VERİ MODELİ (2)

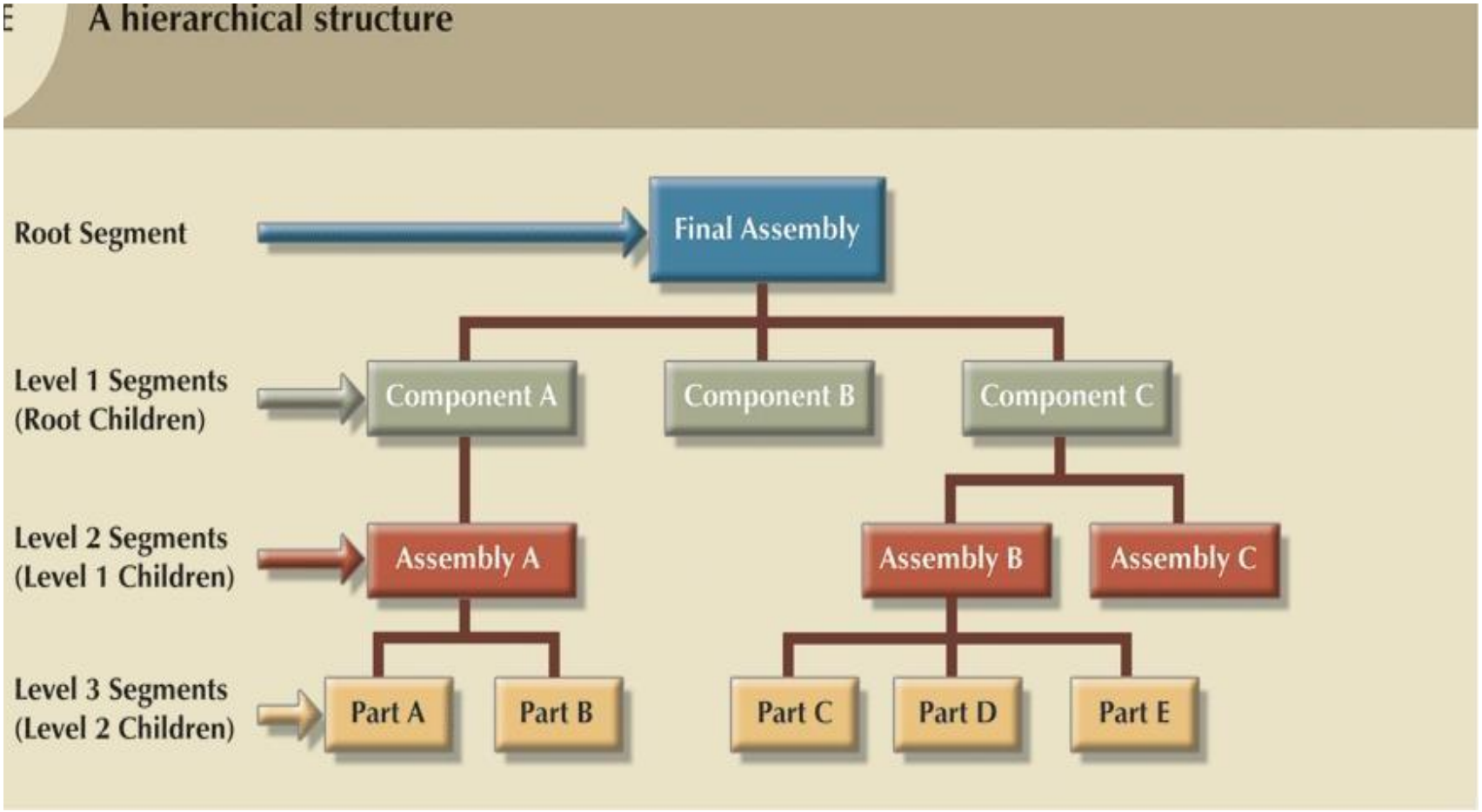
Veri Modellerinin Evrimi

GENERATION	TIME	DATA MODEL	EXAMPLES	COMMENTS
First	1960s–1970s	File system	VMS/VSAM	Used mainly on IBM mainframe systems Managed records, not relationships
Second	1970s	Hierarchical and network	IMS, ADABAS, IDS-II	Early database systems Navigational access
Third	Mid-1970s	Relational	DB2 Oracle MS SQL Server MySQL	Conceptual simplicity Entity relationship (ER) modeling and support for relational data modeling
Fourth	Mid-1980s	Object-oriented Object/relational (O/R)	Versant Objectivity/DB DB2 UDB Oracle 12c	Object/relational supports object data types Star Schema support for data warehousing Web databases become common
Fifth	Mid-1990s	XML Hybrid DBMS	dbXML Tamino DB2 UDB Oracle 12c MS SQL Server	Unstructured data support O/R model supports XML documents Hybrid DBMS adds object front end to relational databases Support large databases (terabyte size)
Emerging Models: NoSQL	Early 2000s to present	Key-value store Column store	SimpleDB (Amazon) BigTable (Google) Cassandra (Apache) MongoDB Riak	Distributed, highly scalable High performance, fault tolerant Very large storage (petabytes) Suited for sparse data Proprietary application programming interface (API)

Hiyerarşik Model

- **Yukarıdan aşağıya** veya **üst-alt** yapı modeli olarak adlandırılır.
- 1960'larda üretim projeleri için *büyük miktarda veriyi yönetmek* için geliştirilmiştir
- Temel mantıksal yapı baş-aşağı bir "ağaç" (Ağaç benzeri yapı) ile temsil edilir.
- Hiyerarşik yapı düzeyler veya segmentler (bölümler) içerir.
  - Kayıt türüne benzer bölümler vardır.
  - Her belirli bölüm arasındaki bire çok (1:M) ilişkileri kümesine sahiptir.

Hiyerarşik Model Örneği



Hiyerarşik Model

- Arama işlemi, **yukarıdan aşağıya** başlatılıp sırasıyla *sol-sağ biçimi* şeklinde devam eder.
- **Güncel veri modelleri için temel oluşturur.**
- **Bu model için anlaması kolaydır.**
- **Hiyerarşik modelin dezavantajları:**
  - Uygulaması Karmaşıktır.
  - Yönetilmesi zordur.
  - Yapısal bağımsızlıkta eksiklik vardır.
  - İlişkiler **M:N** formuna uymamaktadır.
  - Nasıl uygulanılacağına ilişkin standart yok

Ağ (Nettwork) Modeli

- Karmaşık veri ilişkilerini daha etkili temsil etmek için oluşturulmuştur.
  - Veritabanı performansı iyileştirilmiştir.
  - Veritabanı standardı zorlamaktadır.
- Hiyerarşik modele benzemektedir.
  - Kaydın birden fazla üst öğesi olabilir.

Ağ (Network) Modeli

- **1:M ilişkilerinde kayıtların toplamaktadır.**
- **İki kayıt türünden oluşan kümeye sahiptir.**
- **Sahip (Owner)**
  - Hiyerarşik modelin *üst öğesine* eşdeğerdir.
- **Üye (Member)**
  - Hiyerarşik modelin *alt çocuğuna* eşdeğerdir.

Ağ Modeli (2)

- Ağ veritabanı modeli genellikle bugün kullanılmaz.

  Ancak, bu model ile ortaya çıkan *standart veritabanı kavramlarının tanımları,* hala modern veri modelleri tarafından kullanılmaktadır:

- **Şema**

  Veritabanı yöneticisi tarafından görüntülenen tüm veritabanının kavramsal organizasyondur.

  Tablolar, **dizinler/indeksler**, **görünümler** ve **sorgular**

Ağ Modeli (3)

- **Alt Şema**

  Uygulama programları tarafından "görülen" veritabanı bölümüdür.

- **Veri yönetimi dili (DML)**

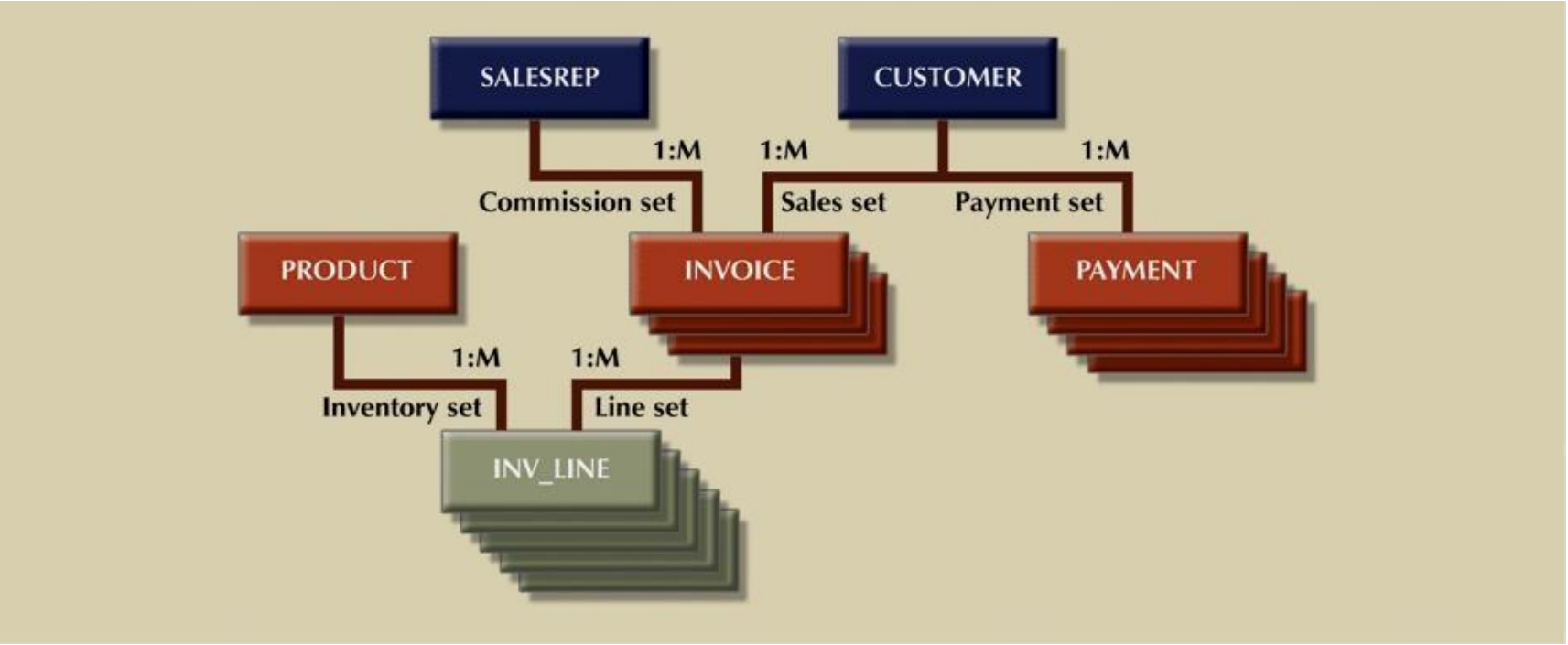
Verilerin yönetilebileceği ortamı tanımlar.

SELECT, INSERT, UPDATE, DELETE, COMMIT, and ROLLBACK.

- **Veri tanım dili (DDL):**

Veritabanı yöneticisinin şema bileşenlerini tanımlamasını sağlar.

Ağ Modeli (4)


SALESREP
CUSTOMER
1:M
1:M
1:M
Commission set
Sales set
Payment set
PRODUCT
INVOICE
PAYMENT
1:M
1:M
Inventory set
Line set
INV_LINE

Ağ Modeli (5)

- **Ağ modelinin dezavantajları**
  - Hantal (Çok zor)
  - Raporlar için geçici sorgu özelliğinin olmaması ve bu nedenle programcılara yük bindirmesi
  - Veritabanındaki bir yapısal değişiklik tüm uygulama programlarında tahribatlar yaratabilir.

İlişkisel Model (1)

- Model kavramsal olarak basittir.
- 1970 yılında E. F. Codd (IBM) tarafından geliştirilmiştir.
- Tablo (ilişkiler)

Satır/sütun kesişimlerinden oluşan bir matristir.

*Bir ilişkideki her satır **tuple** olarak adlandırılır.*

İlişkisel Model (1)

- **İlişkisel veri yönetim sistemi (RDBMS)**

  Hiyerarşik model tarafından sağlanan işlevlerin aynısını gerçekleştirir.

  Kullanıcıdan karmaşıklığı gizlemektedir.

İlişkisel Model (2)

SQL tabanlı ilişkisel veritabanı uygulaması üç parçadan oluşur.

I. Kullanıcı arayüzü
   - **Son kullanıcının verilerle etkileşim kurmasına izin vermektedir.**

II. Veritabanında depolanan tablo kümesi
   - **Her tablo diğerinden bağımsızdır.**
   - **Satırlar, farklı tablolardaki ortak özniteliklerdeki değerleri temel alır.**

III. SQL "motoru"
   - **Tüm sorguları çalıştırır.**

İlişkisel Model (3)

Table name: AGENT (first six attributes)

Database name: Ch02_InsureCo

AGENT_CODE	AGENT_LNAME	AGENT_FNAME	AGENT_INITIAL	AGENT_AREACODE	AGENT_PHONE
501	Alby	Alex	B	713	228-1249
502	Hahn	Leah	F	615	882-1244
503	Okon	John	T	615	123-5589

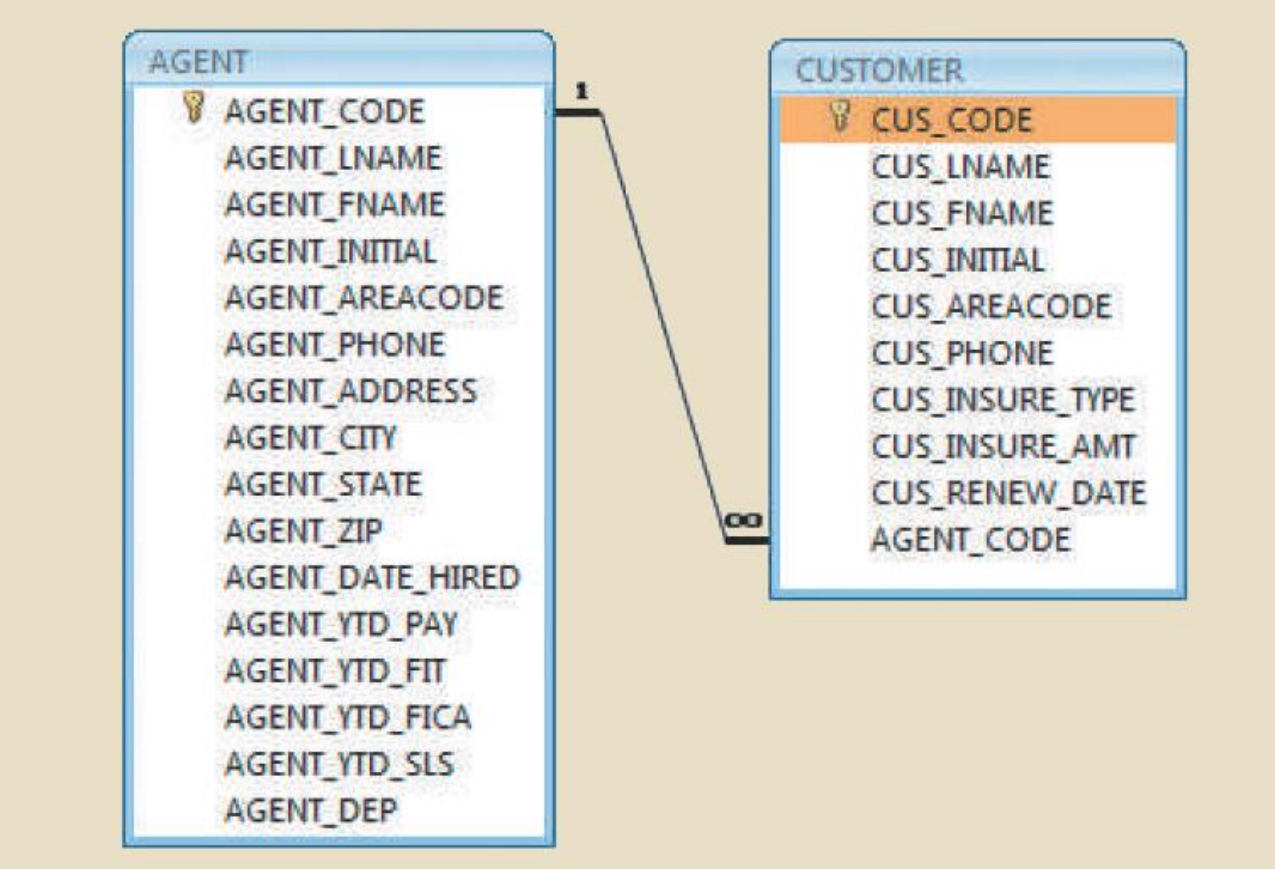
Link through AGENT_CODE

Table name: CUSTOMER

CUS_CODE	CUS_LNAME	CUS_FNAME	CUS_INITIAL	CUS_AREACODE	CUS_PHONE	CUS_INSURE_TYPE	CUS_INSURE_AMT	CUS_RENEW_DATE	AGENT_CODE
10010	Ramas	Alfred	A	615	844-2573	T1	100.00	05-Apr-2018	502
10011	Dunne	Leona	K	713	894-1238	T1	250.00	16-Jun-2018	501
10012	Smith	Kathy	W	615	894-2285	S2	150.00	29-Jan-2019	502
10013	Olowski	Paul	F	615	894-2180	S1	300.00	14-Oct-2018	502
10014	Orlando	Myron		615	222-1672	T1	100.00	28-Dec-2019	501
10015	O'Brian	Amy	B	713	442-3381	T2	850.00	22-Sep-2018	503
10016	Brown	James	G	615	297-1228	S1	120.00	25-Mar-2019	502
10017	Williams	George		615	290-2556	S1	250.00	17-Jul-2018	503
10018	Farriss	Anne	G	713	382-7185	T2	100.00	03-Dec-2018	501
10019	Smith	Olette	K	615	297-3809	S2	500.00	14-Mar-2019	503

İlişkisel Model (4)

RELATIONAL DIAGRAM



Yukarıdaki şekil, Acces veritabanından alınmıştır. (1: M ilişki)

Varlık İlişki modeli (6)

- **Varlık** ilişkisel bir tabloyla eşlenmektedir.
- Varlık örneği (veya oluşumu) tablodaki **satırdır.**
- **Varlık kümesi,** benzeri varlıkların koleksiyonudur.
- **Bağlantı ilişki türlerini etiketlemektedir.**
- **Entity relationship diagram (ERD)**
  - *Veritabanı bileşenlerini:* varlıklar (entities), öznitelikler (attributes) ve ilişkiler (relationships),
  - Bu veritatabanı bileşenleri modellemek için grafik gösterimlerini kullanılır.

Varlık İlişki modeli (7)

Üç tip ER gösterimleri kullanılır:

1. Chen notasyonu

2. Kazayağı notasyonu

3. Birleşik Modelleme Dili (UML)

Varlık İlişki modeli (3)

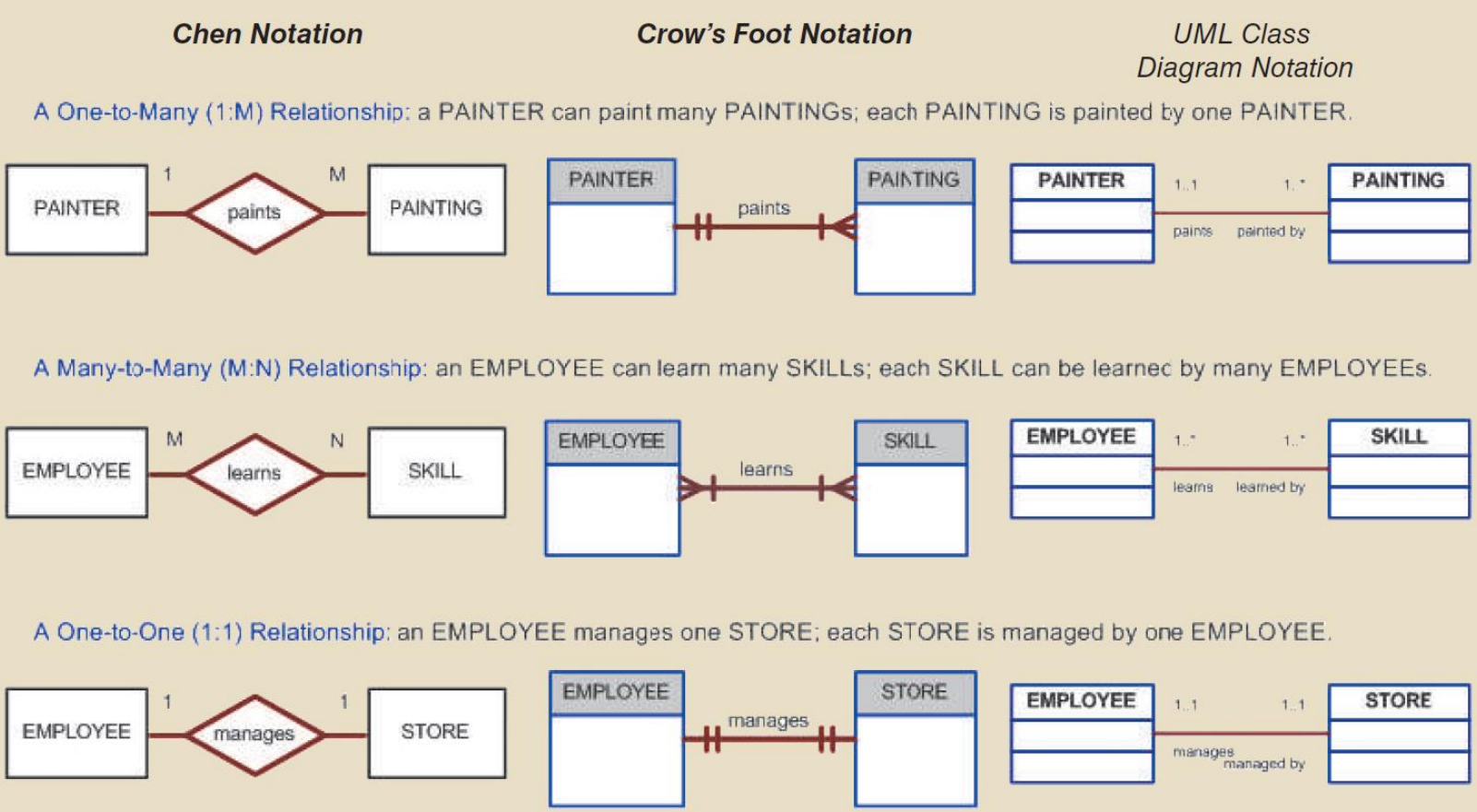
Chen notasyonu

Varlık, *dikdörtgenle temsil edilir.*

İlişkiler*, elmas/bakla dilimi ile temsil edilir.*

İlişki adı*, elmas/bakla dilimi içerisine yazılır.*

The ER Model Notations



Nesne Yönelimli (OO) Model

- Hem **veriler** hem de **ilişkileri nesne** olarak bilinen tek bir yapıda bulunur.
- OODM (nesne yönelimli veri modeli) geliştirme, bir nesnenin tüm işlemleri de içermesine izin verdi.
  - **İşlemler:** Veri değerlerini değiştirme, belirli bir veri değerini bulma ve veri değerlerini yazdırma

The Object-Oriented (OO) Model

- **OODBMSM:** Nesne yönelimli bir veritabanı modelinde verileri yönetmek için kullanılan yazılımdır.
- **OODM'nin anlamsal** bir veri modeli olarak ta görülmektedir.

  Anlamsal Veri Modeli: Nesne olarak bilinen tek bir yapıda hem veriler hem de ilişkilerdir.

Nesne Yönelimli (OO) Model

Nesneye Yönelimli veri modeli, aşağıdaki bileşenleri temel alır:

- **Nesne,** gerçek dünyadaki bir varlığın soyutlanmasıdır.
- **Öznitelikler**, bir nesnenin özelliklerini açıklar. Örneğin, bir PERSON nesnesi; *Ad*, *Sosyal Güvenlik Numarası* ve *Doğum Tarihi* özniteliklerini içerir.

Nesne Yönelimli (OO) Model

Sınıf, *paylaşılan yapıya (özniteliklere) ve davranışa (yöntemlere) sahip benzer nesnelerin bir koleksiyonudur.*

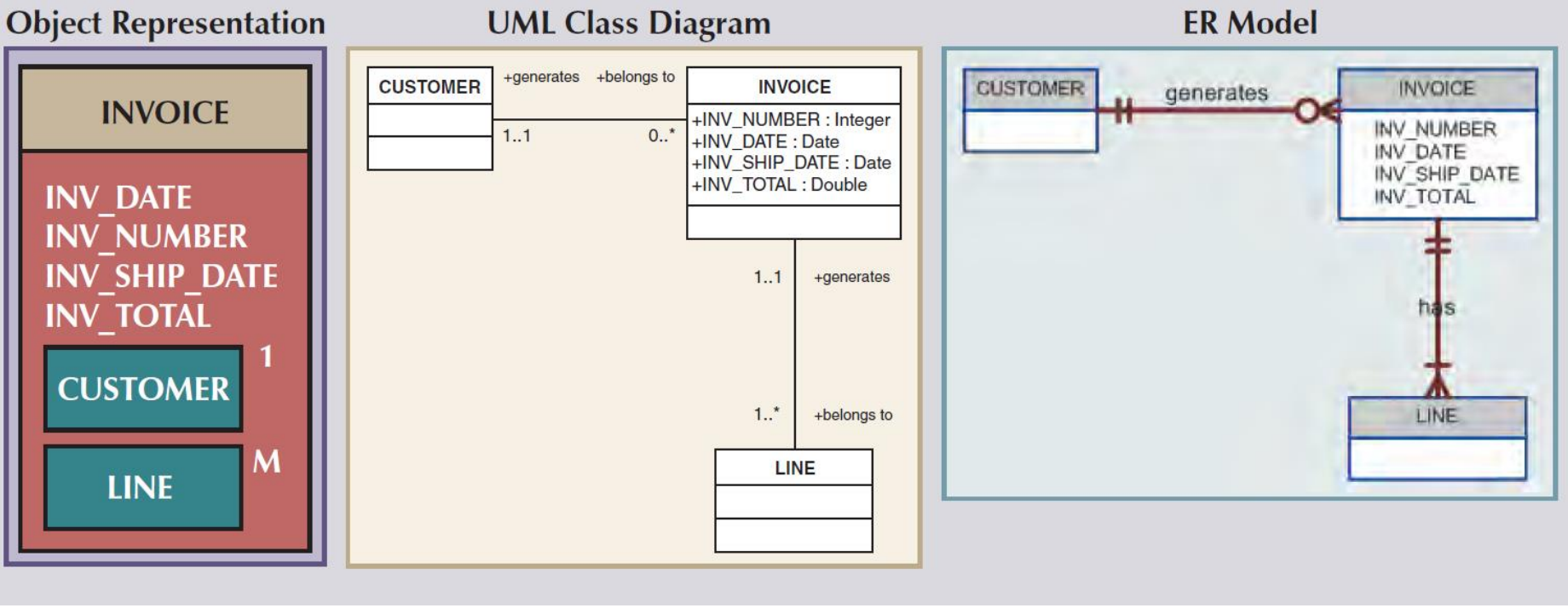
- Sınıf, varlık kümesinden farklıdır.
- *Yöntemler* olarak bilinen **bir dizi yordam içerir.**

Nesne Yönelimli (OO) Model

- **Sınıflar bir sınıf hiyerarşisinde düzenlenir.**
  - *Örneğin, CUSTOMER sınıfı ve EMPLOYEE sınıfı, üst PERSON sınıfından kalıtım devralır.*
- Nesne yönelimli veri modelleri genellikle **Birleşik Modelleme Dili (UML)** sınıf diyagramları kullanılarak gösterilir.

Nesne Yönelimli (OO) Model

FIGURE 2.4 A comparison of OO, UML, and ER models



Nesne/İlişkisel ve XML

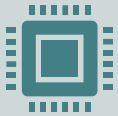
Genişletilmiş ilişkisel veri modeli (ERDM)

- Uygulamaların karmaşıklığının artmasına **yanıt olarak geliştirilen anlamsal veri modelidir.**
- Nesne yönelimli modelinin en iyi özelliklerinin çoğunu içerir.
- Genellikle nesne/ilişkisel veritabanı yönetim sistemi (O/RDBMS) olarak tanımlanır.
- Öncelikle iş uygulamalarına yöneliktir.

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (1)

Büyük Veri (Big Data):

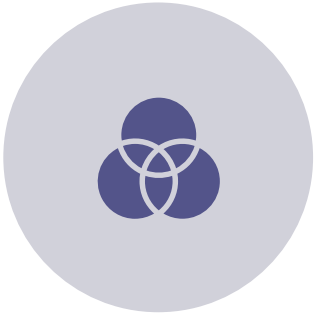
Büyük miktarda **web tarafından** oluşturulan verileri yönetmek ile ilgilidir.



Benzer şekilde **sensör tarafından** oluşturulan verileri yönetmek ile ilgilidir.

Ayrıca işe **yönelik bir içgörü elde etmenin yeni** ve **daha iyi yollarını bulmak** için bir hareketi ifade eder.

BIG DATA (Büyük Veri)



BÜYÜKLÜK/BOYUT (VOLUME):

DEPOLANAN VERI MIKTARLARIDIR.

HIZ (VELOCİTY):

VERILERIN BÜYÜME HIZIDIR.

AYNI ZAMANDA BİLGİ VE İÇGÖRÜ OLUŞTURMAK İÇİN BU VERİLERİ HIZLI BİR ŞEKİLDE İŞLEME İHTİYACINI DA İFADE EDER.

ÇEŞITLILIK (VARIETY):

TOPLANAN VERİLERİN BİRDEN ÇOK FARKLI VERİ BİÇİMİNDE GELDİĞİ GERÇEĞİNİ IFADE EDER.

3V ÇERÇEVESI, VERITABANLARININ BOYUT VE KARMAŞIKLIK OLARAK KATLANARAK BÜYÜDÜĞÜNÜ GÖSTERMEKTEDIR.

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (2)

- **Sorun,** *ilişkisel yaklaşımın* her zaman *Büyük Veri* zorlukları olan kuruluşların ihtiyaçlarıyla eşleşmemesidir.
- Yapılandırılmamış, ***sosyal medya*** ve *sensör tarafından oluşturulan verilerin* satır ve sütunların geleneksel ilişkisel yapısına sığdırılmış olması **her zaman mümkün değildir.**
- **Daha fazla depolama**, **işlem gücü** ve *karmaşık veri çözümleme araçlarına* duyulan ihtiyacı, **ilişkisel veri modeli karşılayamamaktadır.**

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (3)

En sık kullanılan Büyük Veri teknolojilerinden bazıları :

- Hadoop
- Hadoop Distributed File System (HDFS)
- MapReduce
- NoSQL databases.

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (4)

- **Hadoop, Java tabanlı, açık kaynaklı, yüksek hızlı, hataya dayanıklı dağıtılmış depolama** ve **hesaplama** çerçevesidir.
- **Hadoop Dağıtılmış Dosya Sistemi (HDFS), yüksek hızlarda büyük miktarda veriyi yönetmek** için **tasarlanmış, yüksek oranda dağıtılmış, hataya dayanıklı bir dosya depolama sistemidir.**

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (4)

- **MapReduce, hızlı veri analizi hizmetleri sağlayan açık kaynaklı bir uygulama programlama arabirimidir (API).**
- **NoSQL, yapılandırılmış ve yapılandırılmamış verileri, verimli bir şekilde depolayan büyük ölçekli dağıtılmış bir veritabanı sistemidir.**

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (5)

NoSQL Veritabanları: Geleneksel ilişkisel veritabanı modelini temel almayan yeni nesil veritabanı yönetim sistemleridir.

Örneğin,

- Amazon'da bir ürün arar,
- Facebook'ta arkadaşlarınıza mesaj gönderir, YouTube'da video izler,
- Google Haritalar'da yol tarifi ararsanız,

NoSQL veritabanı kullanıyorsunuzdur.

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (2)

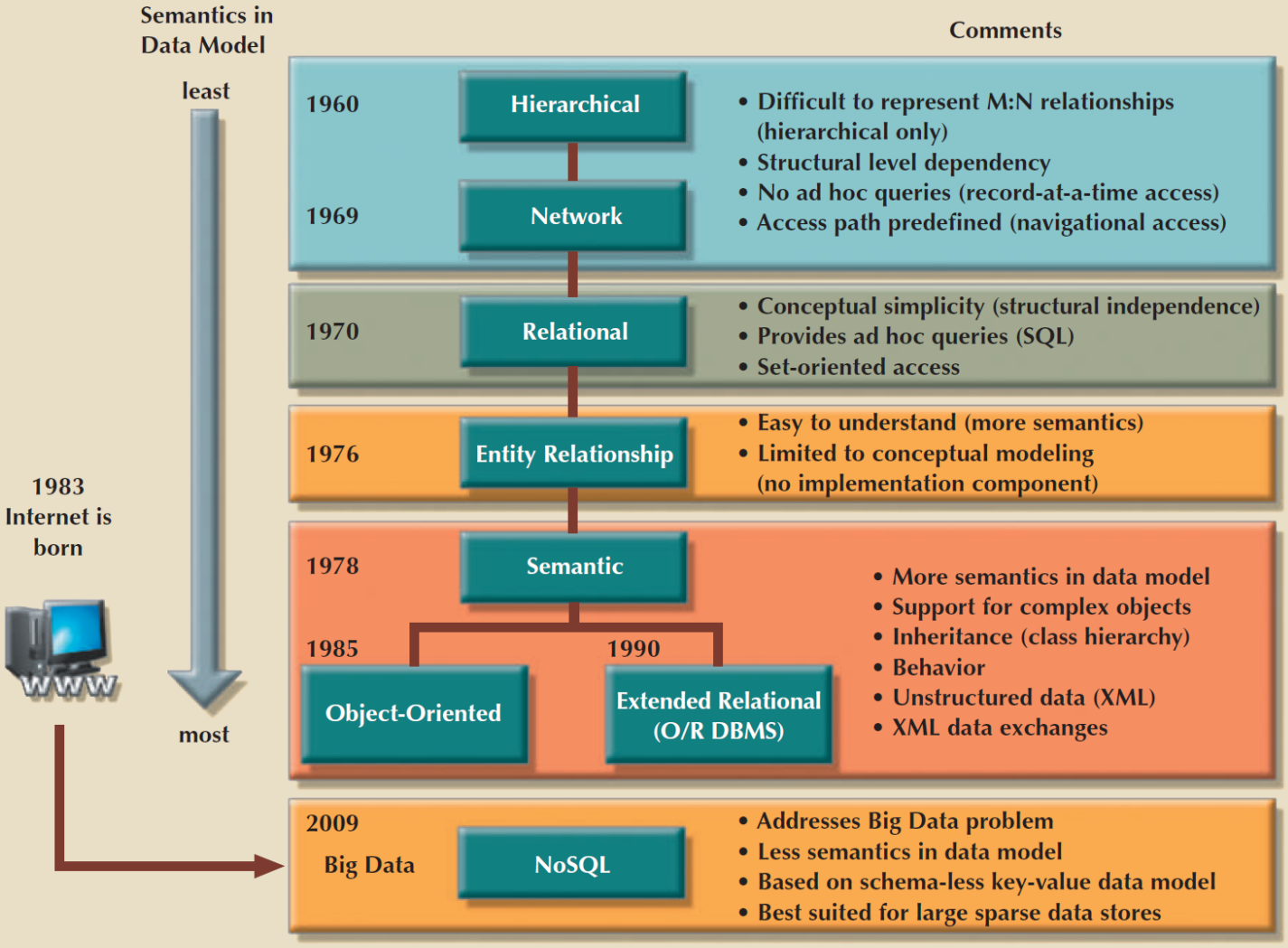
NoSQL, Büyük veri döneminin belirli zorluklarını ele almakta ve aşağıdaki genel özelliklere sahiptir:

- NoSQL ilişkisel modeli ve SQL'i temel almaz.
- Yüksek oranda dağıtılmış veritabanı mimarilerini destekler.
- Çok büyük miktarlarda seyrek verileri destekler.
  - çok sayıda öznitelik
  - gerçek veri örneği sayısı düşük

Ortaya Çıkan Veri Modelleri: Büyük Veri ve NoSQL (2)

- Yüksek ölçeklenebilirlik, yüksek kullanılabilirlik ve hataya dayanıklılık sağlanmalıdır.
- Hareket tutarlılığı yerine performansı göz önünde bulundurulmalıdır.

Veri Modellerinin Evrimi: Bir Özet



Çeşitli Veritabanı Modellerinin Avantajları ve Dezavantajları

ADVANTAGES AND DISADVANTAGES OF VARIOUS DATABASE MODELS

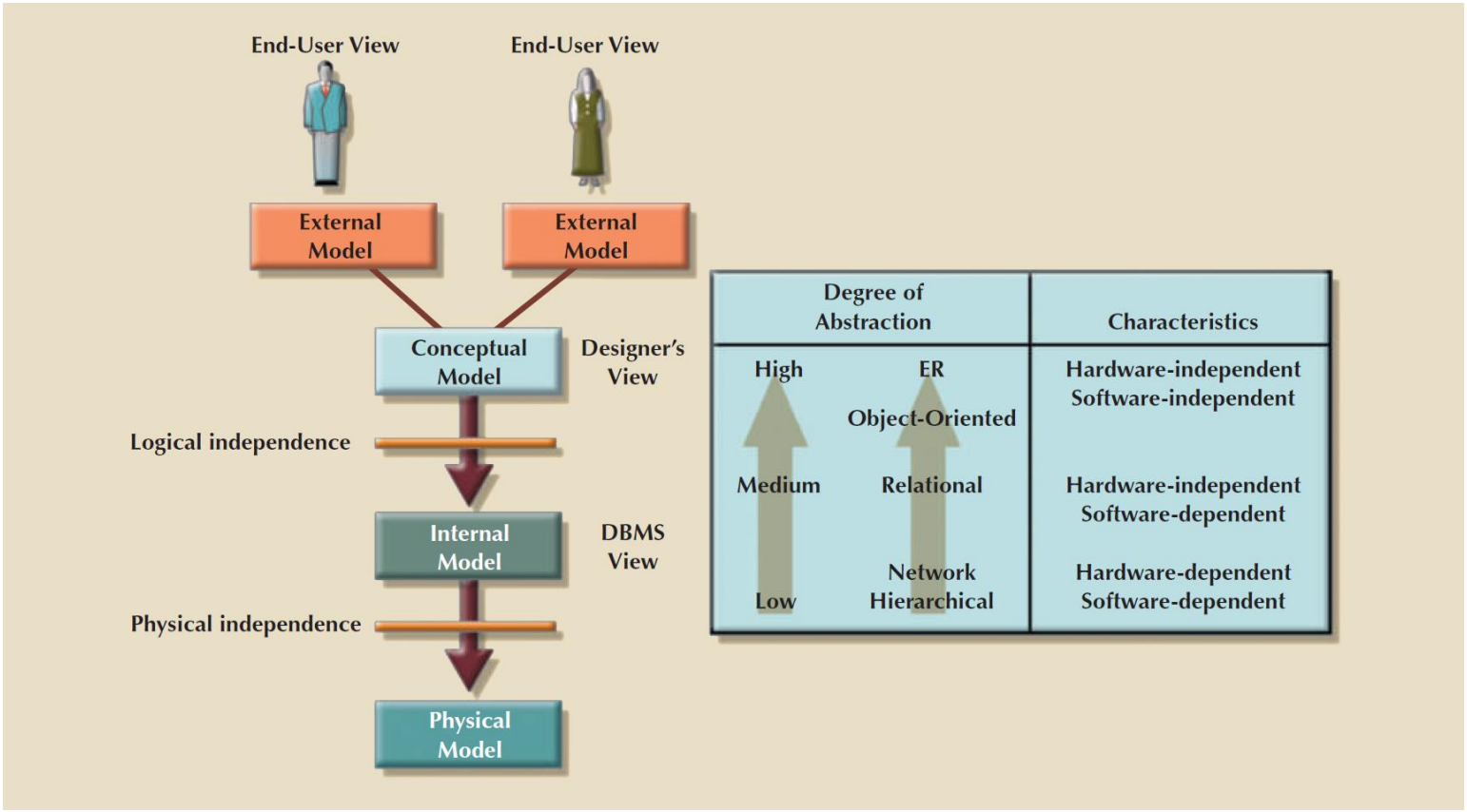
DATA MODEL	DATA INDEPENDENCE	STRUCTURAL INDEPENDENCE	ADVANTAGES	DISADVANTAGES
Hierarchical	Yes	No	1. It promotes data sharing. 2. Parent/child relationship promotes conceptual simplicity. 3. Database security is provided and enforced by DBMS. 4. Parent/child relationship promotes data integrity. 5. It is efficient with 1:M relationships.	1. Complex implementation requires knowledge of physical data storage characteristics. 2. Navigational system yields complex application development, management, and use; requires knowledge of hierarchical path. 3. Changes in structure require changes in all application programs. 4. There are implementation limitations (no multiparent or M:N relationships). 5. There is no data definition or data manipulation language in the DBMS. 6. There is a lack of standards.
Network	Yes	No	1. Conceptual simplicity is at least equal to that of the hierarchical model. 2. It handles more relationship types, such as M:N and multiparent. 3. Data access is more flexible than in hierarchical and file system models. 4. Data owner/member relationship promotes data integrity. 5. There is conformance to standards. 6. It includes data definition language (DDL) and data manipulation language (DML) in DBMS.	1. System complexity limits efficiency—still a navigational system. 2. Navigational system yields complex implementation, application development, and management. 3. Structural changes require changes in all application programs.
Relational	Yes	Yes	1. Structural independence is promoted by the use of independent tables. Changes in a table's structure do not affect data access or application programs. 2. Tabular view substantially improves conceptual simplicity, thereby promoting easier database design, implementation, management, and use. 3. Ad hoc query capability is based on SQL. 4. Powerful RDBMS isolates the end user from physical-level details and improves implementation and management simplicity.	1. The RDBMS requires substantial hardware and system software overhead. 2. Conceptual simplicity gives relatively untrained people the tools to use a good system poorly, and if unchecked, it may produce the same data anomalies found in file systems. 3. It may promote islands of information problems as individuals and departments can easily develop their own applications.
Entity relationship	Yes	Yes	1. Visual modeling yields exceptional conceptual simplicity. 2. Visual representation makes it an effective communication tool. 3. It is integrated with the dominant relational model.	1. There is limited constraint representation. 2. There is limited relationship representation. 3. There is no data manipulation language. 4. Loss of information content occurs when attributes are removed from entities to avoid crowded displays. (This limitation has been addressed in subsequent graphical versions.)
Object-oriented	Yes	Yes	1. Semantic content is added. 2. Visual representation includes semantic content. 3. Inheritance promotes data integrity.	1. Slow development of standards caused vendors to supply their own enhancements, thus eliminating a widely accepted standard. 2. It is a complex navigational system. 3. There is a steep learning curve. 4. High system overhead slows transactions.
NoSQL	Yes	Yes	1. High scalability, availability, and fault tolerance are provided. 2. It uses low-cost commodity hardware. 3. It supports Big Data. 4. Key-value model improves storage efficiency.	1. Complex programming is required. 2. There is no relationship support—only by application code. 3. There is no transaction integrity support. 4. In terms of data consistency, it provides an eventually consistent model.

Veri Modeli Temel Terminoloji Karşılaştırması

DATA MODEL BASIC TERMINOLOGY COMPARISON							
REAL WORLD	EXAMPLE	FILE PROCESSING	HIERARCHICAL MODEL	NETWORK MODEL	RELATIONAL MODEL	ER MODEL	OO MODEL
A group of vendors	Vendor file cabinet	File	Segment type	Record type	Table	Entity set	Class
A single vendor	Global supplies	Record	Segment occurrence	Current record	Row (tuple)	Entity occurrence	Object instance
The contact name	Johnny Ventura	Field	Segment field	Record field	Table attribute	Entity attribute	Object attribute
The vendor identifier	G12987	Index	Sequence field	Record key	Key	Entity identifier	Object identifier

VERI SOYUTLAMA DERECELERI

Kullanılabilir bir veritabanı tasarlanması, aynı temel işlemi izler.



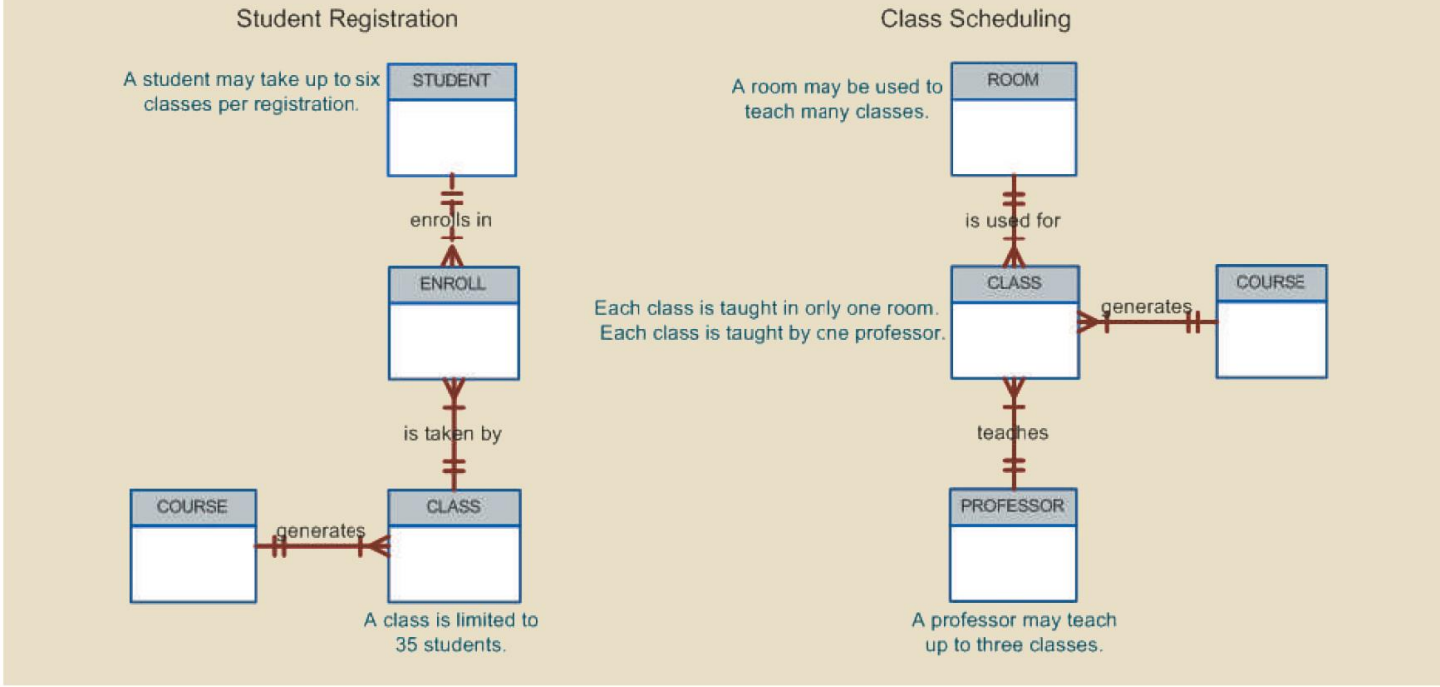
1. Dış Model

- Dış model, **son kullanıcıların veri ortamına bakışıdır.**
- **Son kullanıcılar terimi**, verileri işlemek ve bilgi oluşturmak için uygulama programlarını kullanan kişileri ifade eder.
- **Son kullanıcılar** genellikle bir uygulamanın belirli bir departman odağına sahip olduğu bir ortamda çalışır.

Dış Model (2)

- ER diyagramları dış görünümleri temsil etmek için kullanılacaktır.
- Dış görünümün (external view), belirli bir gösterimi **dış şema (external schema)** olarak bilinir.

FIGURE 2.7 EXTERNAL MODELS FOR TINY COLLEGE



Dış Model (2)

Veritabanının alt kümelerini temsil eden dış görünümlerin kullanımının bazı önemli avantajları vardır:

- Her departmanın operasyonlarını desteklemek için gereken **belirli verileri tanımlamak kolaydır.**
- Modelin yeterliliği hakkında geri bildirim sağlayarak **tasarımcının işini kolaylaştırır.**

Dış Model (2)

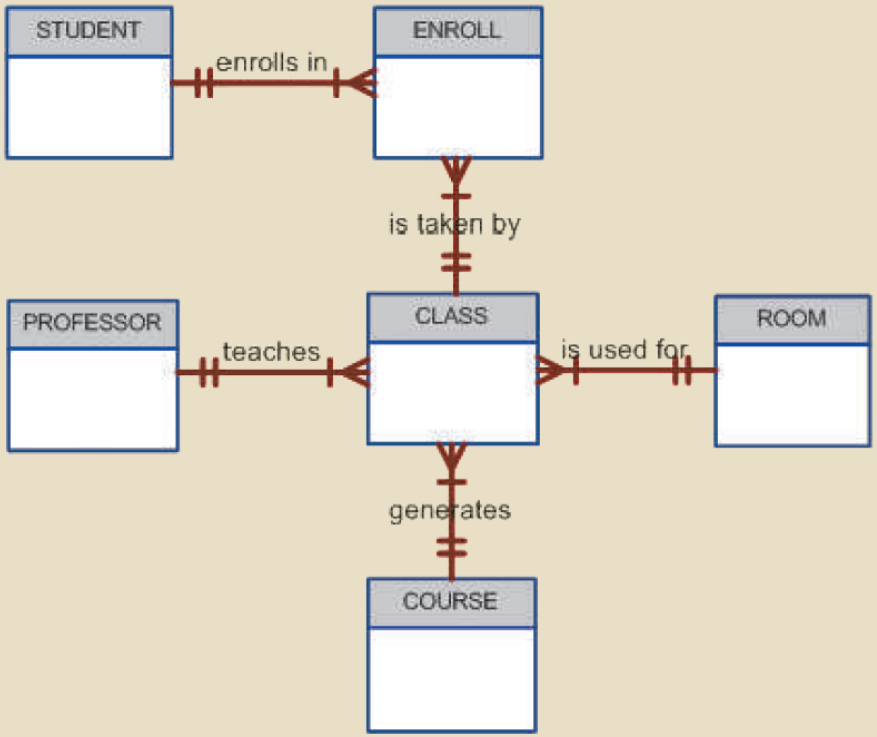
- Özellikle, modelin **tüm süreçleri** (operasyonel gereksinimler ve kısıtlamalar) desteklediğinden emin olmak için kontrol edilebilir.
- Veritabanı tasarımında güvenlik kısıtlamalarının sağlanmasına yardımcı olur.
- **Uygulama programı geliştirmeyi** çok daha basit hale getirir.

2. Kavramsal Model

- **Kavramsal model,** tüm kuruluş tarafından tüm **veritabanının genel görünümünü temsil eder.**
- **Kavramsal şema olarak da bilinir.**
- **Ana veri nesnelerinin tanımlanması** ve **üst düzey açıklaması için temel oluşturur.** (veritabanı modeline özgü ayrıntılardan kaçınmak).

Kavramsal Model (2)

FIGURE 2.8 A CONCEPTUAL MODEL FOR TINY COLLEGE



Kavramsal Model (3)

Kavramsal model bazı önemli avantajlar sağlar.

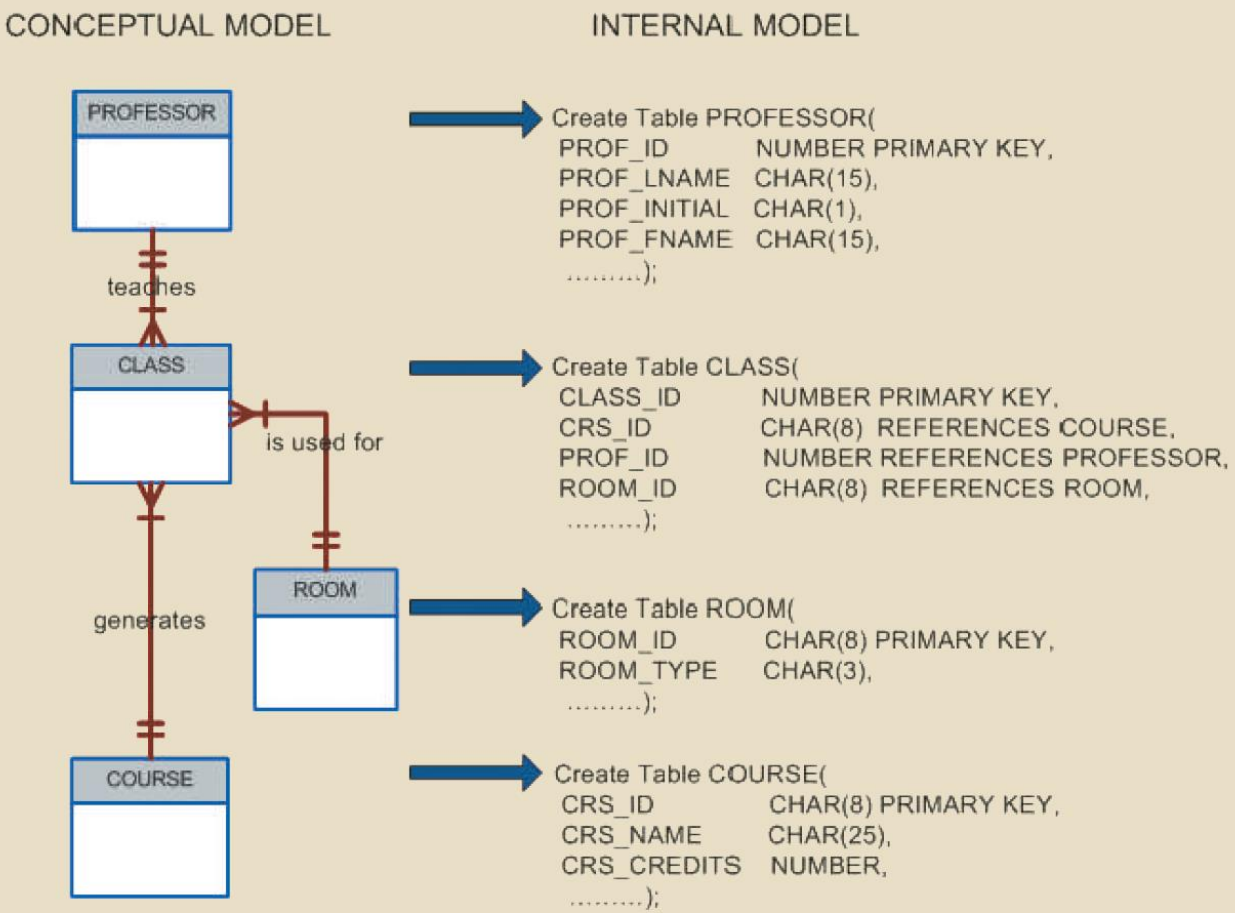
- Anlaşılması nispeten kolay olan veri ortamının kuş bakışı (makro düzeyi) görünümünü sağlar.
- Kavramsal model hem yazılımdan hem de donanımdan bağımsızdır.

3. Dahili (İç) model

- İç (internal) model, veritabanının seçilen DBMS tarafından "görüldü" olarak temsilidir.
- İç şema, seçilen veritabanı tarafından desteklenen veritabanı yapılarını kullanarak bir iç modelin belirli bir gösterimini tasvir eder.
- Mantıksal (logical) tasarım, **kavramsal tasarımı DB2**, **SQL Server**, **Oracle**, **IMS**, **Informix**, **Access** veya **Ingress** gibi seçili bir veritabanı yönetim sistemi için iç modele çevirmek için kullanılır.

Dahili (İç) Model (2)

FIGURE 2.9 INTERNAL MODEL FOR TINY COLLEGE



Dahili Model (3)

- İç model düzeyinde, hiyerarşik veya ağ modellerinin daha fazla detaya ihtiyaç duyar.
- Fakat ilişkisel model daha az ayrıntı gerektirir.
- Dahili model belirli veritabanı yazılımına bağlıdır, yazılıma bağımlı olduğu söylenir.

Fiziksel Model

- **Fiziksel model:** *Verilerin manyetik, katı hal* veya *optik ortam* gibi *depolama ortamlarında kaydedilme şeklini açıklar.*
- **İlk veri modelleri**, *veritabanı tasarımcısını fiziksel modelin veri depolama gereksinimlerinin ayrıntılarını almaya zorlamıştır.*
- **Artık baskın olan ilişkisel model,** fiziksel düzeyden ziyade büyük ölçüde mantıksal düzeyde hedeflenmektedir.

Veri Abstracti Düzeyleri: Özet

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

REFERENCES

- Carlos Coronel, Steven Morris, DATABASE SYSTEMS, Design, Implementation, and Management, Cengage Learning, 13. edition