ISE 305-Veri Tabanı Yönetim Sistemleri

Hafta 2 - Veri Modelleri

Arş. Gör. Dr. Muhammed Kotan

Temel Kavramlar

Veri Modelleme

- Veri tabanı tasarım yolculuğunun ilk adımıdır ve gerçek dünyadaki nesneler ile bilgisayar veri tabanı arasında bir köprü görevi görür.
- Belirli bir problem alanı için belirli bir veri modeli oluşturma süreci.

Model

- Karmaşık gerçek dünya nesnelerinin veya olayların bir soyutlamasıdır.

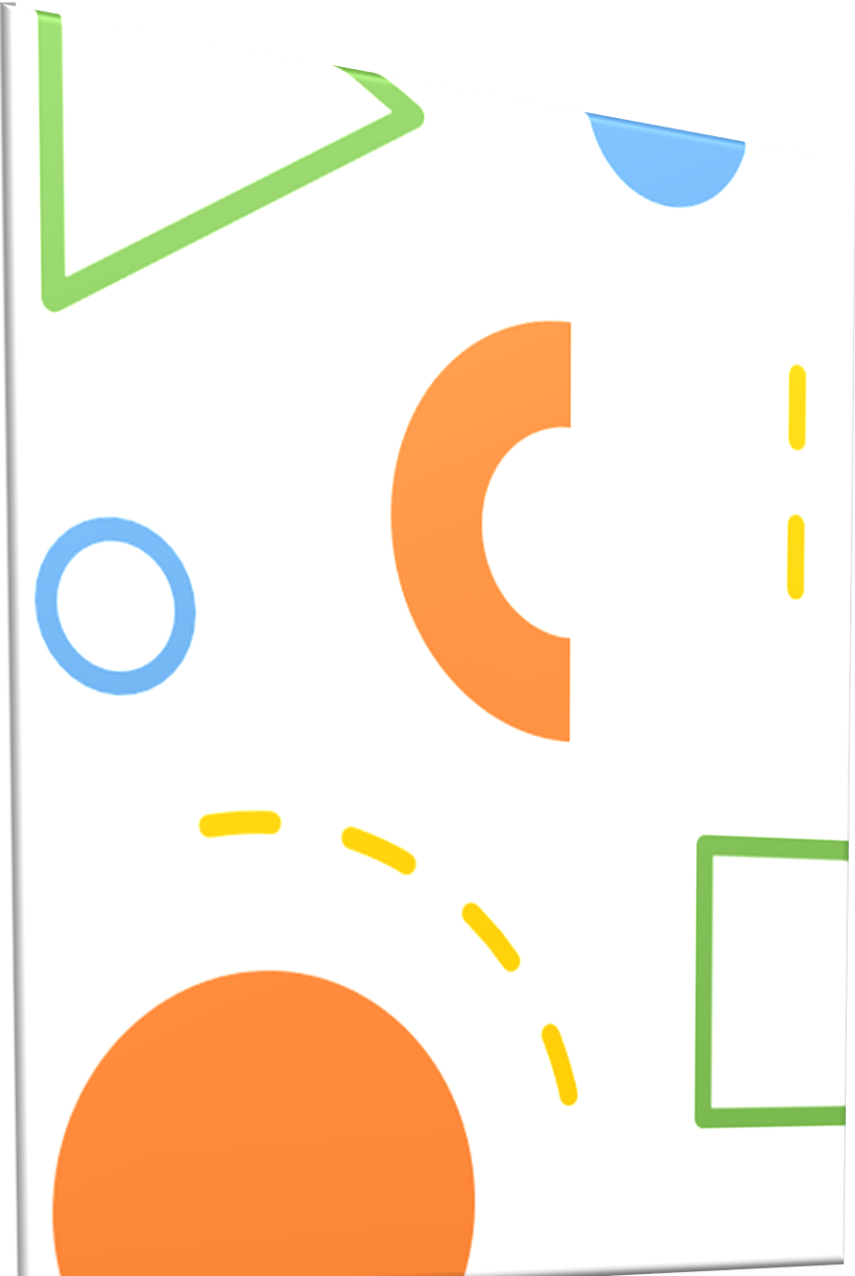
Veri Modeli

- Karmaşık bir "gerçek dünya" veri yapısının genellikle grafiksel bir temsili.
- Veri modelleri, Veritabanı Yaşam Döngüsünün veritabanı tasarımı aşamasında kullanılır.

***Veritabanı ortamında veri modeli,* belirli bir problem alanına destek olmak amacıyla veri yapılarını ve özelliklerini, ilişkileri, kısıtlamaları, dönüşümleri ve diğer yapıları temsil eder.**

Veri Modelleri?

- Karmaşık gerçek dünya veri yapılarının basit olarak gösterilmesi (genellikle grafiksel) için kullanılan araçlara *veri modeli* ismi verilir.
- Tasarımcı, uygulama programcısı ve son kullanıcı arasındaki etkileşimi kolaylaştırır.
- Veri modelleri sayesinde veritabanı tasarımı daha kolay hale gelmektedir.
- Son kullanıcıların veriler üzerinde farklı görüşleri ve ihtiyaçları vardır. Veri modeli, çeşitli kullanıcılar için verileri düzenler.

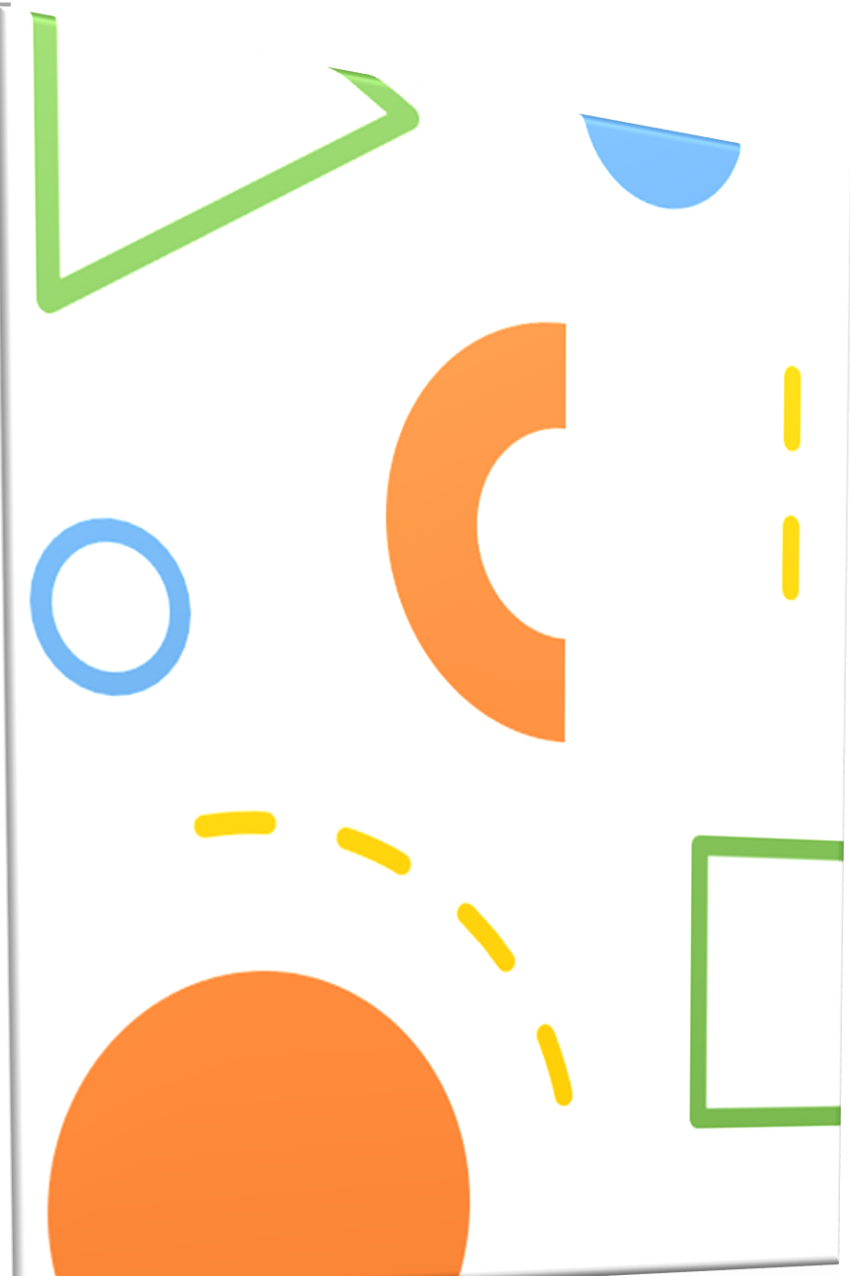


Veri Modelleri?

- Veri modelleme yinelemeli ve aşamalı bir süreçtir.
- Önce basit model oluşturulur. Daha sonra ayrıntılar eklenir. En sonunda veri tabanı tasarımında kullanılan şablon (blueprint) elde edilir.

«*Problem alanının basit bir şekilde anlaşılmasıyla başlanır ve anlayış arttıkça, veri modelinin ayrıntı düzeyi de artar. Doğru bir şekilde yapıldığında, nihai veri modeli, tüm son kullanıcı gereksinimlerini karşılayacak bir veritabanı oluşturmak için tüm talimatları içeren bir şablon elde edilir. Bu şablon, anlatımsal ve grafikseldir, yani hem net bir dilde metin açıklamaları hem de ana veri öğelerini tasvir eden açık, kullanışlı diyagramlar içerir.*»

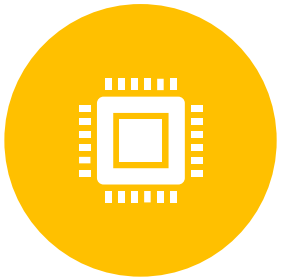
- Uygulamaya hazır bir veri modeli en azından aşağıdaki bileşenleri içermelidir:
  - Son kullanıcı verilerini depolayacak veri yapısının açıklamaları
  - Verilerin bütünlüğünü garanti altına almak için bir dizi uygulanabilir kurallar
  - Gerçek dünyada veri dönüşümlerini desteklemek için bir veri işleme metodolojisi



Veri Modellerinin Önemi

Veri modelleri; tasarımcı, uygulama programcısı ve son kullanıcı arasındaki etkileşimi kolaylaştırabilir.

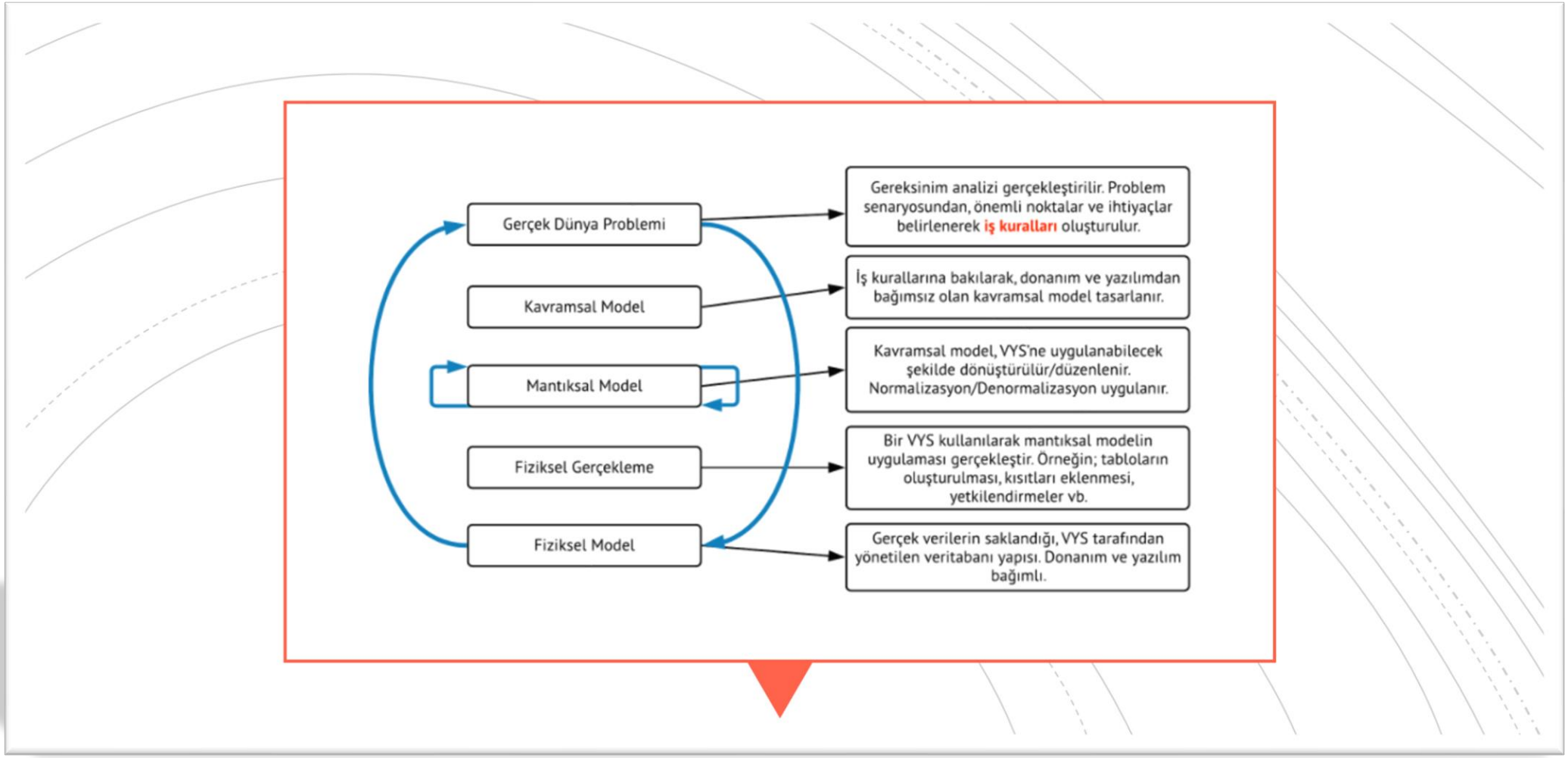
İyi geliştirilmiş bir veri modeli, veri tabanı tasarımının kendisi için geliştirildiği organizasyonun daha iyi anlaşılmasını sağlayabilir.



İyi bir veri tabanı planı mevcut olduğunda, bir uygulama programcısının verilere ilişkin görüşünün yöneticinin veya son kullanıcının görüşünden farklı olması önemli değildir.

Veri modeli bir soyutlamadır, gerekli veriler veri modelinden elde edilemez. Planı olmaksızın iyi bir ev inşa edilemeyeceği gibi önce uygun bir veri modeli oluşturulmadan da iyi bir veri tabanı oluşturma olasılığı oldukça düşüktür..

Veri Tabanı Geliştirme Yaşam Döngüsü



Veri Modeli Temel Bileşenler

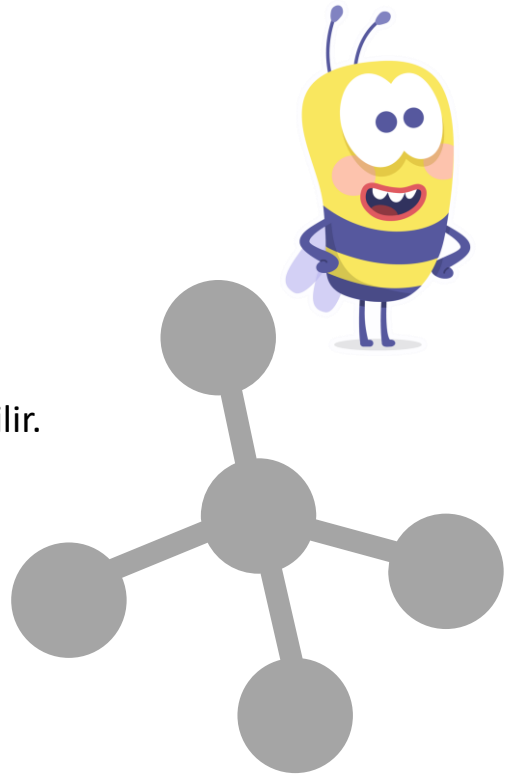
- **Tüm veri modellerinin** temel yapı taşları; *varlıklar, nitelikler, ilişkiler (bağıntı) ve kısıtlamalardır.*

- **Varlık (Entity)** - Hakkında veri toplanan ve saklanan her şey (öğrenci, ders, personel vb.).
  - Gerçek dünyadaki belirli bir nesne türünü temsil eder.
  - Bir varlık "ayırt edilebilir" - yani, her varlık oluşumu benzersiz ve farklıdır. Örneğin, bir MÜŞTERİ varlığı
  - Varlıklar, müşteriler veya ürünler gibi fiziksel nesneler olabildiği gibi uçuş rotaları gibi soyut nesneler de olabilir.

- **Varlık kümesi (Entity set)** - Aynı türden varlıkların oluşturduğu kümeye denir (Öğrenciler, Dersler vb.).

- **Nitelik (Attribute)** - Varlığın sahip olduğu özellikler.

  - Örneğin, bir ÖĞRENCİ varlığı, öğrenci okul numarası, adı, soyadı, telefon numarası, adresi vb. niteliklerle tanımlanabilir.



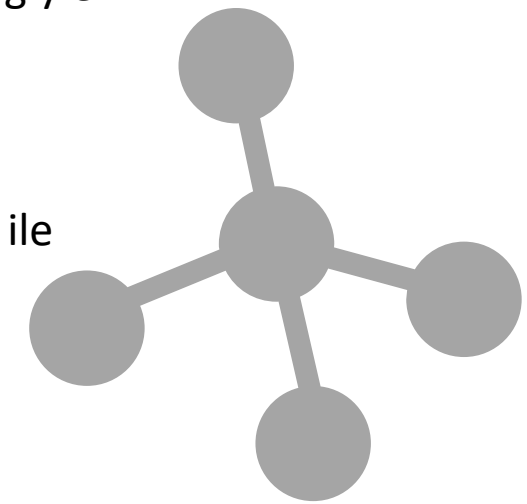
Veri Modeli Temel Bileşenler

- **Tüm veri modellerinin** temel yapı taşları; varlıklar, nitelikler, ilişkiler ve kısıtlamalardır.

- **Bağıntı (Relationship)** - Varlıklar arasındaki ilişkiyi ifade eder.
  - **Bir-Çok (One-to-many (1:M veya 1..*)) bağıntı:** Bir varlık örneği, ilişkili varlığın birçok örneğiyle ilişkilendirilir.
    - Bir müşteri çok sayıda sipariş verebilir.
    - Her sipariş yalnızca bir müşteri tarafından verilir.

  - **Çok-Çok (Many-to-many (M:N or *..*)) bağıntı:** Bir varlık örneği, ilişkili varlığın birçok örneğiyle ilişkilidir ve ilişkili varlığın bir örneği, ilk varlığın birçok örneğiyle ilişkilidir.
    - Bir öğrenci çok sayıda ders alabilir.
    - Her ders çok sayıda öğrenci tarafından alınabilir.

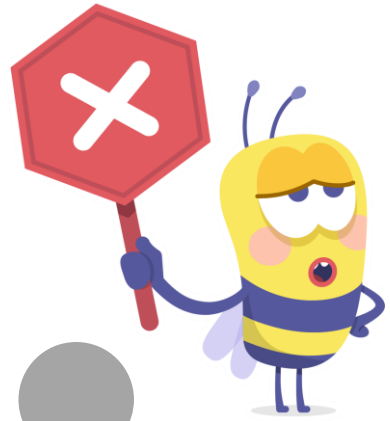
  - **Bir-Bir (One to one (1:1 or 1..1)) bağıntı**. Bir varlık örneği, ilişkili varlığın yalnızca bir örneği ile ilişkilendirilir.
    - Bir mağaza bir personel tarafından yönetilir.
    - Bir personel bir mağazayı yönetir.



Veri Modeli Temel Bileşenler

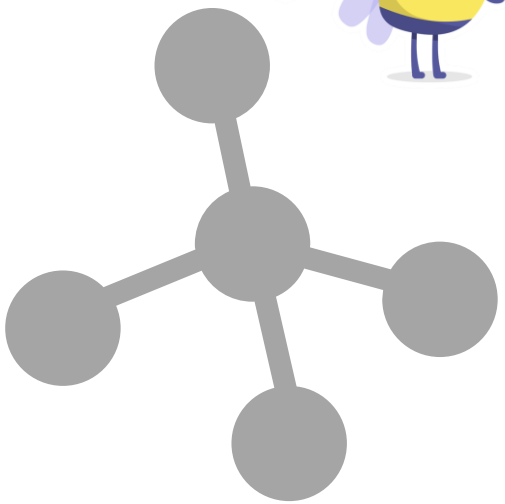
- **Tüm veri modellerinin** temel yapı taşları; varlıklar, nitelikler, ilişkiler ve kısıtlamalardır.

- **Kısıtlar (Constraints) -** Veri üzerindeki sınırlamalardır. Veri bütünlüğünün sağlanması açısından önemlidir.
  - Öğrenci notunun 0-100 arasında olması
  - T.C. kimlik numarasının 11 karakter olması
  - Aynı ürünün birden fazla kayıt edilememesi
  - Bir çalışanın maaşının 6000 TL ve 25000 TL arasında olması.



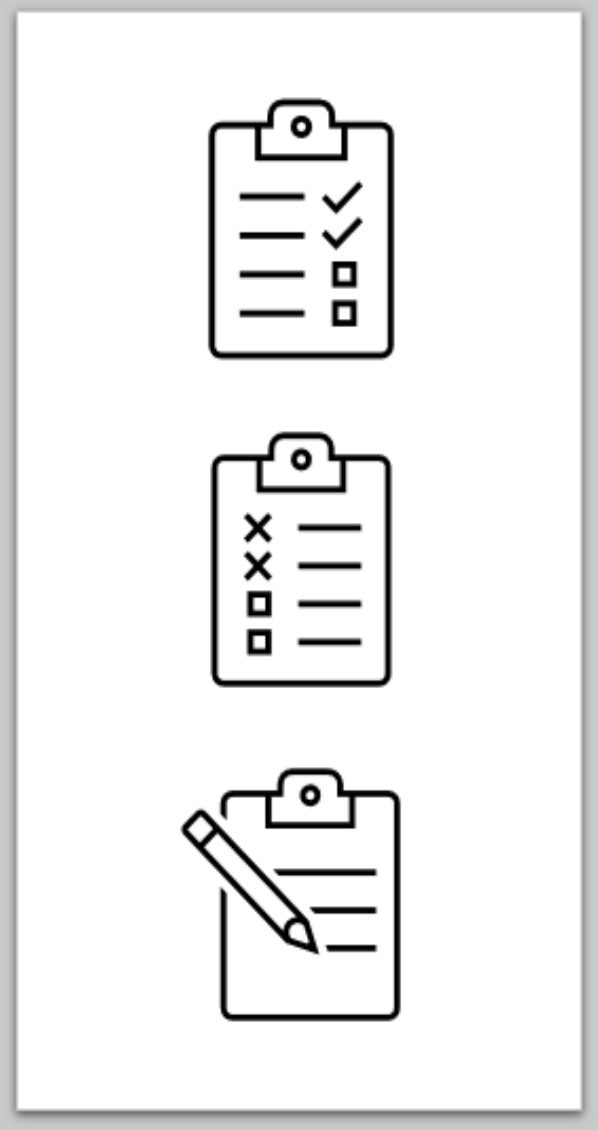
Varlıkları, nitelikleri, ilişkileri ve kısıtlamaları nasıl doğru bir şekilde tanımlarsınız?

➢ İlk adım, modellemekte olduğunuz problem alanı için iş kurallarını açıkça belirlemektir.



İş Kuralları

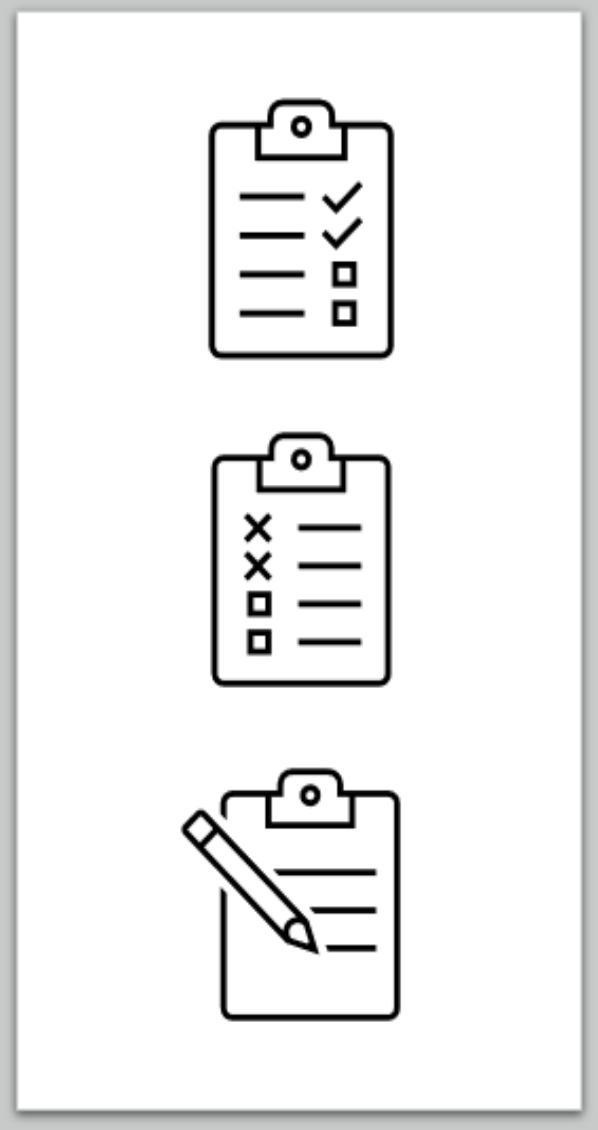
- Belirli bir organizasyondaki prosedürlerin veya ilkelerin kısa, kesin ve net açıklamalarıdır.
- Veritabanı tasarımı yapılacak organizasyon ile ilgili işleyiş, kural ya da yönetmeliğin özetlenmiş şekline iş kuralları denilebilir.
- Verilerin temel ve ayırt edici özelliklerini açıklar.
- İş kuralları ihtiyaçlar listesine benzer.
- Güncel tutulmalıdır.
- Anlaşılması kolay olmalıdır.
- İş kurallarının kaynağı, son kullanıcılar, yöneticiler, kural koyucular ve yazılı dokümanlar (standart, yönetmelik vs.) olabilir.
- İş kurallarını oluşturmak için doğrudan son kullanıcılarla görüşmek oldukça etkili bir çözümdür.
- Veri tabanı (varlık, nitelik, bağıntı ve kısıtlar) oluşturulurken iş kurallarına bakılır.
- Örnekler:
  - Bir müşteri birçok sayıda fatura oluşturabilir.
  - Bir fatura yalnızca bir müşteri tarafından oluşturulur.
  - 10'dan az veya 30'dan fazla öğrenci için bir eğitim oturumu planlanamaz.
  - Bir müşteri çok sayıda sipariş verebilir.
  - Her müşterinin adı, soyadı, telefon numarası vb. istenir.
  - Öğrenciler bir ara sınav ve bir final sınavına girerler.



İş Kuralları

- <u>Veritabanı tasarımı açısından iş kurallarının önemi;</u>
  - Kuruluşun veriye bakışını standart haline getirir.
  - Kullanıcılar ile tasarımcılar arasındaki iletişimi sağlar.
  - Tasarımcının verinin doğasını, önemini ve kapsamını anlamasını sağlar.
  - Tasarımcının iş süreçlerini anlamasını sağlar.
  - Tasarımcının doğru bir veri modeli geliştirmesine yardım eder (veriler arası ilişkiler ve kısıtların kolayca belirlenmesini sağlar).

- İş kuralları oluşturulduktan sonra, gerçekleştirilecek veri tabanının modellenmesi aşamasına geçilir.



İş Kurallarının Veri Modeline Dönüştürülmesi

- Genel bir kural olarak, bir iş kuralındaki bir «isim», modeldeki bir «varlığa» çevrilir ve isimleri ilişkilendiren bir «fiil (aktif veya pasif)» varlıklar arasında bir «bağıntıya» dönüşür.

  Örn: "Bir müşteri birçok fatura oluşturabilir" kuralı iki isim (müşteri ve fatura) ve isimleri ilişkilendiren bir fiil (oluşturmak) içerir.

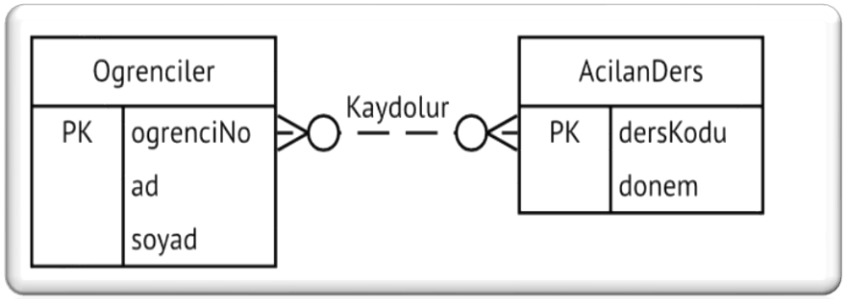
  - *Müşteri* ve *fatura*, ilgilenilen nesnelerdir ve ilgili varlıklar tarafından temsil edilmelidir.
  - Müşteri ve fatura arasında bir *oluşturma* ilişkisi vardır.

- Hakkında bilgi bulunan isim ya da isim tamlamaları varlık adayı iken, bilgi bulunmayanlar varlığa ait nitelik adayıdır.

  *Örn: "Müşterinin **ad, soyad, numara, adres** bilgileri saklanır."*

- Bağıntılar iki yönlüdür.
  - "Bir müşteri birçok fatura oluşturabilir" kuralı «Bir fatura yalnızca bir müşteri tarafından oluşturulabilir» iş kuralı ile tamamlanmış olur. Bir-Çok Bağıntı (1:M).
  - **1 öğretim üyesi çok** sayıda (4) **ders** verebilir - **1 ders** sadece **1 öğretim üyesi** tarafından verilebilir.
  - **1 kişi 1 bölüme** yönetici olabilir - **1 bölüm** sadece **1 kişi** tarafından yönetilebilir.
  - **1 öğrenci çok** sayıda **derse** kayıt yaptırabilir - **1 ders çok** sayıda **öğrenci** tarafından alınabilir.

- B'nin kaç örneği bir A örneğiyle ilişkilidir?- A'nın kaç örneği bir B örneğiyle ilişkilidir?
  - Örn: öğrenci ve açılan dersler:
    - Bir öğrenci kaç açılan dersi seçmiş olabilir? - Bir açılan derse kaç öğrenci kayıt olabilir?



İsimlendirme Kuralları

- Uygun bir isimlendirme kuralının kullanılması, veri modelinin tasarımcı, uygulama programcısı ve son kullanıcı arasındaki iletişimi kolaylaştırma becerisini geliştirecektir.

- Varlık adları, iş ortamındaki nesneleri açıklayıcı nitelikte olmalı ve kullanıcıların aşina olduğu terminolojiyi kullanmalıdır.

- Bir nitelik adı, o nitelik tarafından temsil edilen veriyi de açıklayıcı olmalıdır.
- Bir niteliğin adının önüne, içinde oluştuğu varlığın adı veya kısaltmasını eklemek de iyi bir uygulamadır.

  Örn: MUSTERI varlığının Kredi limiti niteliğinin MUS_KREDI_LIMIT olarak gösterilmesi.
  - MUS gösterimi bu niteliğin MUSTERI varlığına ait olduğunu gösterirken, KREDI_LIMIT adlandırması bu nitelikten tutulan verinin tanımlanmasını kolaylaştırır.

Veri Modellerinin Gelişimi

Nesil	Tarih	Veri Modeli	Örnek	Açıklama
Birinci	1960-1970 yılları	Dosya Sistemi	VMS/VSAM	• Genellikle IBM mainframe sistemlerinde kullanıldı • Yönetilen kayıtlar • İlişki yok
İkinci	1970 yılları	Hiyerarşik ve Ağ	IMS, ADABAS, IDS-II	• İlk veri tabanı sistemleri • Yönlendirmeli erişim
Üçüncü	1970 yıllarının ortaları	İlişkisel	• DB2 • Oracle • MS SQL Server • MySQL	• Kavramsal basitlik • Varlık İlişki modelleme • İlişkisel veri modelleme desteği
Dördüncü	1980 yıllarının ortaları	Nesne yönelimli Nesne/İlişkisel (O/R)	• Versan • Objectivity/DB • DB2 UDB • Oracle 12c	• Nesne/ilişlisel nesne veri tiplerini desteklemekte • Web veritabanları yaygınlaştı • Veri ambarları için StarSchema desteği
Beşinci	1990 yılları ortaları	XML Hybrid VTYS	• dbXML • Tamino • DB2 UDB • Oracle 12c • MS SQL Server	• Yapısal olmayan veri desteği • XML dokümanları desteği • Geniş veri tabanlarını destekler
Gelişen Modelller: NoSQL	2000 li yılların başları ve günümüz	Key-value store Column store	• Simple DB(Amazon) • BigTable(Google) • Cassandra (Apache) • MongoDB • Riak	• Dağıtık ve yüksek ölçeklenebilir • Yüksek performans, hata toleransı • Çok yüksek depolama (petabyte) • Ayrık veriler için uygun • Özel API (Application programming interface)

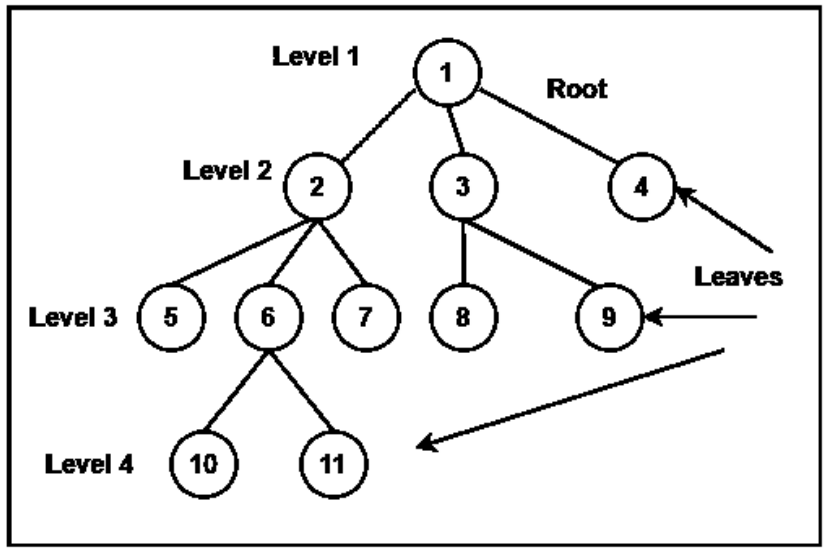
Veri Modellerinin Gelişimi

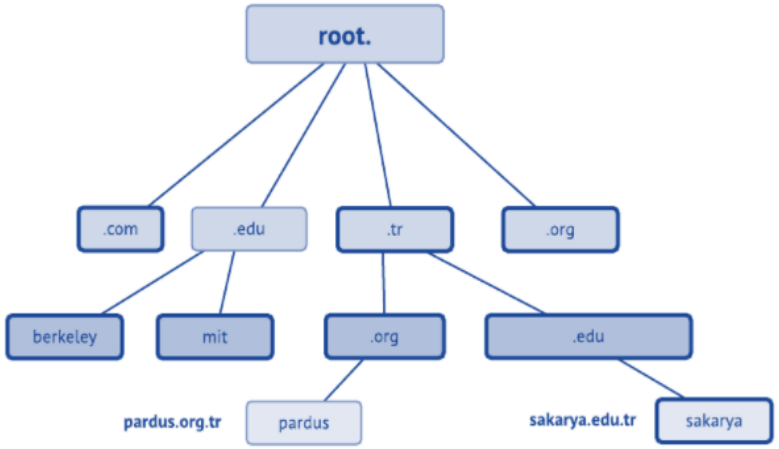
- **Dosya Sistemleri**
  - 1960-1970 lerde çoğunlukla IBM ana çatı (mainframe) sistemlerde kullanılmıştır.
  - Dosyalar arasında ilişki yoktur.

Eğitim Birimi	Program	GrupAd	OgretimTur	DersKod	DersAd	DersGun
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	MAT 217	SAYISAL ANALİZ	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 303	BİLGİSAYAR AĞLARI VE İNTERNET	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 303	BİLGİSAYAR AĞLARI VE İNTERNET	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 303	BİLGİSAYAR AĞLARI VE İNTERNET	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 305	VERİTABANI YÖNETİM SİSTEMLERİ	Pazartesi
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 305	VERİTABANI YÖNETİM SİSTEMLERİ	Pazartesi
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 305	VERİTABANI YÖNETİM SİSTEMLERİ	Pazartesi
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	B	1	ISE 305	VERİTABANI YÖNETİM SİSTEMLERİ	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	B	1	ISE 305	VERİTABANI YÖNETİM SİSTEMLERİ	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	B	1	ISE 305	VERİTABANI YÖNETİM SİSTEMLERİ	Perşembe
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 307	ÜRETİM VE SERVİS SİSTEMLERİ YÖNETİMİ	Salı
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 307	ÜRETİM VE SERVİS SİSTEMLERİ YÖNETİMİ	Salı
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 307	ÜRETİM VE SERVİS SİSTEMLERİ YÖNETİMİ	Salı
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 307	ÜRETİM VE SERVİS SİSTEMLERİ YÖNETİMİ	Salı
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 309	WEB PROGRAMLAMA	Pazartesi
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ	BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ PR.	A	1	ISE 309	WEB PROGRAMLAMA	Pazartesi

Veri Modellerinin Gelişimi

- **Hiyerarşik Model**
  - 1960'larda karmaşık üretim projeleri için büyük miktarda veriyi yönetmek üzere geliştirildi.
  - Veriler ağaç yapısı şeklinde organize edilir.
  - Düzeyler (level) veya segmentler (segment) içerir. Segment, bir dosya sistemindeki «kayıt» türüne eşdeğerdir.
  - Ebeveyn-çocuk (parent-child) arasında 1:M ilişkisi vardır. Kayıtların sadece 1 ebeveyn (parent) kaydı vardır.

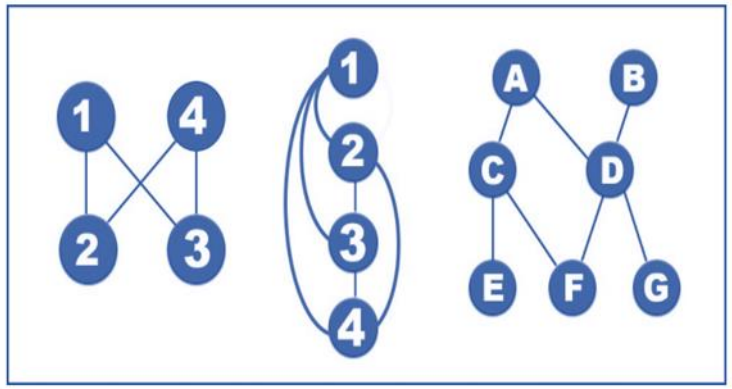


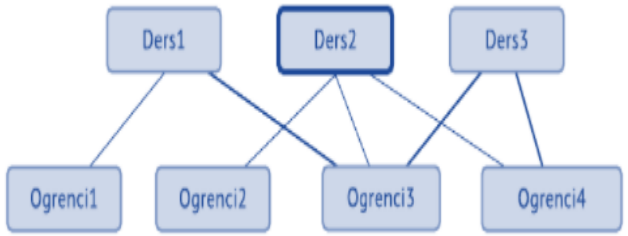


Veri Modellerinin Gelişimi

Ağ Modeli

- 1970'lerde geliştirilmiştir. Veri tabanı başarımını artırmak üzere daha karmaşık ilişkilere izin verilir.
- Hiyerarşik modelden farklı olarak kayıtların birden fazla ebeveyn (parent) kayıtları olabilir.
- Ağ modeliyle birlikte ortaya çıkan ve hala kullanılan bazı kavramlar:
  - **Şema:** Tüm veri tabanının, veritabanı yöneticisi tarafından görünen kavramsal organizasyonu.
  - **Alt şema:** Veri tabanının istenen bilgiyi üreten uygulama programı tarafından görünen kısmı.
  - **Veri işleme dili (data manipulation language, DML):** Veri tabanında bulunan verilerin, sorgulama işlemleri yapılarak güncellenmesi, yeni verilerin eklenmesi ve olan verilerin silinme işlemlerinin yapılmasını sağlayan dil.
  - **Veri tanımlama dili (data definition language, DDL):** Veri tabanında bulunan verilerin tip, yapı ve kısıtlamalarının tanımlanmasını sağlayan dil.
  - **Anlık sorgu (ad hoc query):** Yazılımlarla birlikte gelmeyen kullanıcının kendi oluşturduğu sorgulara verilen isimdir.
- Ağ modelinin dezavantajı, çok basit sorgular için bile karmaşık program kodlarının kullanımını gerektirmesidir.





Veri Modellerinin Gelişimi

İlişkisel Model (1/3)

- 1970'de E. F. Codd tarafından ortaya atılmıştır. (IBM)
  - "A Relational Model of Data for Large Shared Databanks" (Communications of the ACM, June 1970, pp. 377–387).
- Matematiksel küme teorisine dayanır ve verileri ilişkiler olarak temsil eder.
- Her ilişki (tablo) kavramsal olarak, kesişen satır ve sütunlardan oluşan iki boyutlu bir yapı olarak temsil edilir.
- İlişkisel Veritabanı Yönetim Sistemleri (Relational Database Management Systems, RDBMS) tarafından kullanılır.
- RDBMS'nin en önemli özelliklerinden birisi ilişkisel modelin karmaşık yapısını kullanıcıdan gizlemesidir.
- RDBMS, tüm fiziksel ayrıntıları yönetirken, kullanıcı ilişkisel veritabanını verilerin depolandığı bir tablolar koleksiyonu olarak görür.

A Relational Model of Data for Large Shared Data Banks

E. F. Codd
IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

Existing noninferential, formatted data systems provide users with tree-structured files or slightly more general network models of the data. In Section 1, inadequacies of these models are discussed. A model based on n-ary relations, a normal form for data base relations, and the concept of a universal data sublanguage are introduced. In Section 2, certain operations on relations (other than logical inference) are discussed and applied to the problems of redundancy and consistency in the user's model.

KEY WORDS AND PHRASES: data bank, data base, data structure, data organization, hierarchies of data, networks of data, relations, derivability, redundancy, consistency, composition, join, retrieval language, predicate calculus, security, data integrity
CR CATEGORIES: 3.70, 3.73, 3.75, 4.20, 4.22, 4.29

The relational view (or model) of data described in Section 1 appears to be superior in several respects to the graph or network model [3, 4] presently in vogue for non-inferential systems. It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation and organization of data on the other.

A further advantage of the relational view is that it forms a sound basis for treating derivability, redundancy and consistency of relations—these are discussed in Section 2. The network model, on the other hand, has spawned a number of confusions, not the least of which is mistaking the derivation of connections for the derivation of relations (see remarks in Section 2 on the "connection trap").

Finally, the relational view permits a clearer evaluation of the scope and logical limitations of present formatted data systems, and also the relative merits (from a logical standpoint) of competing representations of data within a single system. Examples of this clearer perspective are cited in various parts of this paper. Implementations of systems to support the relational model are not discussed.

1.2. Data Dependencies in Present Systems

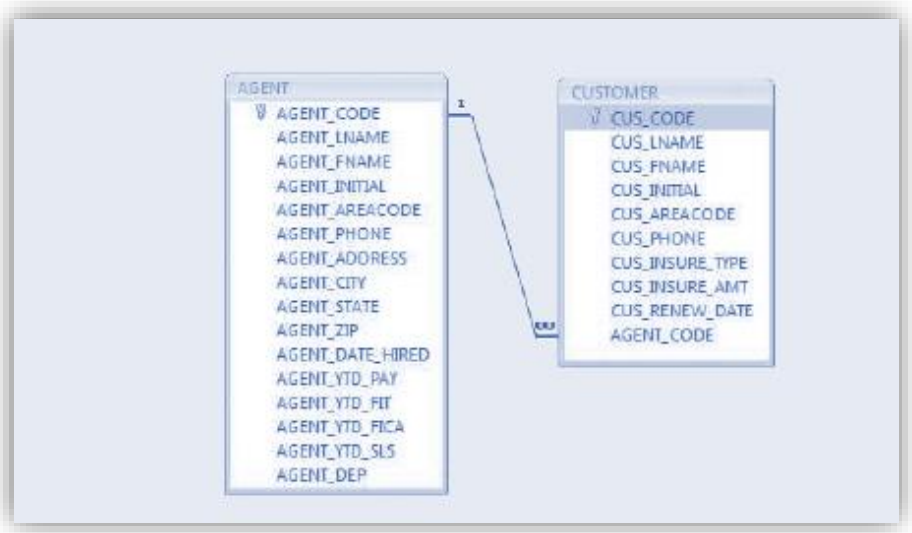
The provision of data description tables in recently developed information systems represents a major advance toward the goal of data independence [5, 6, 7]. Such tables facilitate changing certain characteristics of the data representation stored in a data bank. However, the variety of data representation characteristics which can be changed *without logically impairing some application programs* is still quite limited. Further, the model of data with which

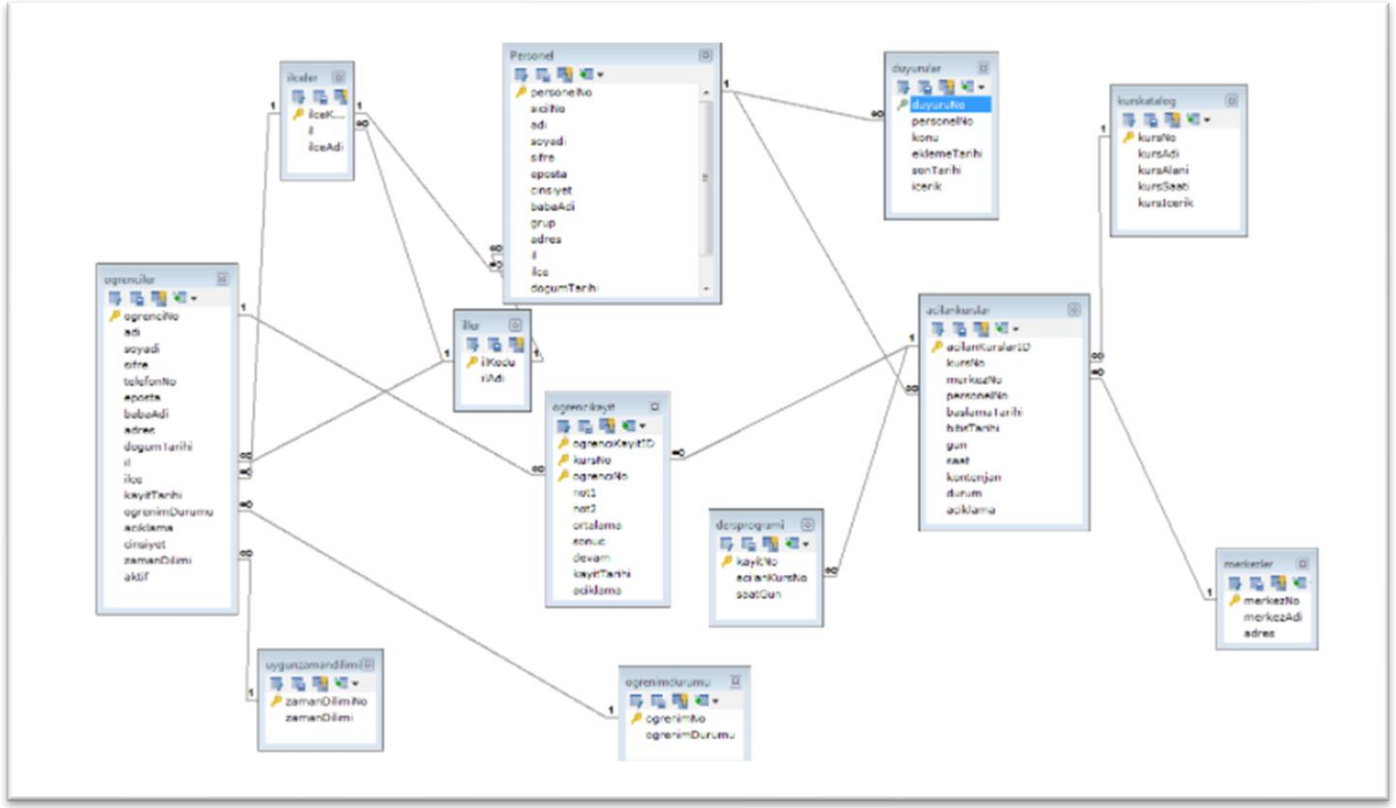
https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf

Veri Modellerinin Gelişimi

İlişkisel Model (2/3)

- Tablolar birbirlerine ortak alanlarla bağlanırlar.(sütundaki bir değer).
- İlişkisel şema (relational diagram); varlıklar, varlıkların nitelikleri ve aralarındaki bağlantıların gösteriminden oluşur.

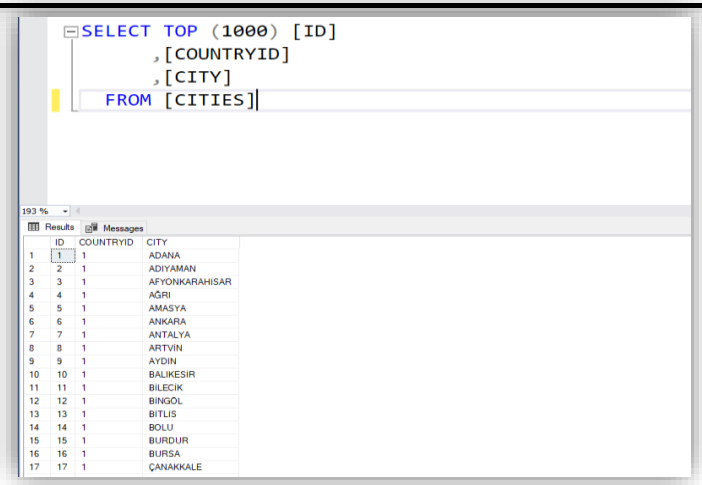


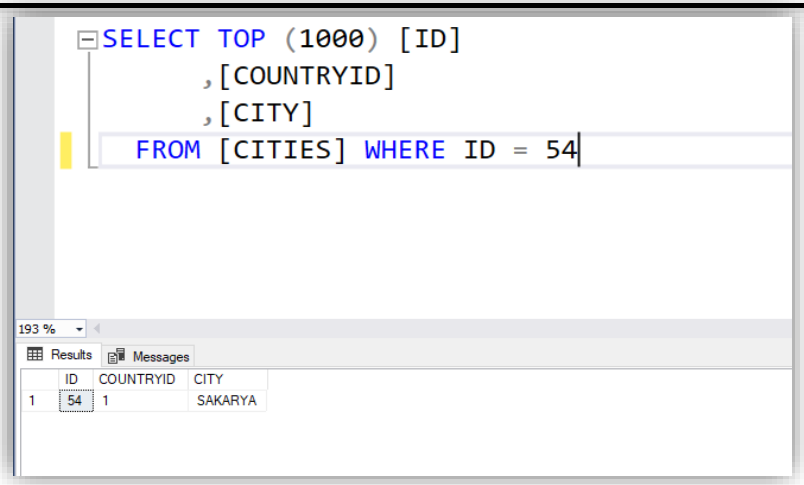


Veri Modellerinin Gelişimi

İlişkisel Model (3/3)

- İlişkisel veritabanı modelinin en güçlü yanlarından birisi, verileri yönetmek için yapısal sorgulama dili (structured query language, SQL) kullanılıyor olmasıdır.
- SQL dili nasıl yapılması gerektiğini anlatmak yerine ne yapılması gerektiğinin ifade edildiği basit bir dildir.
- Bu nedenle, SQL kullanılarak veri tabanlarının tasarımı ve yönetimi daha kolaydır.
- Son kullanıcı açısından bakıldığında, herhangi bir SQL tabanlı ilişkisel veritabanı uygulaması üç bölümden oluşur: Kullanıcı arayüzü, veritabanında depolanan tablolar (veritabanı) ve SQL motoru.
- Arayüz: Kullanıcının verilerle etkileşime girmesine izin verir(SQL)
- Verilerin saklandığı veritabanı: İlişkisel bir veritabanında, tüm veriler tablolarda depolanmış olarak algılanır.
- SQL motoru (SQL engine): Tüm sorguları veya veri taleplerini yürütür. SQL motoru, VTYS yazılımının bir parçasıdır. Son kullanıcı, tablo yapıları oluşturmak ve veri erişimi ve tablo bakımı gerçekleştirmek için SQL kullanır. SQL motoru, büyük ölçüde perde arkasında ve son kullanıcının bilgisi olmadan tüm kullanıcı isteklerini işler.

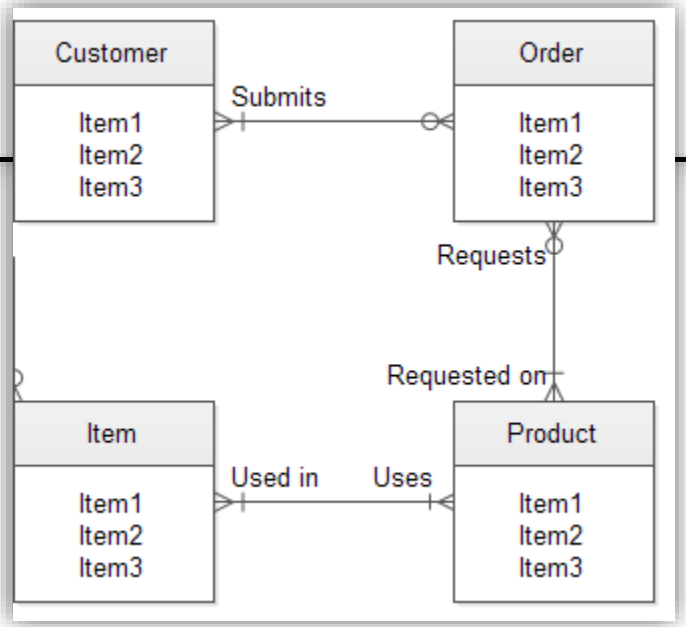


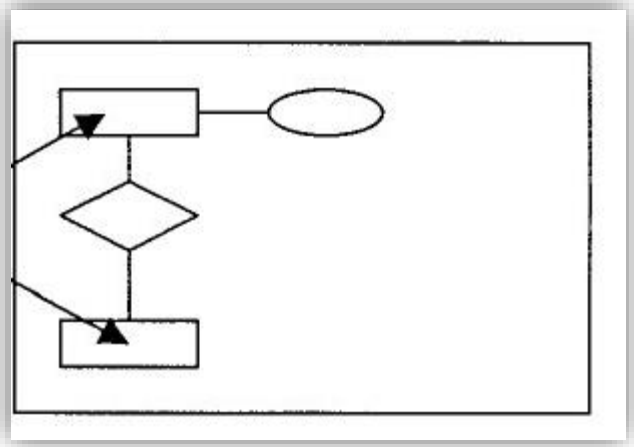


Veri Modellerinin Gelişimi

Varlık Bağıntı Modeli

- İlişkisel model, hiyerarşik ve ağ modellerine göre büyük bir gelişme olsa da, onu etkili bir veritabanı tasarım aracı yapacak özelliklerden hâlâ yoksundu.
- 1976'da Peter Chen tarafından önerilmiştir.
- Veritabanı tasarımcıları, varlıkların ve ilişkilerinin resmedildiği bir grafik aracı kullanmayı tercih ederler. Bu nedenle, varlık bağıntı (ER) modeli veya ERM, veri modelleme için yaygın olarak kabul edilen bir standart haline gelmiştir.
- İlişkisel modelinin tamamlayıcısı olduğu için kullanımı oldukça yaygınlaşmıştır.
- ER modelleri normalde veritabanı bileşenlerini modellemek için grafik gösterimler kullanan bir varlık ilişki diyagramında (ERD) temsil edilir.
- Chen gösterimi ve Crow's Foot gösterimi sıkça kullanılan gösterim şekillerindendir.





Veri Modellerinin Gelişimi

Varlık Bağıntı Modeli

- *ER model bileşenleri:*

- ***Varlık(Entity):*** Hakkında bilgi toplanıp depolanan her şey.
  - VBD'de dikdörtgen ile temsil edilir.
  - Varlık ismi dikdörtgenin ortasına yazılır.
  - Varlık ismi genellikle büyük harflerle ve tekil formda yazılır: PERSONEL,MÜŞTERİ

- ***Nitelik (Attributes)*** Varlığın belirli özelliklerini tanımlar. ÖĞRENCİ varlığının bir Öğrenci Numarası, Adı ve Soyadı bulunur.

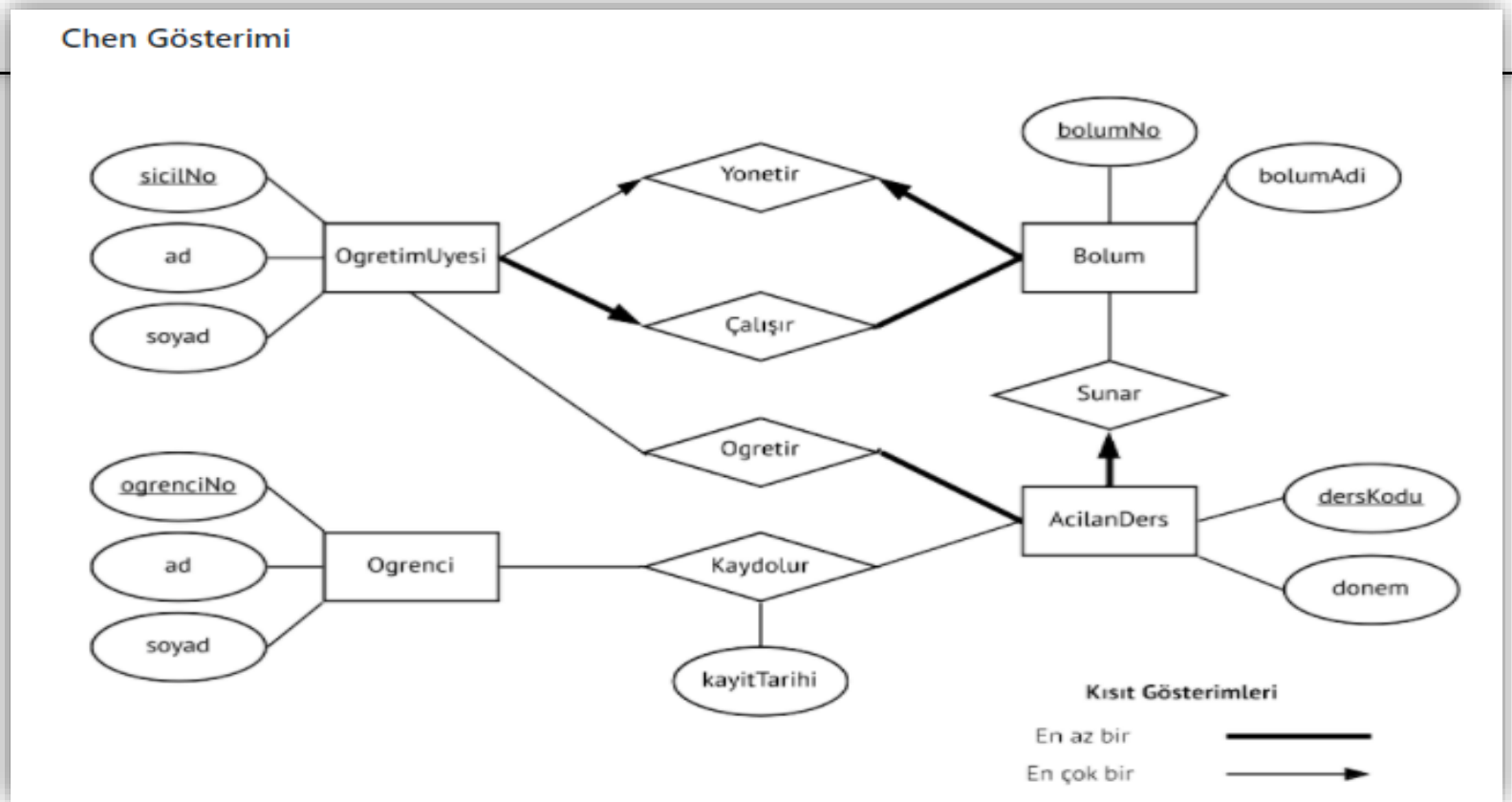
- ***Bağıntı (Relationships)***. Veriler arasındaki ilişkileri tanımlar.
  - Bir-Çok (one-to-many (1:M)), Çok-Çok (many-to-many (M:N)), ve Bir-Bir (one-to-one (1:1)).
  - İlişki adı genellikle aktif veya pasif bir fiildir.: Bir ÇALIŞAN yalnızca bir MAĞAZAYI *yönetir.*

Veri Modellerinin Gelişimi

Varlık Bağıntı Modeli

Chen Gösterimi

- Bağlantılar her varlık kutusunun yanında yazılır.
- Bağıntılar, ilgili varlıklara bir ilişki çizgisi ile bağlanan bir elmas şekliyle temsil edilir.
- Bağıntı adı elmas şeklinin içerisine yazılır.

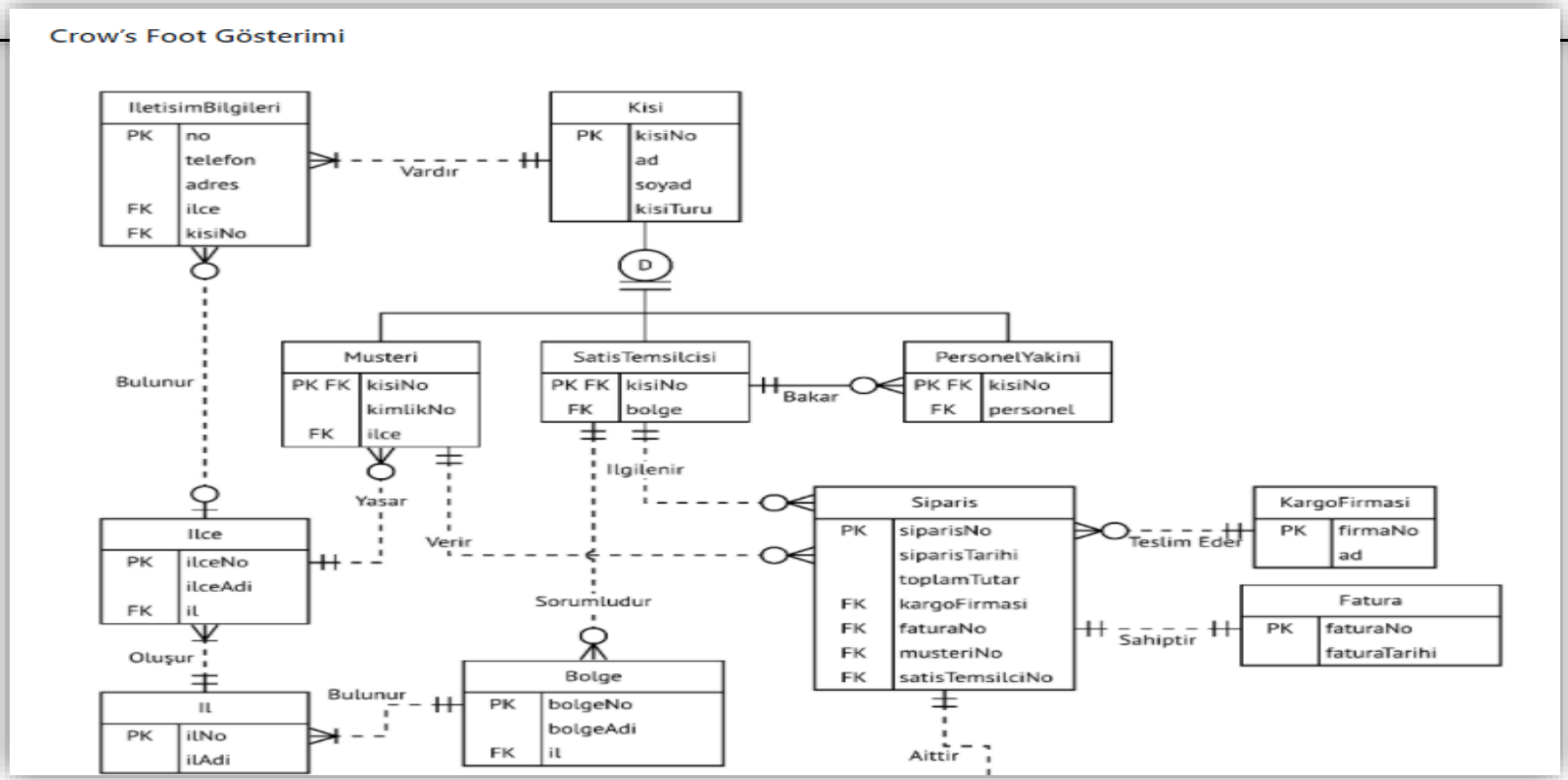


Veri Modellerinin Gelişimi

Varlık Bağıntı Modeli

Crow's Foot Gösterimi.

- Crow's Foot adı, ilişkinin "çok" tarafını temsil etmek için kullanılan üç kollu sembolden türetilmiştir.
- Bağlantılar sembollerle temsil edilir.
- Bağıntı adı, bağıntı çizgisinin üzerine yazılır.

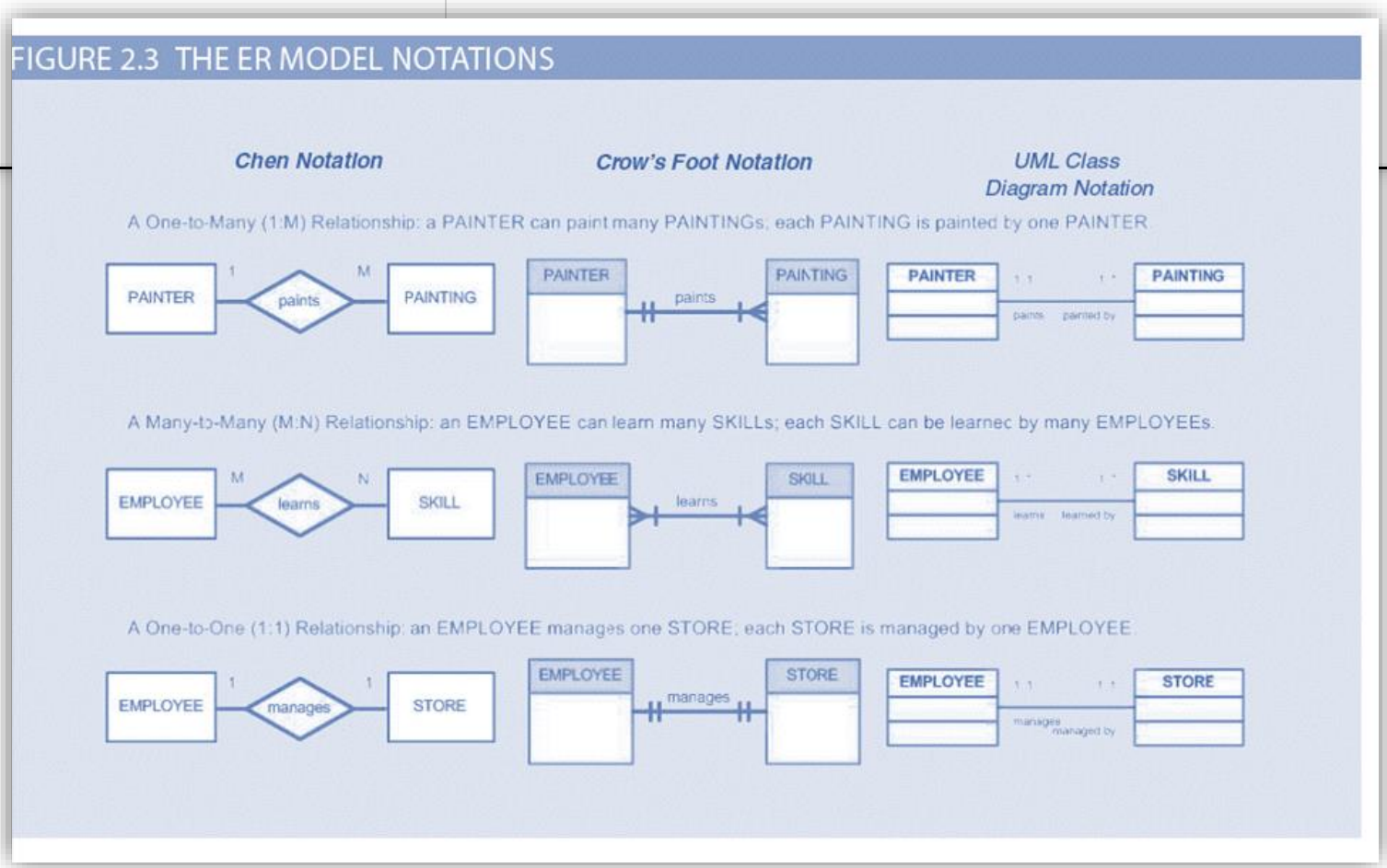


Veri Modellerinin Gelişimi

Varlık Bağıntı Modeli

UML Class Diagram gösterimi

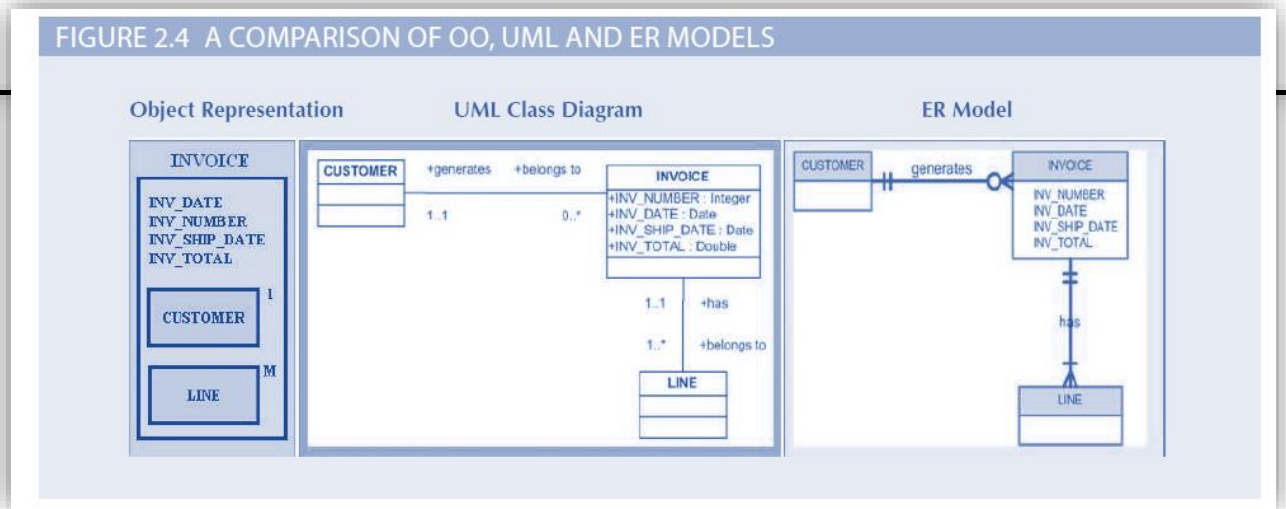
Bağıntılar semboller olan (1…1, 1…*) çizgiler ile gösterilir.
Bağıntının her iki tarafında da isimler kullanılır.



Veri Modellerinin Gelişimi

Nesne Yönelimli Model (Object Oriented Model)

- Nesneye yönelik veri modelinde (OODM), hem veriler hem de ilişkileri, nesne olarak bilinen tek bir yapıda bulunur.
- Nesne yönelimli programlama paradigmasından esinlenerek geliştirilen modeldir.
- Nesne, gerçek dünya varlığının bir soyutlamasıdır. Varlık bağıntı (VB - ER) modelindeki varlık (entity) bu modelde nesne olarak adlandırılır.
- Nesne hakkındaki bilgi, VB modelindeki niteliklere karşılık gelir. Ör. KİŞİ nesnesi Kimlik No,Ad,Soyad, Doğum Tarihi niteliklerini içerir.
- Sınıf, paylaşılan yapıya (niteliklere) ve davranışa (yöntemler - methods) sahip benzer nesnelerin bir koleksiyonudur. Sınıf, ER modelinin varlık kümesine benzer. Bununla birlikte, yöntemler olarak bilinen bir dizi prosedür içerir.
- VB modelinden farklı olarak sınıflar üye fonksiyonlara da sahiptirler. Kisi ara, Ad listele vb.
- Kalıtım, sınıf hiyerarşisi içindeki bir nesnenin, üstündeki sınıfların niteliklerini ve yöntemlerini miras alma yeteneğidir. Örneğin, MÜŞTERİ ve PERSONEL adlı iki sınıf, KİŞİ sınıfından alt sınıflar olarak oluşturulabilir.
- Nesneye yönelik veri modelleri genel olarak Birleşik Modelleme Dili (Unified Modeling Language - UML) sınıf diyagramları kullanılarak gösterilir.



Veri Modellerinin Gelişimi

Yeni Veri Modelleri

Nesne/İlişkisel (Object/Relational) Model

- İlişkisel modelle nesne yönelimli modelin birleştirilmesi sonucu ortaya çıkmıştır.

Genişletilebilir İşaretleme Dili (Extensible Markup Language, XML)

- Çoğunlukla, farklı platformlar arası veri değişimi için kullanılan veri tanımlama standardıdır. Yapısal olmayan verileri tanımlamak için de kullanılır.

JavaScript Nesne Gösterimi (JavaScript Object Notation, JSON)

- Çoğunlukla, farklı platformlar arası veri değişimi için kullanılan veri tanımlama standardıdır. Yapısal olmayan verileri tanımlamak için de kullanılır.

NoSQL

- İlişkisel modelin yetersiz kaldığı büyük hacimli verilerin yönetimi için tercih edilir.

Veri Modellerinin Gelişimi

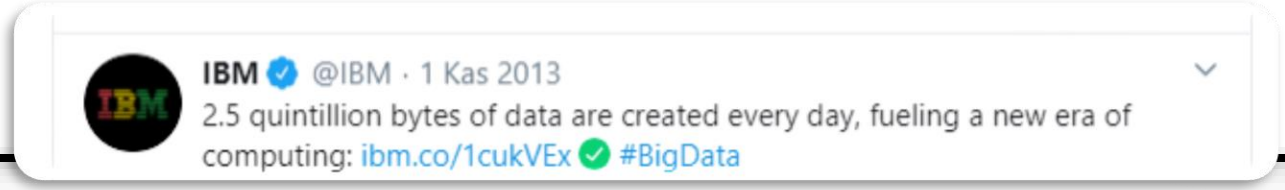
Gelişen Yeni Veri Modelleri – Big Data

- Toplanan veri miktarı her gün katlanarak artıyor.
    - IBM, "Her gün 2,5 kentilyon bayt veri oluşturuyoruz - o kadar ki bugün dünyadaki verilerin yüzde 90'ı yalnızca son iki yılda oluşturuldu ".

- Hızlı veri artışı, performans, ölçeklenebilirlik ve daha düşük maliyetleri yönetme ve bunlardan yararlanma ihtiyacı -> "Büyük Veri".

- **Büyük Veri (Big Data)**, aynı anda makul bir maliyetle yüksek performans ve ölçeklenebilirlik sağlarken, büyük miktarlarda web ve sensörler tarafından oluşturulan verileri yönetmek ve bunlardan iş görüleri elde etmek için yeni ve daha iyi yollar bulmaya yönelik bir hareketi ifade eder.

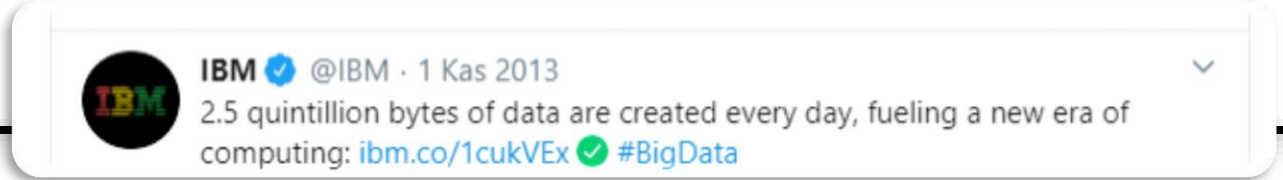
- **3V s** (Big Data veri tabanlarının temel özellikleri - Douglas Laney):
    - *Volume (Hacim):* saklanan veri miktarı.
    - *Velocity (Hız) :* sadece verilerin büyüme hızına değil, aynı zamanda bilgi ve içgörü üretmek için bu verileri hızlı bir şekilde işleme ihtiyacı.
    - *Variety (Çeşitlilik):* birden çok farklı veri biçiminde toplanan veriler



Veri Modellerinin Gelişimi

Gelişen Yeni Veri Modelleri – Big Data

- İlişkisel yaklaşım her zaman Büyük Veri zorlukları olan kuruluşların ihtiyaçlarını karşılayamamakta.
- Yapılandırılmamış, sosyal medya ve sensörler tarafından oluşturulan verileri satır ve sütunların geleneksel ilişkisel yapısına uydurmak her zaman mümkün değildir.
- Günlük olarak milyonlarca satırlık çok formatlı (yapılandırılmış ve yapılandırılmamış) veri eklemek, kaçınılmaz olarak daha fazla depolama, işleme gücü ve ilişkisel ortamda bulunmayan karmaşık veri analizi araçlarına ihtiyaç duyulmasına yol açacaktır.
- Kuruluşların çok sayıda biçimdeki devasa veri depolarını uygun maliyetli yöntemlerle işlemesine olanak tanıyan yeni Büyük Veri Teknolojilerini kullanan şirketler.
  - *Hadoop*, Java tabanlı, açık kaynak, yüksek hızlı, hataya dayanıklı dağıtılmış depolama ve hesaplama ortamı.
  - *MapReduce*, hızlı veri analizi hizmetleri sağlayan açık kaynaklı uygulama programlama arayüzü (API)
  - *NoSQL veri tabanları.* Yapılandırılmış ve yapılandırılmamış verileri verimli şekillerde depolayan büyük ölçekli dağıtılmış bir veritabanı sistemi



Veri Modellerinin Gelişimi

NoSQL Veri Tabanları

- **NoSQL** (Not only SQL); genellikle geleneksel ilişkisel veritabanı modeline dayalı olmayan yeni nesil veritabanı yönetim sistemlerini tanımlamak için kullanılır.
- NoSQL, ilişkisel veritabanlarında bulunan geleneksel tablo yapılarının ötesinde verilerin depolanması ve alınması için esnek şemalar sağlayan veritabanı tasarımına yönelik bir yaklaşımdır.
- NoSQL veritabanları, bulut, büyük veri ve yüksek hacimli web ve mobil uygulamalar çağında yakın zamanda daha popüler hale geldi.
- En yaygın NoSQL veri tabanı türleri: key-value, document, column ve graph veri tabanlarıdır.
- «NoSQL» deki «No» ifadesi «Not only» için bir kısaltmadır, «SQL Yok» anlamında değildir. Birçok NoSQL veritabanı SQL benzeri sorguları desteklemektedir.
- İlişkisel modele dayanmazlar.
- Dağıtık veritabanı mimarilerini desteklerler.
- Yüksek ölçeklenebilirlik, yüksek kullanılabilirlik ve hata toleransı sağlarlar.
- Çok büyük miktarlarda seyrek veriyi desteklerler.
- İşlem tutarlılığından çok performansa yöneliktirler.

NoSQL, günümüzde veritabanı teknolojilerindeki en yeni kavramlardan biridir. Ancak bu, veri yönetiminde ortaya çıkan birçok trendden yalnızca biridir. Hangi veritabanı teknolojisini kullanırsanız kullanın, her teknolojinin artılarını ve eksilerini anlayarak iş için en iyi aracı seçebilmeniz gerekir!

Veri Modelleri - Özet

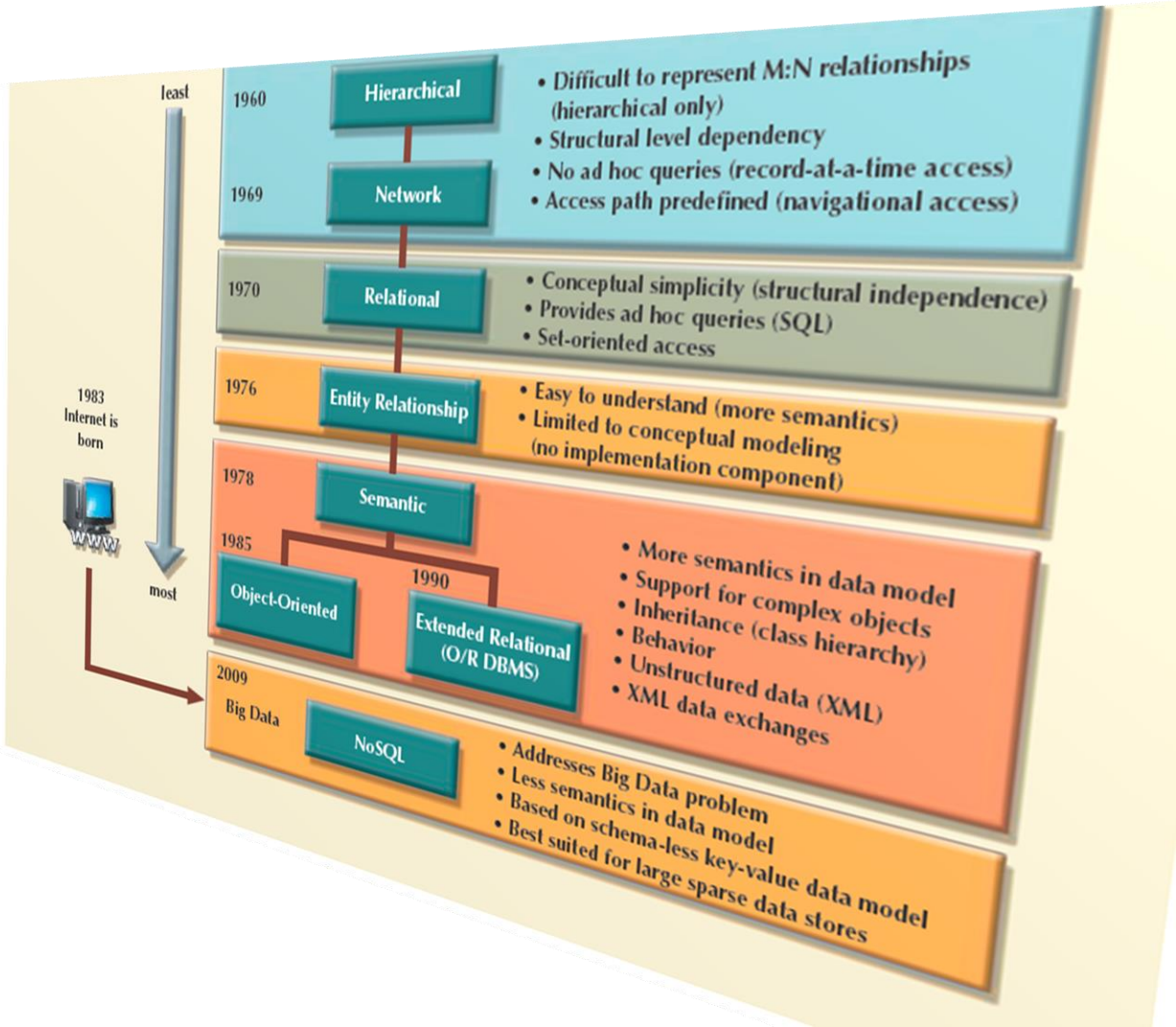
Bir *veri modeli*, bir dereceye kadar kavramsal basitlik göstermelidir.

Veri modeli net olmalı ve problem alanına uygulanabilir olmalıdır.

Gerçek dünyayı olabildiğince yakından temsil etmelidir.

Ağ modeli, hiyerarşik modelin yerini aldı çünkü karmaşık (çoka-çok) ilişkileri temsil etmeyi çok daha kolay hale getirdi.

İlişkisel model ise hiyerarşik ve ağ modellerinin üzerine birkaç yönden avantaj sağlar: daha basit veri gösterimi, üstün veri bağımsızlığı ve kullanımı kolay sorgulama dili vb.; Bu özellikler ilişkisel modeli iş uygulamaları için tercih edilen bir veri modeli yapmaktadır.



Veri Modelleri - Özet

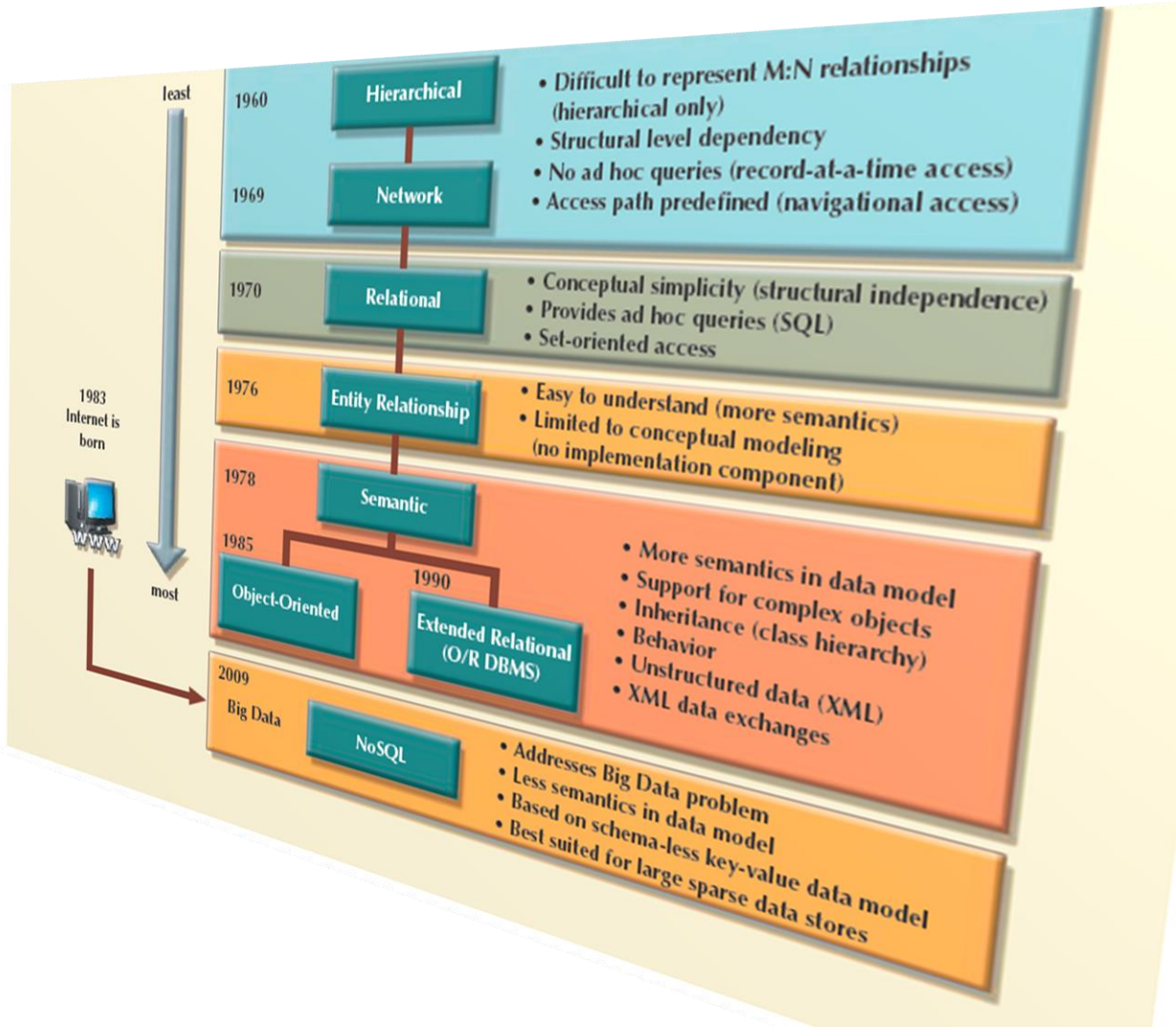
OO veri modeli karmaşık veriler için destek sağlamıştır.

ERDM, ilişkisel modele birçok OO özelliği ekledi ve iş ortamında güçlü pazar payını korumasına izin verdi.

Son yıllarda, Büyük Veri kavramı geleneksel veri yönetiminden bir kopuşu temsil eden verileri modellemek, depolamak ve yönetmek için alternatif yollar geliştirilmesini teşvik etti.

Kavramsal modeller üst seviye veri modellemesi için daha uygunken, uygulama modelleri, uygulama amaçları için depolanan verileri yönetmek için daha iyidir.

Varlık ilişkisi modeli kavramsal bir modelin bir örneğiyken, hiyerarşik ve ağ modelleri uygulama modellerinin örnekleridir. Aynı zamanda, ilişkisel model ve OODM gibi bazı modeller hem kavramsal hem de uygulama modelleri olarak kullanılabilir.



Veri Modelleri - Özet

Veri Modeli	Avantajlar	Dezavantajlar
Hiyerarşik	➢ Veri paylaşımını destekler ➢ Ebeveyn/Çocuk ilişkisi kavramsal kolaylık sağlar ➢ Ebeveyn/Çocuk ilişkisi veri bütünlüğünü destekler ➢ 1:M bağıntılarda etkili	➢ Karmaşık uygulama zorluğu, fiziksel veri depolama özellikleri hakkında bilgi gerektirir. ➢ Navigasyonal sistemi; karmaşık uygulama geliştirme; hiyerarşik yol bilgisi gerektirir. ➢ Yapıdaki değişiklikler tüm uygulama programlarında değişiklik yapılmasını gerektirir. ➢ Uygulama sınırlamaları vardır (multiparent veya M: N ilişkileri yok). ➢ VTYS'de veri tanımı veya veri işleme dili yoktur. ➢ Standart eksikliği var.
Ağ	➢ Kavramsal basitlik en azından hiyerarşik modelinkine eşittir. ➢ M: N ve multiparent gibi daha fazla ilişki türünü ele alır. ➢ Veri erişimi, hiyerarşik ve dosya sistemi modellerinden daha esnektir. ➢ Veri sahibi / üye ilişkisi veri bütünlüğünü destekler. ➢ Standartlara uygunluk vardır. ➢ VTYS'de Veri tanımlama dilini (DDL) veri işleme dilini (DML) içerir.	➢ Sistem karmaşıklığı verimliliği sınırlar - hala bir navigasyonel sistemi. ➢ Navigasyonal sistemi; karmaşık uygulama, uygulama geliştirme ve yönetime neden olur. ➢ Yapısal değişiklikler, tüm uygulama programlarında değişiklik yapılmasını gerektirir.

Veri Modelleri - Özet

Veri Modeli	Avantajlar	Dezavantajlar
İlişkisel	➢ Yapısal bağımsızlık, bağımsız tabloların kullanılmasıyla desteklenir. Bir tablonun yapısındaki değişiklikler veri erişimini veya uygulama programlarını etkilemez. ➢ Tablo görünümü kavramsal basitliği önemli ölçüde geliştirir, böylece daha kolay veri tabanı tasarımı, uygulaması, yönetimi ve kullanımını teşvik eder. ➢ Ad hoc sorgu yeteneği SQL'e dayanır. ➢ Güçlü RDBMS, son kullanıcıyı fiziksel düzeydeki ayrıntılardan izole eder ve uygulama ile yönetim kolaylığını geliştirir.	➢ RDBMS, önemli donanım ve sistem yazılımı ek yükü gerektirir. ➢ Kavramsal basitlik göreceli olarak eğitimsiz insanlara iyi bir sistemi kötü kullanma araçları verir ve kontrol edilmezse, dosya sistemlerinde bulunan veri anormalliklerinin aynısını üretebilir. ➢ Bireysel olarak «bilgi adalarını» problemlerine destek verebilir ve departmanlar kendi uygulamalarını geliştirebilirler.
Varlık İlişki	➢ Görsel modelleme olağanüstü kavramsal basitlik sağlar. ➢ Görsel temsil, onu etkili bir iletişim aracı haline getirir. ➢ İlişkisel model ile bütünleşmiştir.	➢ Sınırlı kısıt temsili vardır. ➢ Sınırlı ilişki temsili vardır. ➢ Veri işleme dili yoktur. ➢ Kalabalık görüntülerden kaçınmak için nitelikler varlıklardan kaldırıldığında bilgi içeriği kaybı meydana gelir. (Bu sınırlama, sonraki grafik sürümlerde ele alınmıştır.)

Veri Modelleri - Özet

Veri Modeli	Avantajlar	Dezavantajlar
Nesne Yönelimli	➢ Anlamsal içerik eklenir. ➢ Görsel sunum anlamsal içeriği içerir. ➢ Miras, veri bütünlüğünü destekler.	➢ Standartların yavaş geliştirilmesi, satıcıların kendi geliştirmelerini tedarik etmelerine neden olarak, yaygın olarak kabul edilen bir standardı ortadan kaldırmıştır. ➢ Karmaşık bir navigasyonel sistemidir. ➢ Dik bir öğrenme eğrisi var. ➢ Yüksek sistem yükü işlemleri yavaşlatır.
NoSQL	1. Yüksek ölçeklenebilirlik, kullanılabilirlik ve hata toleransı sağlanır. 2. Düşük maliyetli kullanıma hazır donanımı kullanır. 3. Büyük Veriyi destekler. 4. Key-value Anahtar-değer modeli, depolama verimliliğini artırır.	1. Karmaşık programlama gereklidir. 2. İlişki desteği yoktur - yalnızca uygulama kodu ile.

Veri Soyutlama (Data Abstraction)

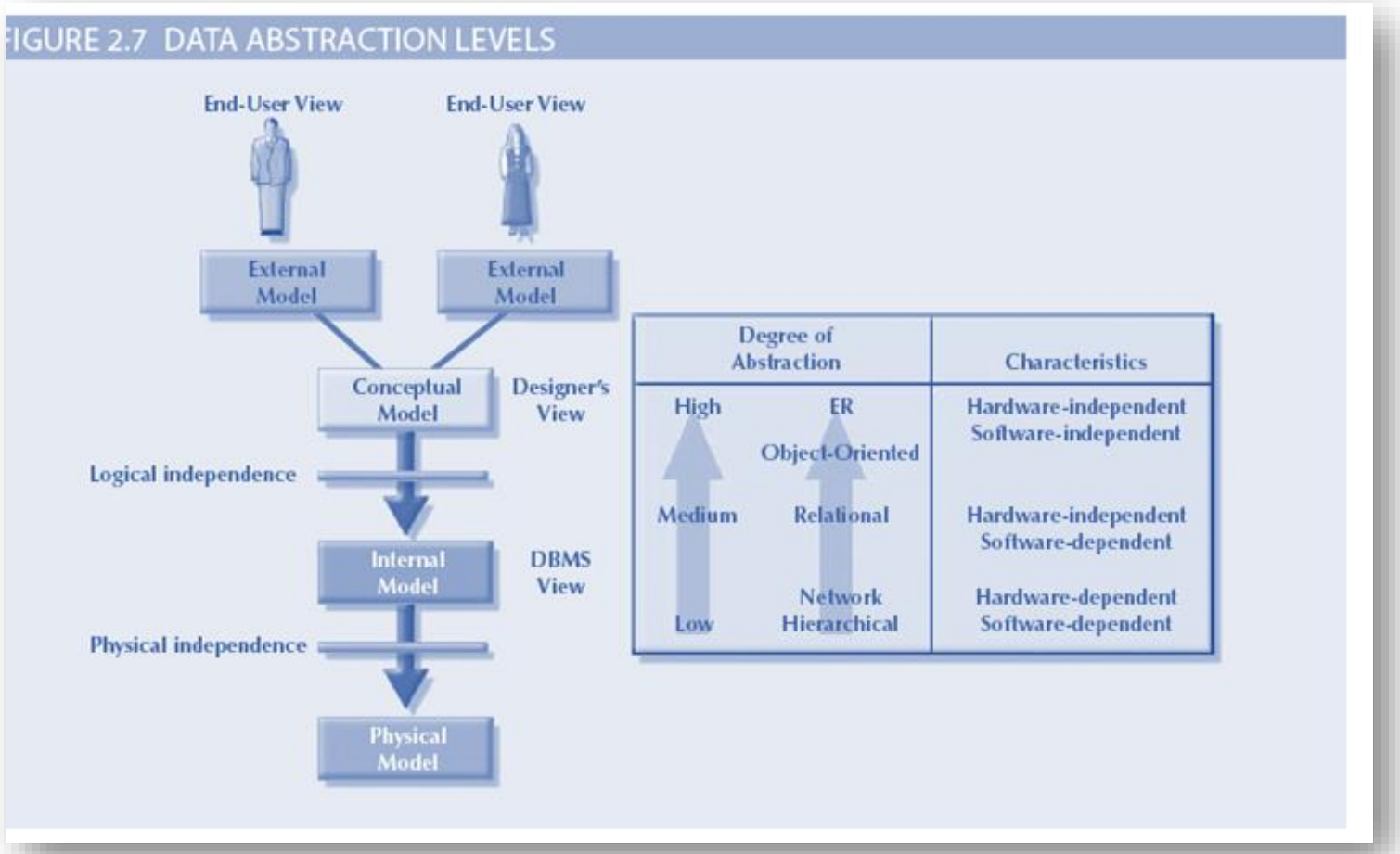
Veri Soyutlama

- Veri modellerinin daha iyi anlaşılabilmesini sağlamak amacıyla ANSI-SPARC, 1970'lerin başında, veri soyutlamanın 3 düzeyini tanımlamıştır. (ANSI-SPARC: American National Standards Institute, Standards Planning and Requirements Committee.)

➢ Harici Model (External Model)

➢ Kavramsal Model (Conceptual Model)

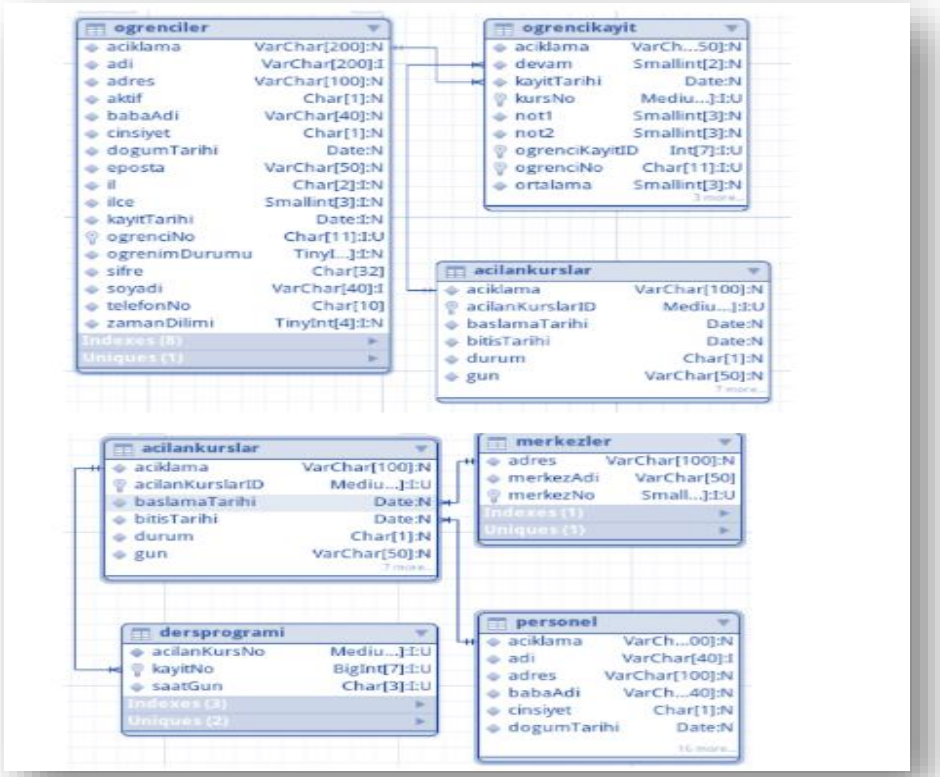
➢ Dahili Model (Internal Model)

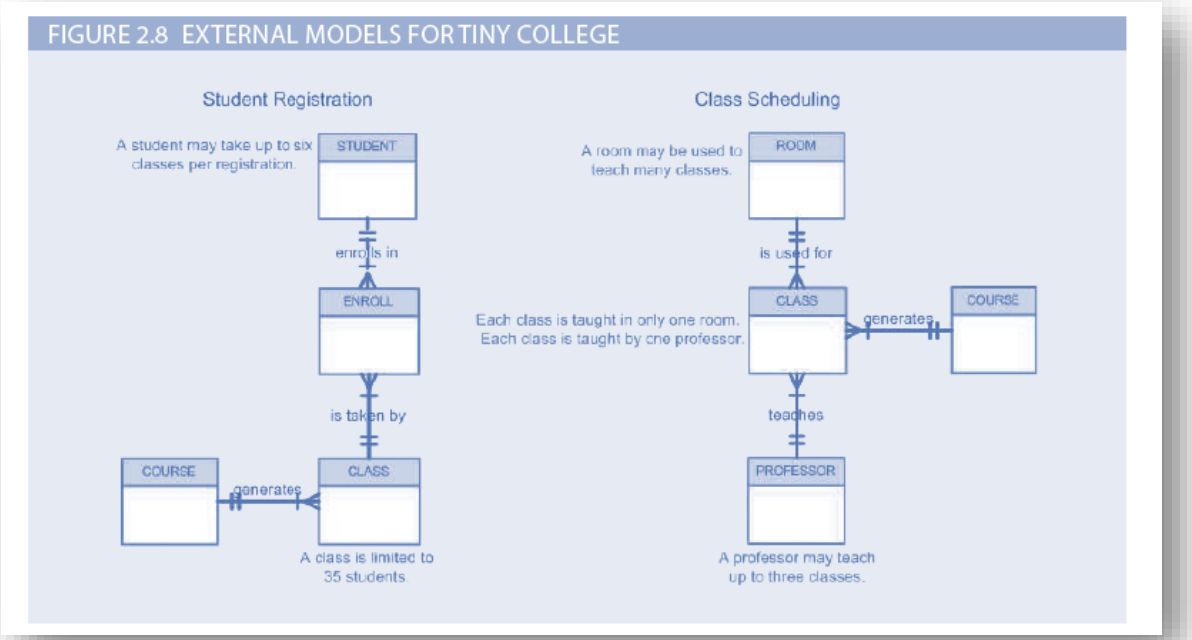


Veri Soyutlama (Data Abstraction)

Harici Model (External Model)

- Veritabanının son kullanıcılar açısından görünen kısmı.
- Veritabanının sadece kullanıcıyla ilgili alt bölümlerini ifade eder.
- Her harici şema, iş birimi tarafından gereken uygun varlıkları, ilişkileri, süreçleri ve kısıtlamaları içerir. Ayrıca, uygulama görünümleri birbirinden izole edilmesine rağmen, her görünümün diğer görünümle ortak bir varlığı paylaşır.

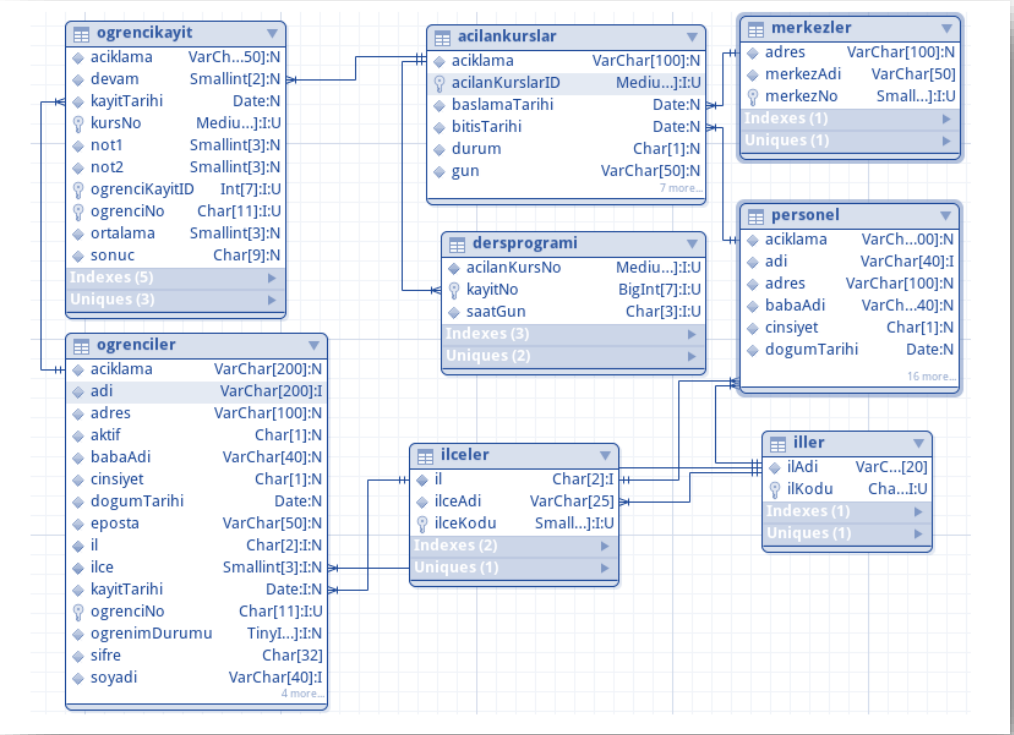


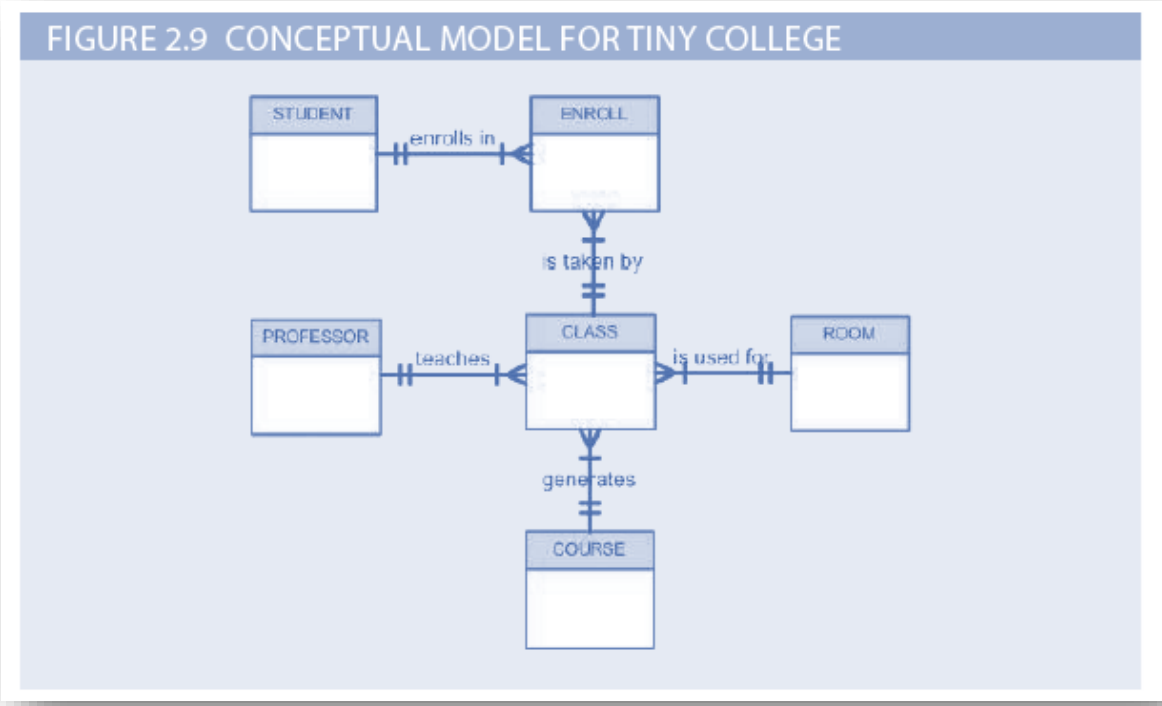


Veri Soyutlama (Data Abstraction)

Kavramsal Model (Conceptual Model

- Veritabanının veritabanı tasarımcısı açısından görünen kısmı.
- Veritabanının tüm alt bölümlerini birleştirerek global olarak görünmesini sağlar.
- Sıklıkla kullanılan kavramsal model ER Modelidir.
- Kullanılan yazılım (DBMS) ve donanımdan bağımsızdır. Donanım ya da yazılım değişikliği kavramsal model tasarımını etkilemez.
- Kavramsal model, tüm veritabanının global bir görünümünü sağlar ve ayrıntılardan kaçınarak ana veri nesnelerini açıklar.
- Kavramsal model mantıksal görünüş olarak da kullanılır.





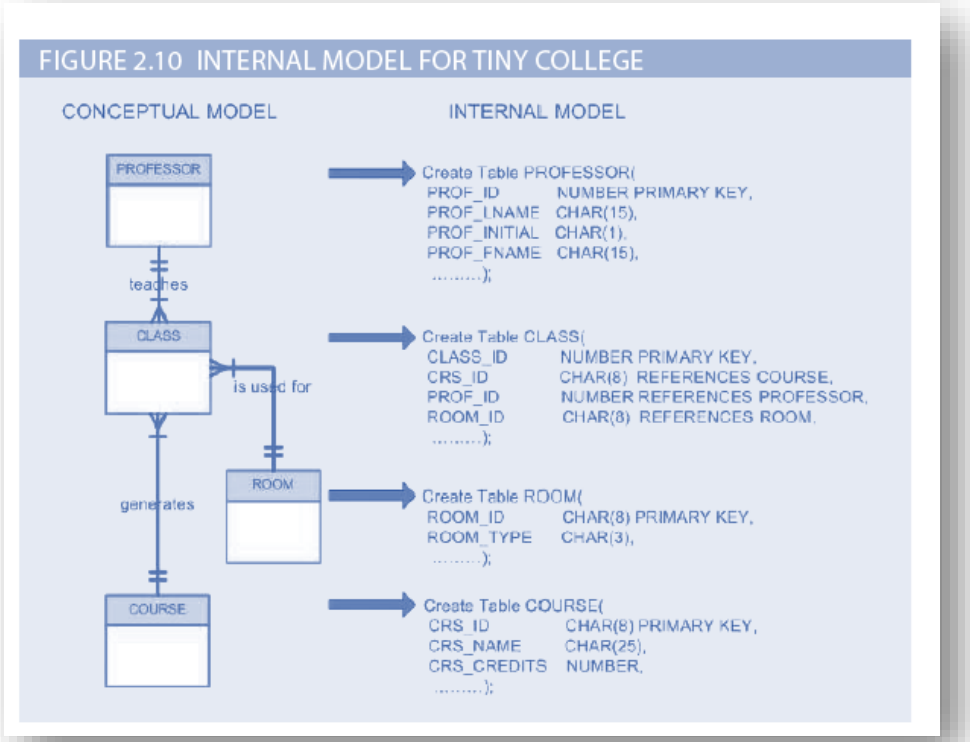
Veri Soyutlama (Data Abstraction)

Dahili Model (Internal Model)

- Kavramsal Modelin seçilen VTYS'ye göre uyarlanışı
- Veritabanının, Veritabanı Yönetim Sistemi tarafından görünen kısmı.
- Dahili model = ilişkisel model
- Donanım bağımsız, yazılım bağımlı.

```
-- CREATE TABLE "dersprogrami" ------------------------------
CREATE TABLE `dersprogrami` (
        `kayitNo` BigInt( 7 ) UNSIGNED ZEROFILL AUTO_INCREMENT NOT NULL,
        `acilanKursNo` MediumInt( 6 ) UNSIGNED ZEROFILL NOT NULL UNIQUE,
        `saatGun` Char( 3 ) NOT NULL UNIQUE,
        PRIMARY KEY ( `kayitNo` ),
        CONSTRAINT `acilanKursNo` UNIQUE( `acilanKursNo`, `saatGun` ) )
ENGINE = InnoDB
AUTO_INCREMENT = 97;
-- ---------------------------------------------------------


-- CREATE TABLE "duyurular" ---------------------------------
CREATE TABLE `duyurular` (
        `duyuruNo` Int( 7 ) UNSIGNED AUTO_INCREMENT NOT NULL,
        `personelNo` Char( 11 ) NOT NULL,
        `konu` VarChar( 50 ) NOT NULL,
        `eklemeTarihi` Date NOT NULL,
        `sonTarihi` Date NOT NULL,
        `icerik` VarChar( 10000 ) NULL,
        PRIMARY KEY ( `duyuruNo` ) )
ENGINE = InnoDB
AUTO_INCREMENT = 6;
-- ---------------------------------------------------------
```



Veri Soyutlama (Data Abstraction)

Fiziksel Model (Physical Model)

- manyetik, katı hal veya optik ortam gibi depolama ortamlarında verilerin kaydedilme şeklini açıklar.
- verilere ulaşmak için hem fiziksel depolama cihazlarının hem de (fiziksel) erişim yöntemlerinin tanımını gerektirir
- Fiziksel model, VTYS'ye, dosyalara erişim yöntemlerine ve işletim sistemi tarafından desteklenen donanım depolama cihazlarının türlerine bağlıdır

TABLE 2.4

LEVELS OF DATA ABSTRACTION

MODEL	DEGREE OF ABSTRACTION	FOCUS	INDEPENDENT OF
External	High	End-user views	Hardware and software
Conceptual	↕	Global view of data (database model independent)	Hardware and software
Internal	↕	Specific database model	Hardware
Physical	Low	Storage and access methods	Neither hardware nor software

ÖZET

- …… karmaşık bir gerçek dünya veri ortamının bir soyutlamasıdır.
- Temel veri modelleme bileşenleri ……., ……, …… .. ve …….
- ……. belirli bir gerçek dünya ortamında temel modelleme bileşenlerini belirlemek ve tanımlamak için kullanılır.
- ……. ve …… .. veri modelleri artık kullanılmayan ilk veri modellerindendir, ancak bazı kavramları mevcut veri modellerinde bulunmaktadır.
- İlişkisel modelde tablolar birbirleriyle ortak niteliklerdeki ……. kullanılarak ilişkilendirilir.
- …… .. modeli, ilişkisel modeli tamamlayan popüler bir veri modelleme aracıdır. Veritabanı tasarımcılarının verilerin farklı görünümlerini - veritabanı tasarımcıları, programcılar ve son kullanıcılar tarafından görüldüğü gibi - görsel olarak sunmasına ve verileri ortak bir çerçeveye entegre etmesine olanak tanır.
- ……, ……. ve ……. gibi yeni ortaya çıkan Big Data teknolojileri, Büyük Veri analitiği için dağıtık, hata toleranslı ve uygun maliyetli destek sağlar.
- ……. veritabanları, ilişkisel modeli kullanmayan ve Büyük Veri kuruluşlarının çok özel ihtiyaçlarını desteklemek için tasarlanmış yeni nesil veritabanlarıdır.
- (ANSI/SPARC) üç seviye veri soyutlama tanımlar:. ……., ……, …… Dördüncü ve en düşük seviye veri soyutlaması olan …… ise yalnızca fiziksel depolama yöntemleriyle ilgilidir.

Referanslar

Carlos Coronel, Steven Morris, and Peter Rob, Database Systems: Design, Implementation, and Management, Cengage Learning.
https://github.com/celalceken/DatabaseManagementSystems
Douglas Laney, “3D data management controlling data volume, velocity and variety,” META Group, February 6, 2011.
IBM, “What is big data? Bringing big data to the enterprise,” http://www-01.ibm.com/software/data/ bigdata/, accessed April 2013.
https://tutorials.ducatindia.com/dbms/types-of-data-models/