientNo **references** Client(clientNo) **Foreign Key** propertyNo **references** PropertyForRent(propertyNo) **Derived** deposit (PropertyForRent.rent*2) **Derived** duration (rentFinish – rentStart)	**Registration** (clientNo, branchNo, staffNo, dateJoined) **Primary Key** clientNo **Foreign Key** cl ent**No references** C ient(clientNo) **Foreign Key** branchNo **references** Branch(branchNo) **Foreign Key** staffNo **references** Staff(staffNo)
Advert (propertyNo, newspaperName, dateAdvert, cost) **Primary Key** propertyNo, newspaperName, dateAdvert **Foreign Key** propertyNo **references** PropertyForRent(propertyNo) **Foreign Key** newspaperName **references** Newspaper(newspaperName)	**Newspaper** (newspaperName, address, telNo, contactName) **Primary Key** newspaperName **Alternate Key** telNo

DREAMHOME IÇIN KÜRESEL MANTIKSAL VERI MODELINI TEMSIL EDEN ILIŞKILER

Branch (branchNo, street, city, postcode, mgrStaffNo) **Primary Key** branchNo **Alternate Key** postcode **Foreign Key** mgrStaffNo **references** Manager(staffNo)	**Telephone** (telNo, branchNo) **Primary Key** telNo **Foreign Key** branchNo **references** Branch(branchNo)
Staff (staffNo, fname, lname, position, sex, DOB, salary, supervisorStaffNo, branchNo) **Primary Key** staffNo **Foreign Key** supervisorStaffNo **references** Staff(staffNo) **Foreign Key** branchNo **references** Branch(branchNo)	**Manager** (staffNo, mgrStartDate, bonus) **Primary Key** staffNo **Foreign Key** staffNo **references** Staff(staffNo)
PrivateOwner (ownerNo, fname, lname, address, telNo) **Primary Key** ownerNo	**BusinessOwner** (ownerNo, bName, bType, contactName, address, telNo) **Primary Key** ownerNo **Alternate Key** bName **Alternate Key** telNo
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo, branchNo) **Primary Key** propertyNo **Foreign Key** ownerNo **references** PrivateOwner(ownerNo) and BusinessOwner(ownerNo) **Foreign Key** staffNo **references** Staff(staffNo) **Foreign Key** branchNo **references** Branch(branchNo)	**Viewing** (c ientNo, propertyNo, dateView, comment) **Primary Key** clientNo, propertyNo **Foreign Key** clientNo **references** Client(clientNo) **Foreign Key** propertyNo **references** PropertyForRent(propertyNo)
Client (clientNo, fname, lname, telNo, prefType, maxRent) **Primary Key** clientNo	**Registration** (clientNo, branchNo, staffNo, dateJoined) **Primary Key** clientNo **Foreign Key** clientNo **references** Client(clientNo) **Foreign Key** branchNo **references** Branch(branchNo) **Foreign Key** staffNo **references** Staff(staffNo)
Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, c ientNo, propertyNo) **Primary Key** leaseNo **Alternate Key** propertyNo, rentStart **Alternate Key** c ientNo, rentStart **Foreign Key** clientNo **references** Client(clientNo) **Foreign Key** propertyNo **references** PropertyForRent(propertyNo) **Derived** deposit (PropertyForRent.rent*2) **Derived** duration (rentFinish – rentStart)	**Newspaper** (newspaperName, address, telNo, contactName) **Primary Key** newspaperName **Alternate Key** telNo
Advert (propertyNo, newspaperName, dateAdvert, cost) **Primary Key** propertyNo, newspaperName, dateAdvert **Foreign Key** propertyNo **references** PropertyForRent(propertyNo) **Foreign Key** newspaperName **references** Newspaper(newspaperName)	

DREAMHOME IÇIN KÜRESEL ILIŞKI DIYAGRAMI



CSE 204 - VERİTABANI SİSTEMLERİNE GİRİŞ FİZİKSEL VERİTABANI TASARIMI

Joseph LEDET
Akdeniz Bilgisayar Mühendisliği Bölümü
josephledet@akdeniz.edu.tr

ANA HATLAR

- Fiziksel veritabanı tasarımının amacı.
- Mantıksal veritabanı tasarımının fiziksel veritabanı tasarımıyla nasıl eşleştirileceği.
- Hedef VTYS için temel ilişkiler nasıl tasarlanır.
- Hedef VTYS için genel kısıtlamalar nasıl tasarlanır.
- İşlemlerin analizine dayalı olarak uygun dosya organizasyonlarının nasıl seçileceği.
- Performansı artırmak için ikincil dizinler ne zaman kullanılmalıdır?
- Veritabanının boyutu nasıl tahmin edilir?
- Kullanıcı görünümleri nasıl tasarlanır?
- Kullanıcı gereksinimlerini karşılamak için güvenlik mekanizmalarının nasıl tasarlanacağı.

MANTIKSAL VE FIZIKSEL VERITABANI TASARIMI

- Fiziksel tasarım süreci için bilgi kaynakları, mantıksal veri modelini ve modeli tanımlayan belgeleri içerir.
- Mantıksal veritabanı tasarımı ne ile ilgilidir, fiziksel tasarımı ise nasıl ile ilgilidir.

FIZIKSEL VERITABANI TASARIMI

- Veritabanının ikincil depolama üzerinde uygulanmasına ilişkin bir açıklama üretme süreci.
- Verilere etkin erişim sağlamak için kullanılan temel ilişkileri, dosya organizasyonlarını ve dizinleri ve bunlarla ilişkili bütünlük kısıtlamalarını ve güvenlik önlemlerini açıklar.

FİZİKSEL VERİ TABANI TASARIM METODOLOJİSİNE GENEL BAKIŞ

- Adım 3 Hedef DBMS için mantıksal veri modelini çevirme
  - Adım 3.1 Temel ilişkileri tasarlayın
  - Adım 3.2 Türetilmiş verilerin tasarım temsili
  - Adım 3.3 Genel kısıtlamaları tasarlayın
- Adım 4 Dosya organizasyonlarını ve dizinleri tasarlama
  - Adım 4.1 İşlemleri analiz edin
  - Adım 4.2 Dosya organizasyonlarını seçin
  - Adım 4.3 Dizinleri seçin
  - Adım 4.4 Disk alanı gereksinimlerini tahmin edin

FİZİKSEL VERİ TABANI TASARIM METODOLOJİSİNE GENEL BAKIŞ

- Adım 5 Kullanıcı görünümlerini tasarlayın
- Adım 6 Güvenlik mekanizmaları tasarlayın
- Adım 7 Kontrollü yedeklilik uygulamasını değerlendirin
- Adım 8 Operasyonel sistemi izleyin ve ayarlayın

ADIM 3 HEDEF DBMS IÇIN MANTIKSAL VERI MODELINI ÇEVIRME

Hedef VTYS'de uygulanabilecek mantıksal veri modelinden ilişkisel bir veritabanı şeması üretmek.

- Temel ilişkilerin nasıl oluşturulacağı ve sistemin tanımını destekleyip desteklemediği gibi hedef DBMS' işlevselliğini bilmek gerekir:
  - PK'ler, FK'ler ve AK'ler;
  - gerekli veriler - yani sistemin NOT NULL'u destekleyip desteklemediği;
  - etki alanları;
  - ilişkisel bütünlük kısıtlamaları;
  - genel kısıtlamalar.

ADIM 3.1 TASARIM TEMEL İLİŞKİLERİ

Mantıksal modelde tanımlanan temel ilişkilerin hedef VTYS'de nasıl temsil edileceğine karar vermek.

- Her bir ilişki için tanımlama yapmanız gerekir:
  - ilişkinin adı;
  - parantez içinde basit niteliklerin bir listesi;
  - PK ve uygun olan yerlerde AK'ler ve FK'ler.
  - Tanımlanan tüm FK'lar için referans bütünlüğü kısıtlamaları.

ADIM 3.1 TASARIM TEMEL İLİŞKİLERİ

- Veri sözlüğünden, her bir öznitelik için
  - bir veri türü, uzunluk ve etki alanı üzerindeki herhangi bir kısıtlamadan oluşan etki alanı;
  - öznitelik için isteğe bağlı bir varsayılan değer;
  - null'ları tutup tutamayacağı;
  - türetilip türetilmediği ve türetilmişse nasıl hesaplanması gerektiği.

PROPERTYFORRENT ILIŞKISI IÇIN DBDL

```
Domain PropertyNumber:      variable length character string, length 5
Domain Street:              variable length character string, length 25
Domain City:                variable length character string, length 15
Domain Postcode:            variable length character string, length 8
Domain PropertyType:        single character, must be one of 'B', 'C', 'D', 'E', 'F', 'H', 'M', 'S'
Domain PropertyRooms:       integer, in the range 1–15
Domain PropertyRent:        monetary value, in the range 0.00–9999.99
Domain OwnerNumber:         variable length character string, length 5
Domain StaffNumber:         variable length character string, length 5
Domain BranchNumber:        fixed length character string, length 4
PropertyForRent(
    propertyNo    PropertyNumber    NOT NULL,
    street        Street            NOT NULL,
    city          City              NOT NULL,
    postcode      Postcode,
    type          PropertyType      NOT NULL DEFAULT 'F',
    rooms         PropertyRooms     NOT NULL DEFAULT 4,
    rent          PropertyRent      NOT NULL DEFAULT 600,
    ownerNo       OwnerNumber       NOT NULL,
    staffNo       StaffNumber,
    branchNo      BranchNumber      NOT NULL,
    PRIMARY KEY (propertyNo),
    FOREIGN KEY (staffNo) REFERENCES Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL,
    FOREIGN KEY (ownerNo) REFERENCES PrivateOwner(ownerNo) and BusinessOwner(ownerNo)
                                ON UPDATE CASCADE ON DELETE NO ACTION,
    FOREIGN KEY (branchNo) REFERENCES Branch(branchNo)
                                ON UPDATE CASCADE ON DELETE NO ACTION);
```

ADIM 3.2 TÜRETILMIŞ VERILERIN TASARIM TEMSILI

Mantıksal veri modelinde bulunan türetilmiş verilerin hedef VTYS'de nasıl temsil edileceğine karar vermek.

- Mantıksal veri modelini ve veri sözlüğünü inceleyin ve türetilmiş tüm öznitelikleri listesini .
- Türetilen öznitelik veritabanında saklanabilir veya her ihtiyaç duyulduğunda hesaplanabilir.
- Seçilen seçenek alınır:
  - Türetilen verileri depolamak ve türetildiği operasyonel verilerle tutarlı tutmak için ek maliyet;
  - her gerekli olduğunda hesaplamak için maliyet.
- Performans kısıtlamalarına tabi olarak daha ucuz seçenek seçilir.

PROPERTYFORRENT ILIŞKISI VE TÜRETILMIŞ NOOFPROPERTIES ÖZNITELIĞINE SAHIP PERSONEL ILIŞKISI

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Staff

staffNo	fName	lName	branchNo	noOfProperties
SL21	John	White	B005	0
SG37	Ann	Beech	B003	2
SG14	David	Ford	B003	1
SA9	Mary	Howe	B007	1
SG5	Susan	Brand	B003	0
SL41	Julie	Lee	B005	1

ADIM 3.3 TASARIM GENEL KISITLAMALARI

Hedef VTYS için genel kısıtlamaları tasarlamak.

- Bazı DBMS'ler kurumsal kısıtlamaları tanımlamak için diğerlerinden daha fazla olanak sağlar. Örnek:

```
CONSTRAINT StaffNotHandlingTooMuch CHECK
(NOT EXISTS (SELECT staffNo
FROM PropertyForRent GROUP
BY staffNo
COUNT(*)> 100))
```


4. ADIM DOSYA ORGANİZASYONLARI VE DİZİNLERİ TASARLAMAK

Kabul edilebilir performans elde etmek için gerekli olan temel ilişkileri ve dizinleri depolamak için en uygun dosya organizasyonlarını belirlemek; yani, ilişkilerin ve tuple'ların ikincil depolamada tutulma şekli.

- Veritabanının desteklemesi gereken tipik iş yükünü anlamalıdır.

ADIM 4.1 İŞLEMLERİ ANALİZ EDİN

Veritabanı üzerinde çalışacak işlemlerin işlevselliğini anlamak ve önemli analiz etmek.

- Aşağıdakiler gibi performans kriterlerini belirlemeye çalışın:
  - Sık çalışan ve performans üzerinde önemli bir etkiye sahip olacak işlemler;
  - iş için kritik olan işlemler;
  - Gün/hafta içinde veritabanına talebin yüksek olacağı zamanlar (pik yük olarak adlandırılır).
- Veritabanının performans sorunlarına neden olabilecek kısımlarını belirlemek için bu bilgileri kullanın.
- Ayrıca işlemlerin üst düzey işlevselliğini de bilmeniz gerekir,
  - güncellenen öznitelikler;
  - Bir sorguda kullanılan arama kriterleri.

ADIM 4.1 İŞLEMLERİ ANALİZ EDİN

- Genellikle tüm işlemleri analiz etmek mümkün değildir, bu nedenle en 'önemli' olanları araştırın.
- Bunları tanımlamaya yardımcı olmak için kullanabilirsiniz:
  - her bir eriştiği ilişkileri gösteren işlem/ilişki çapraz referans matrisi ve/veya
  - işlem kullanım haritası, hangi ilişkilerin potansiyel olarak yoğun bir şekilde kullanıldığını gösterir.
- Sorunlu olabilecek alanlara odaklanmak:
  1. Tüm işlem yollarını ilişkilerle eşleyin.
  2. İşlemler tarafından en sık hangi ilişkilere erişildiğini belirleyin.
  3. Bu ilişkileri içeren seçilmiş işlemlerin veri kullanımını analiz edin.

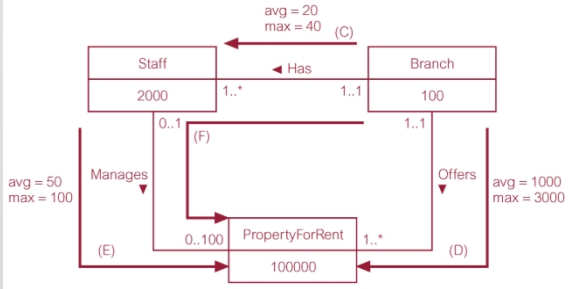
IŞLEMLERIN VE ILIŞKILERIN ÇAPRAZ REFERANSLANMASI

Table 17.1 Cross-referencing transactions and relations.

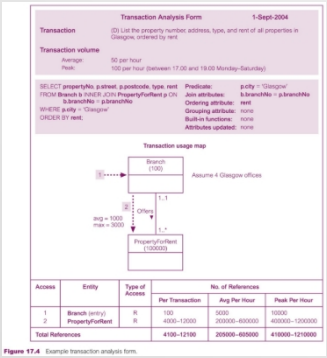
Transaction/ Relation	(A)				(B)				(C)				(D)				(E)				(F)			
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D
Branch										X				X								X		
Telephone																								
Staff		X				X				X								X				X		
Manager																								
PrivateOwner	X																							
BusinessOwner	X																							
PropertyForRent	X					X	X	X						X				X				X		
Viewing																								
Client																								
Registration																								
Lease																								
Newspaper																								
Advert																								

I = Insert; R = Read; U = Update; D = Delete

ÖRNEK IŞLEM KULLANIM HARITASI



ÖRNEK İŞLEM ANALİZ FORMU



Transaction Analysis Form 1-Sept-2004

Transaction (D) List the property number, address, type, and rent of all properties in Glasgow, ordered by rent

Transaction volume
Average: 50 per hour
Peak: 100 per hour (between 17.00 and 19.00 Monday–Saturday)

```
SELECT propertyNo, p.street, p.postcode, type, rent
FROM Branch b INNER JOIN PropertyForRent p ON
b.branchNo = p.branchNo
WHERE p.city = 'Glasgow'
ORDER BY rent;
```

Predicate: p.city = 'Glasgow'
Join attributes: b.branchNo = p.branchNo
Ordering attribute: rent
Grouping attribute: none
Built-in functions: none
Attributes updated: none

Transaction usage map

Access	Entity	Type of Access	No. of References		
			Per Transaction	Avg Per Hour	Peak Per Hour
1	Branch (entry)	R	100	5000	10000
2	PropertyForRent	R	4000–12000	200000–600000	400000–1200000
Total References			4100–12100	205000–605000	410000–1210000

Figure 17.4 Example transaction analysis form.

ADIM 4.2 DOSYA ORGANIZASYONLARINI SEÇIN

Her bir temel ilişki için verimli bir dosya organizasyonu belirlemek.

- Dosya organizasyonları arasında Yığın, Karma, Dizinlenmiş Sıralı Erişim Yöntemi (ISAM), B+-Ağacı ve Kümeler bulunur.
- Bazı DBMS'ler dosya organizasyonlarının seçilmesine izin vermeyebilir.

ADIM 4.3 DIZINLERI SEÇIN

Dizin eklemenin sistemin artırıp artırmayacağını belirlemek için.

- Bir yaklaşım, tuple'ları sırasız tutmak ve çok sayıda gerektiğinde **ikincil indeksler**.
- Diğer bir yaklaşım ise bir birincil veya kümeleme indeksi belirterek ilişkideki çiftleri sıralamaktır.
- Bu durumda, tuple'ları sıralamak veya kümelemek için özniteliği şu şekilde seçin:
  - birleştirme işlemleri için en sık kullanılan öznitelik - bu, birleştirme işlemini daha verimli hale getirir veya
  - özniteliği, bir ilişkideki tuple'lara bu öznitelik sırasına göre erişmek için en sık kullanılan özniteliktir.

ADIM 4.3 DIZINLERI SEÇIN

- Seçilen sıralama özniteliği ilişkinin anahtarı ise, dizin bir birincil dizin olacaktır; aksi takdirde, dizin bir kümeleme dizini olacaktır.
- Her ilişki yalnızca bir birincil dizine veya bir kümeleme dizinine sahip olabilir.
- İkincil dizinler, verileri daha verimli bir şekilde almak için kullanılabilecek bir temel ilişki için ek bir anahtar belirtmek için bir mekanizma sağlar.

ADIM 4.3 DIZINLERI SEÇIN

- İkincil dizinlerin bakımı ve kullanımıyla ilgili ek yükü, verileri alırken elde edilen performans artışına karşı dengelemelidir.
- Buna şunlar dahildir:
  - her tuple eklendiğinde her ikincil dizine bir dizin kaydı ekler;
  - İlgili tuple ikincil indeksi güncellemek;
  - ikincil dizini depolamak için gereken disk alanında artış;
  - tüm ikincil dizinleri dikkate almak için sorgu optimizasyonu sırasında olası performans düşüşü.

ADIM 4.3 DİZİN SEÇİMİ - 'İSTEK LİSTESİ' SEÇİMİ İÇİN KILAVUZ

1. Küçük ilişkileri indekslemeyin.
2. Dosya organizasyonunun anahtarı değilse bir ilişkinin PK dizini.
3. Sık erişiliyorsa bir FK'ya ikincil dizin ekleyin.
4. Ağırlıklı olarak ikincil anahtar olarak kullanılan herhangi bir özniteliğe ikincil dizin ekleyin.
5. Aşağıdakilerle ilgili özniteliklere ikincil dizin ekleyin: seçim veya ölçütleri; ORDER BY; GROUP BY; ve sıralama içeren diğer işlemler (UNION veya DISTINCT gibi).

ADIM 4.3 DİZİN SEÇİMİ - 'İSTEK LİSTESİ' SEÇİMİ İÇİN KILAVUZ

6. Yerleşik işlevlerde yer alan özniteliklere ikincil dizin ekleyin.
7. Yalnızca dizin planıyla sonuçlanabilecek özniteliklere ikincil dizin ekleyin.
8. Sık güncellenen bir özniteliği veya ilişkiyi indekslemekten kaçının.
9. Sorgu ilişkinin önemli bir bölümünü getirecekse bir özniteliği indekslemekten kaçının.
10. Uzun karakter dizelerinden oluşan öznitelikleri indekslemekten kaçının.

ADIM 4.4 DISK ALANI GEREKSINIMLERINI TAHMIN ETME

- Veritabanının ihtiyaç duyacağı disk alanı miktarını tahmin etmek için.

ADIM 5 KULLANICI GÖRÜNÜMLERINI TASARLAYIN

- Veritabanı sistemi geliştirme yaşam döngüsünün Gereksinim Toplama ve Analiz aşamasında belirlenen kullanıcı görünümlerini tasarlamak.

ADIM 6 GÜVENLIK ÖNLEMLERI TASARLAYIN

- Veritabanı için güvenlik önlemlerini kullanıcılar tarafından belirtildiği şekilde tasarlamak.