

Veri Tabanı Yönetimi ve Modellemesi

HAFTA 2

Haftalık Ders Akışı

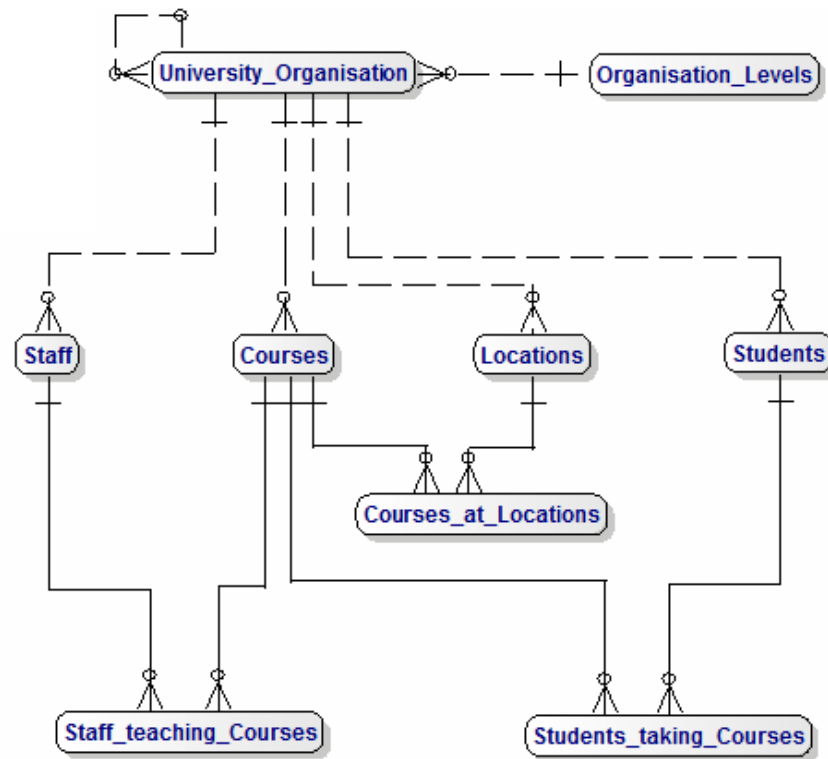
1. Veritabanı Kavramlarına Giriş
2. **Veri Tabanı Türleri, İlişkisel Veri Tabanı Tasarımı**
3. ER Diyagramları ve Normalizasyon
4. SQL Server Arayüzü, Veri Tabanı Nesneleri
5. T-SQL ve SQL Sorguları
6. İndeks ve View
7. Stored Procedure ve Fonksiyonlar
8. Ara Sınav
9. Tetikleyiciler
10. Transaction Kavramları ve Yedekleme
11. Kullanıcı Türleri ve Kullanıcı Yönetimi
12. No-SQL Veri Tabanları
13. No-SQL Veri Tabanları
14. Proje Sunumu
15. Proje Sunumları

Kaynak Kitaplar

- Raghu Ramakrishnan & Johannes Gehrke, Database Management Systems, 3rd Edition, 2003
- Jan L. Harrington, Relational Database Design and Implementation, 4th Edition, 2009
- Vijay Krishna Pallaw, Database Management Systems, 2nd Edition, 2013
- Thomas Connolly & Carolyn Begg, Database Systems A Practical Approach to Design, Implementation, and Management, 6th Edition, 2015
- Carlos Coronel & Steven Morris, Database Systems Design, Implementation, and Management, 12th Edition, 2016
- R. Elmasri & S.B.Navathe, Fundamentals of Database Systems, 7th Edition, 2016
- Louis Davidson & Jessica Moss, Pro SQL Server Relational Database Design and Implementation, 5th Edition, 2016

DBMS


- Gerçek dünya problemi
- Verinin Saklanması
- Veriler arasındaki bağlantılar
- İlişkisel Veri Modeli



Veri Modeli

- Gerçek dünya problemi
- Veri yapıları
- Grafiksel gösterimi

Veri Modeli Tasarlarken?

- Problem Ne
- Problemin yazınsal ve grafiksel gösterimi
- Genelden  Detaya
- Detayların şekilsel desteklenmesi



Veri Modeli Tasarlarken?

- Problemin yazınsal ve grafiksel gösterimi
- Veri yapılarının belirlenmesi (şemalar)
- Problemin içinde tanımlı kural yapıları
- Veri dönüşümleri için uygulanacak yöntem
- Nihai Tasarım: Deneme Yanılma



Problem Belirleme Adımları

- Firmanın kuralları
 - Yasal prosedürler
 - Standartlar
- İşin kuralları
 - İşin yapılmasında takip edilen iş planları
 - Hasta –ilaç: Hasta kaydı açılması
 - İşin müşteri/çalışan/diğer işler ile ilişkisi
 - Müşteri-Fatura
- İsimlendirme kuralları
 - Firmanın kullandığı özel terimler

Veri Modeli Yapı Taşları

○ Varlıklar

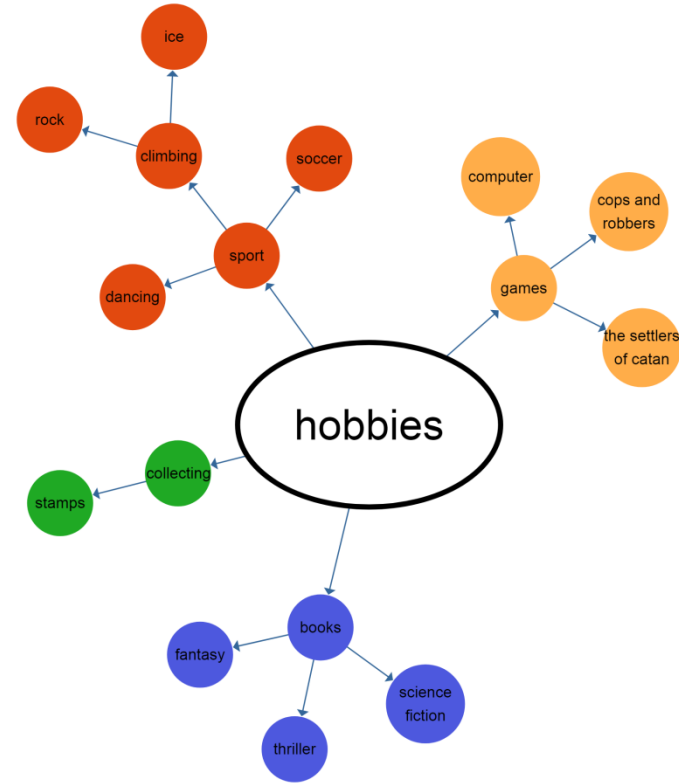
- Kişi
- Yer
- Eşya
- Olay

○ Nitelikler

- Kişi :ad,soyad
- Yer:Konum

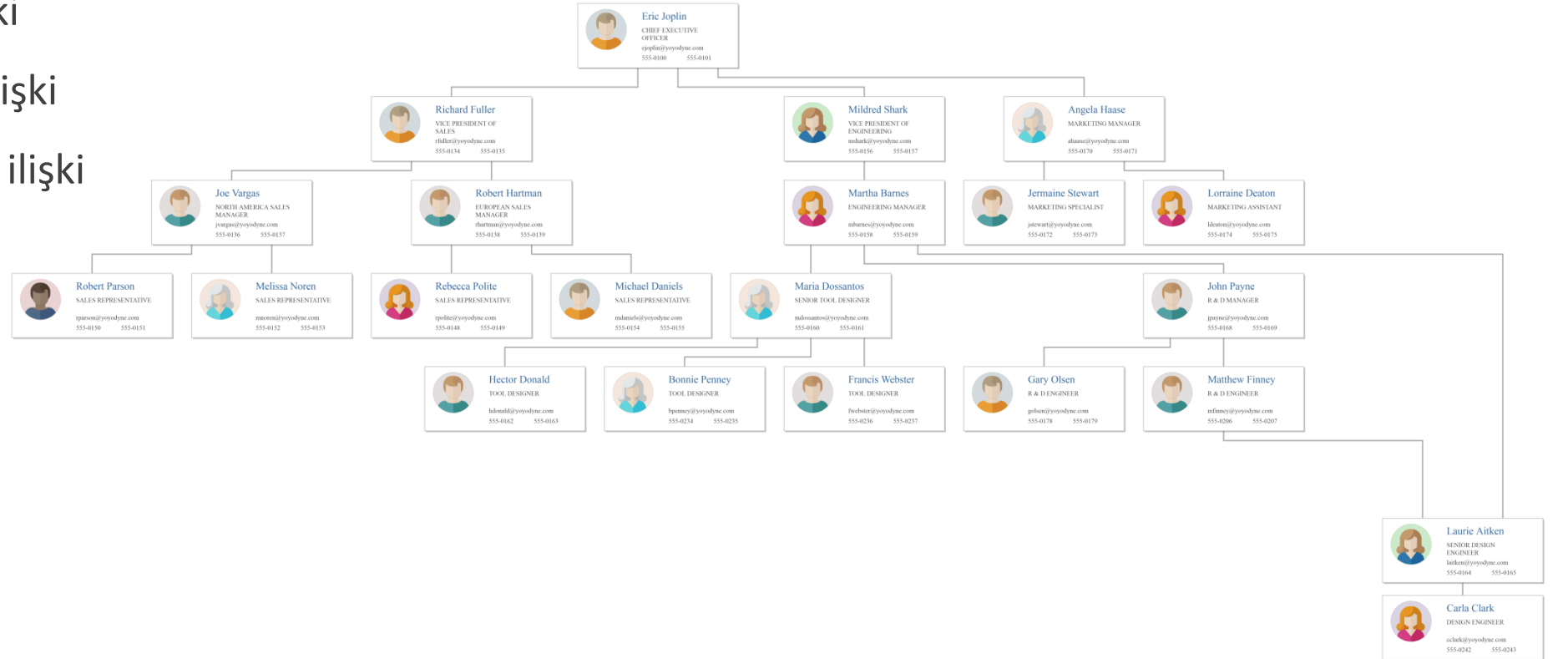
○ İlişkiler

○ Kısıtlamalar



İlişki Türleri

- Bire bir (1:1) ilişki
- Bire çok (1:M) ilişki
- Çoka Çok (N:M) ilişki



Veri Modelinin Önemi

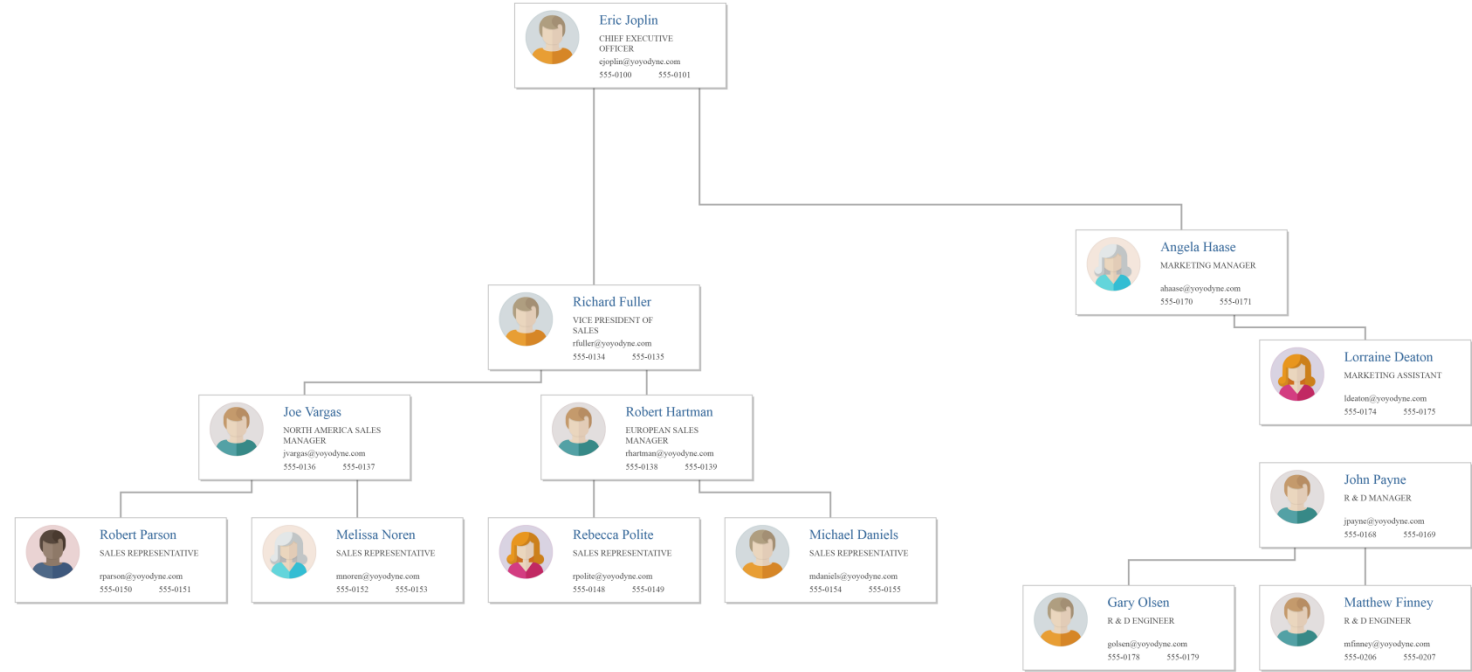
- Tasarımcı, uygulama geliştiricisi ve son kullanıcı
- Tasarlanan birime genel bakış
- Yetki Kontrolleri

Veri Modeli Türleri

- Dosya Sistemleri
- Hiyerarşik Model
- Ağ Modeli
- İlişkisel(Relational) Model
- Varlık-İlişki(Entity Relationship) Modeli
- Nesne Tabanlı (Object Oriented) Model
- Xml tabanlı model
- NoSQL

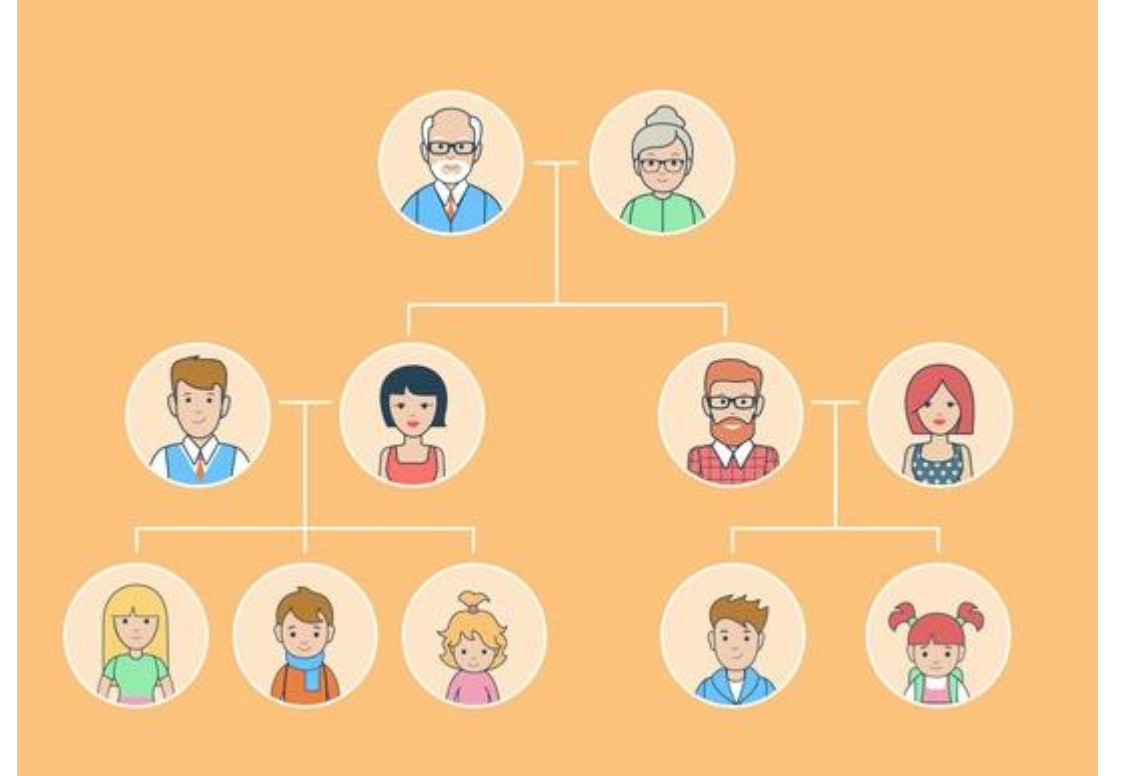
Hiyerarşik Model

- Ağaç yapısında:
 - Ebeveyn- çocuk ilişkisi
- Avantajları:
 - Basit
 - 1:M ilişki
 - Veri bağımsız
 - Veri Entegrasi kolay
- Dezavantajları:
 - Uygulamak zor
 - Esnek değil



Ağ Modeli

- Ağaç modelinde birden fazla root olması
- Avantajları:
 - Basit
 - 1:M ve N:M ilişkiler
- Dezavantajları:
 - Karmaşık
 - Kullanıcı dostu değil
- Bugün kullanılan Şema,DML,DDL gibi kavramların Tanımı Ağ Modeli sayesinde.



İlişkisel Model

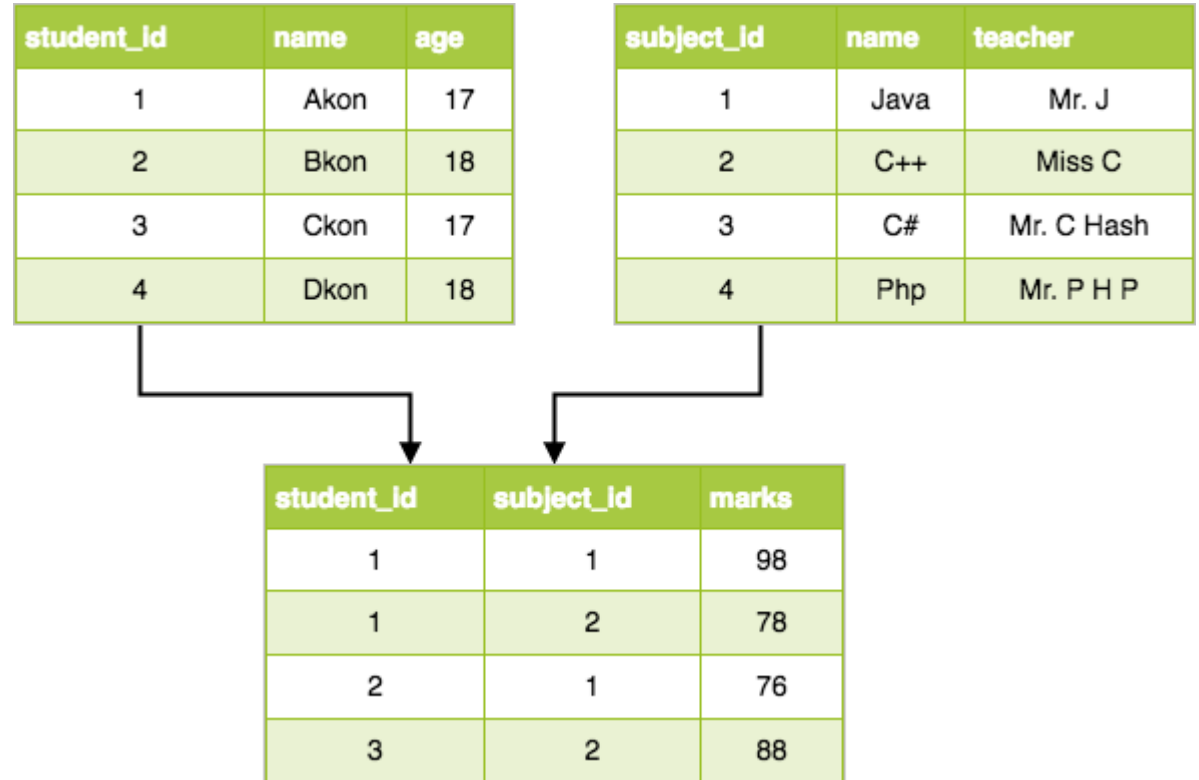
- Codd tarafından, 1970
- İlişkisel veri tabanındaki tüm bilgiler tablolardaki değerlerle tamamen tek ve tek bir şekilde temsil edilir.
- Her bir verinin (atomik değer), tablo adı, anahtar değeri veya sütun ismi gibi farklı parametreler kullanılarak erişilebilir olmalıdır.
- Eksik bilgiden kaynaklı oluşabilecek NULL değerler desteklenir.
- Veri tabanındaki açıklamalar; mantıksal seviyede, sıradan verilerle aynı şekilde gösterilir. Yani yetkili kullanıcılar, aynı ilişkisel dili kullanarak hem sıradan verilere hemde açıklamalara ulaşabilirler.
- İlişkisel bir sistem birkaç dili veya çeşitli metodolojileri destekleyebilir. Ancak veri tanımlama, görüntüleme, kısıtlama, yetkilendirme gibi çeşitli özelliklerin tanımlanabileceği en az bir dil olmalıdır.

İlişkisel Model

- Tüm görünümüler güncellenebilir olmalıdır.
- Yüksek seviyede; ekleme, silme, güncelleme işlemlerinin yapılabilir olmalıdır.
- Depolamada veya erişim yöntemlerinde bir değişiklik yapıldığında uygulama programlarında bir değişim yapılmasına gerek duyulmamalıdır.
- Tablo sütunlarında herhangi bir değişim yapıldığında uygulama programlarında bir bozulmanın olmaması gerekir
- Kısıtlamalar veritabanı tarafından tanımlanabilmelidir.
- Verinin hangi diskte saklandığı yada fiziksel olarak nerede bulunduğı ile ilgili işlemlerin DBMS tarafından yapılması gerekir.
- DBMS düşük seviye bir dil tarafından destekleniyorsa, yüksek seviyeli dil tarafında yazılmış kuralları uygulamak zorundadır.

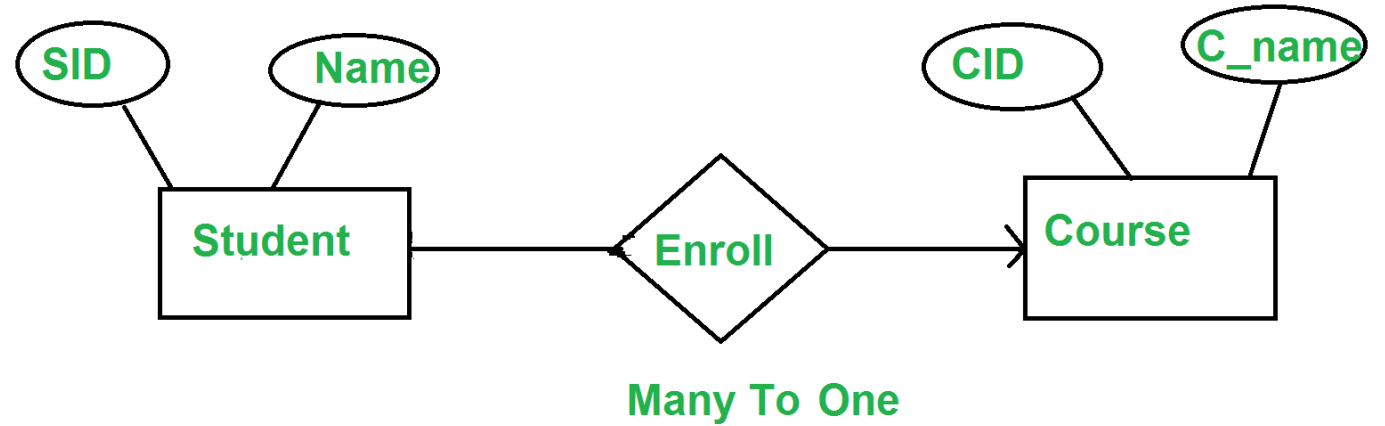
İlişkisel Model

- ilişkisel model en çok kullanılan veritabanı



ER Model

- Veritabanına geçiş sürecinde kolaylık sağlar
- Grafikselle gösterimi sayesinde anlaşılır, basit
- Varlık
 - Çalışan
 - Öğrenci
- Özellik
 - Çalışana yada öğrenciye ait isim
- İlişkiler
 - Çalışanın birimi
 - Öğrencinin aldığı ders

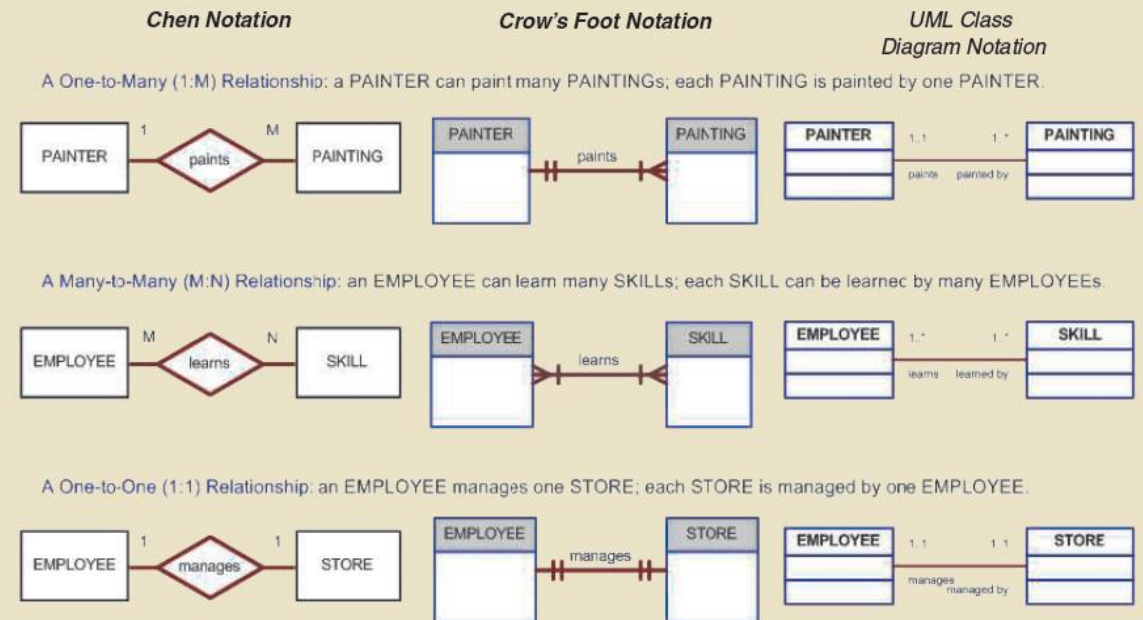


ER Model

3 Farklı gösterimi

- Chen
- Crow's Foot
- UML

FIGURE 2.3 THE ER MODEL NOTATIONS



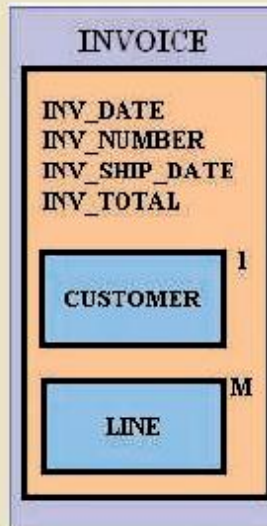
Nesneye Dayalı Model

- Er Modelindeki varlıklar, bu modelde Nesne olarak düşünülebilir
- Özellikler ise, nesnenin nitelikleridir.
- Benzer özellikteki nesneler gruplanır.
- Sınıflar hiyerarşik olarak düzenlenir.
- UML diyagramları kullanılarak gösterimler gerçekleştirilir.

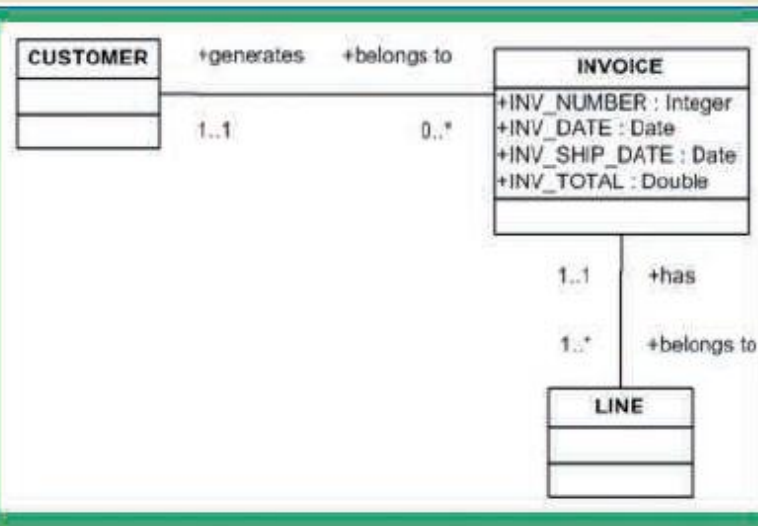
Nesneye Dayalı Model

FIGURE 2.4 A COMPARISON OF OO, UML AND ER MODELS

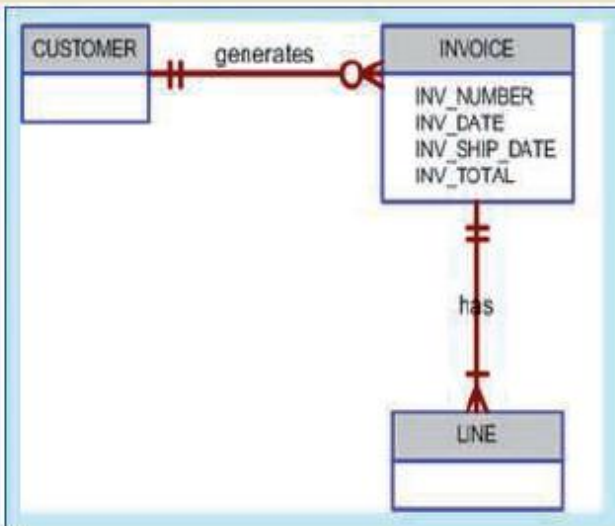
Object Representation



UML Class Diagram



ER Model



Nesne/İlişkisel ve XML veri Modeli

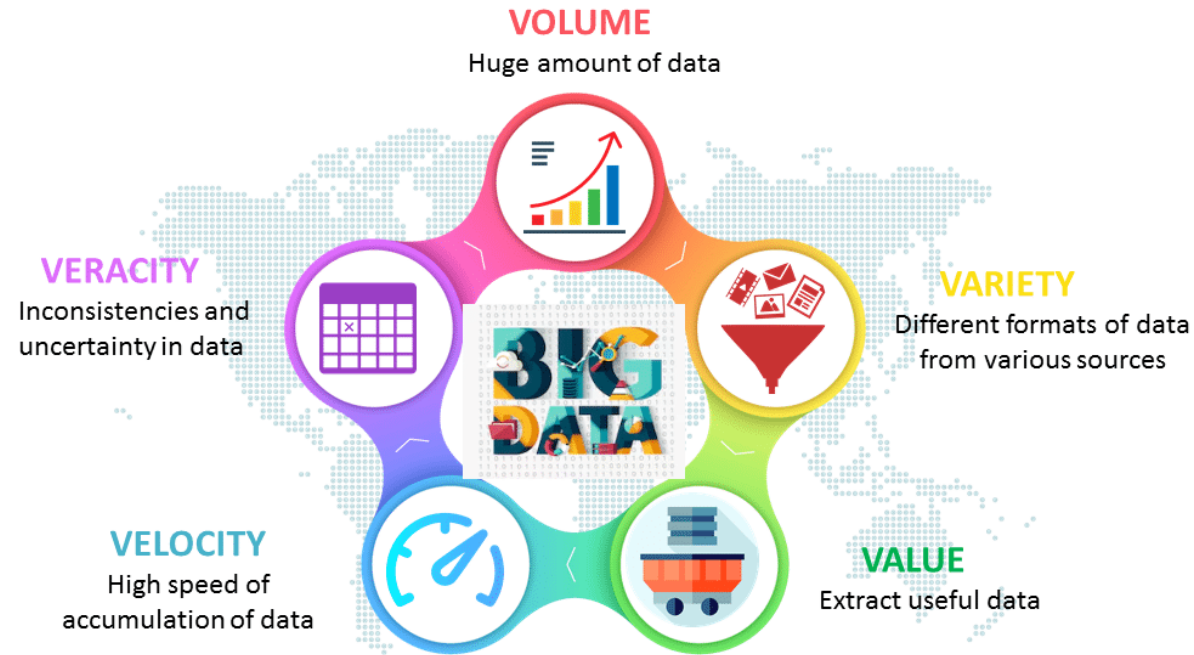
- Yapılandırılmamış veri alışverişi için ortaya çıkmıştır
- XML verilerinde içerik olarak; sözcük, belge, web sayfası...vb

NoSQL

Büyük Veri?

- Uzun yıllardır devam eden müşteri geçmişi
- Sensör verileri
- Sosyal medyadaki davranış kalıpları
- GPS verileri

Büyük veri ve V kavramı



Sık kullanılan teknolojiler

- Hadoop
 - Java tabanlı
 - Dağıtık çalışan
 - Hesaplama kapasitesine sahip
- MapReduce
 - API
 - Verinin paralel olarak dağıtılması
 - Hesaplama sonuçlarının birleştirilmesi
- NoSql
 - Dağıtılmış veritabanı sistemi

NoSql

- İlişkisel veri modelini kullanmazlar
- Dağıtık veritabanı mimarisini destekler
- Ölçeklenebilir
- Hata toleransına sahip
- Yüksek miktarda veri işleyebilen
- Önceliği işlem tutarlılığı yerine performansa verir

Applications that work best with NoSQL



Social

IoT

Web

Mobile



Enterprise

Document
database

FIGURE 2.5 A SIMPLE KEY-VALUE REPRESENTATION

Trucks-R-Us

Data stored using traditional relational model

DID	CERT1	CERT2	CERT3	DOB	LICTYPE
2732	80		95	1/24/1962	P
2946		92		4/11/1970	
3650	86			11/27/1963	R

- In the relational model:
 - Each row represents one entity instance.
 - Each column represents one attribute of the entity.
 - The values in a column are of the same data type.
- In the key-value model:
 - Each row represents one attribute/value of one entity instance.
 - The “key” column could represent any entity’s attribute.
 - The values in the “value” column could be of any data type and therefore it is generally assigned a long string data type.

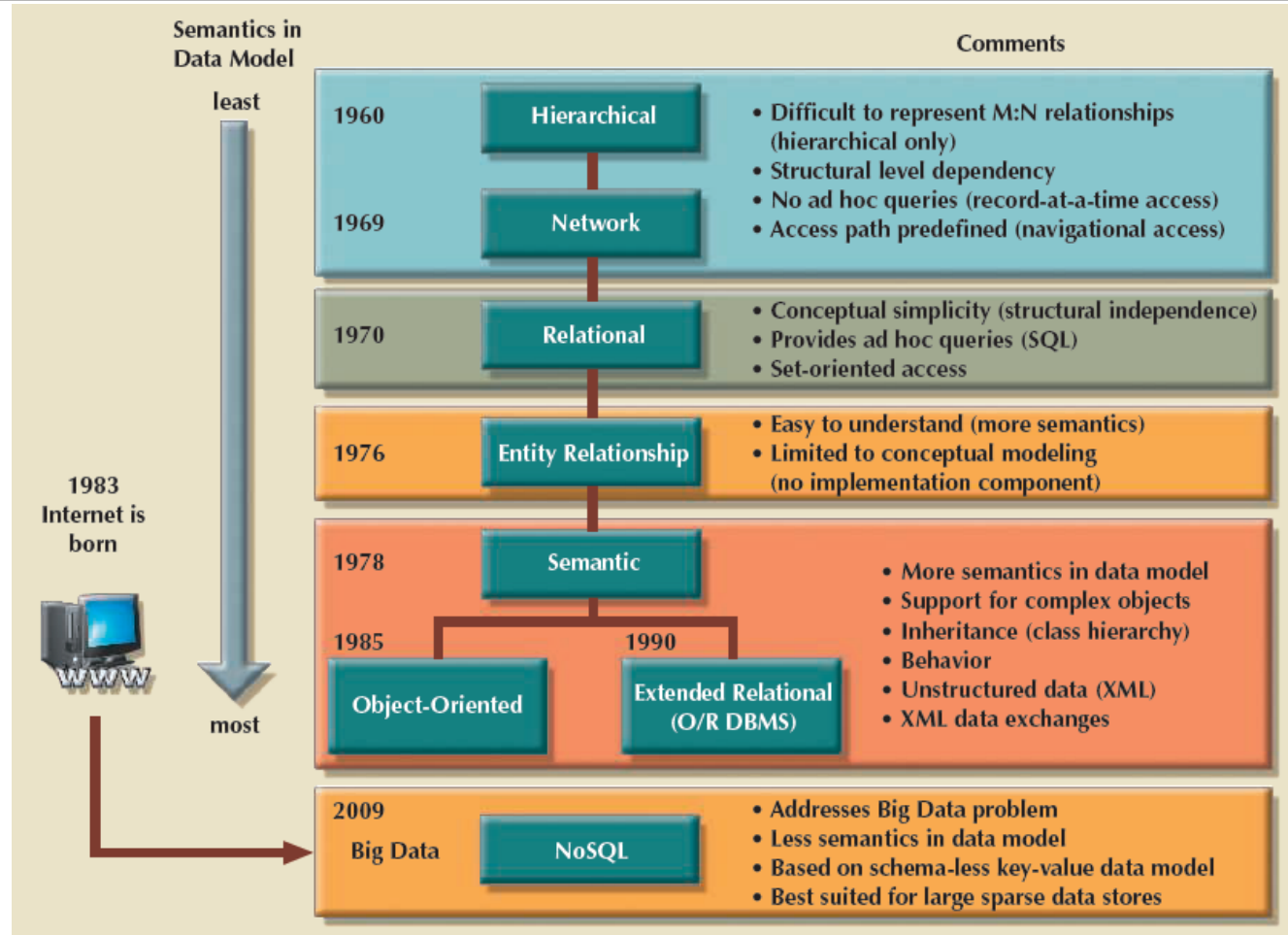
Data stored using key-value model

DID	KEY	VALUE
2732	CERT1	80
2732	CERT3	95
2732	DOB	1/24/1962
2732	LICTYPE	P
2946	CERT2	92
2946	DOB	4/11/1970
3650	CERT1	86
3650	DOB	11/27/63
3650	LICTYPE	R



Driver 2732

Veri Modellerinin Tarihsel Gelişimi



DATA MODEL	DATA INDEPENDENCE	STRUCTURAL INDEPENDENCE	ADVANTAGES	DISADVANTAGES
Hierarchical	Yes	No	<ol style="list-style-type: none"> 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. 	<ol style="list-style-type: none"> 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	<ol style="list-style-type: none"> 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. 	<ol style="list-style-type: none"> 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	<ol style="list-style-type: none"> 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. 	<ol style="list-style-type: none"> 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	<ol style="list-style-type: none"> 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. 	<ol style="list-style-type: none"> 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	<ol style="list-style-type: none"> 1. Semantic content is added. 2. Visual representation includes semantic content. 3. Inheritance promotes data integrity. 	<ol style="list-style-type: none"> 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	<ol style="list-style-type: none"> 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. 	<ol style="list-style-type: none"> 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 Tabanı Türleri

- Kullanıcı türüne göre
 - Tek kullanıcı
 - Çok kullanıcı
- Konumuna göre:
 - Merkezi DBMS
 - Dağıtık DBMS
 - Paralel DBMS
 - Client/Server DBMS

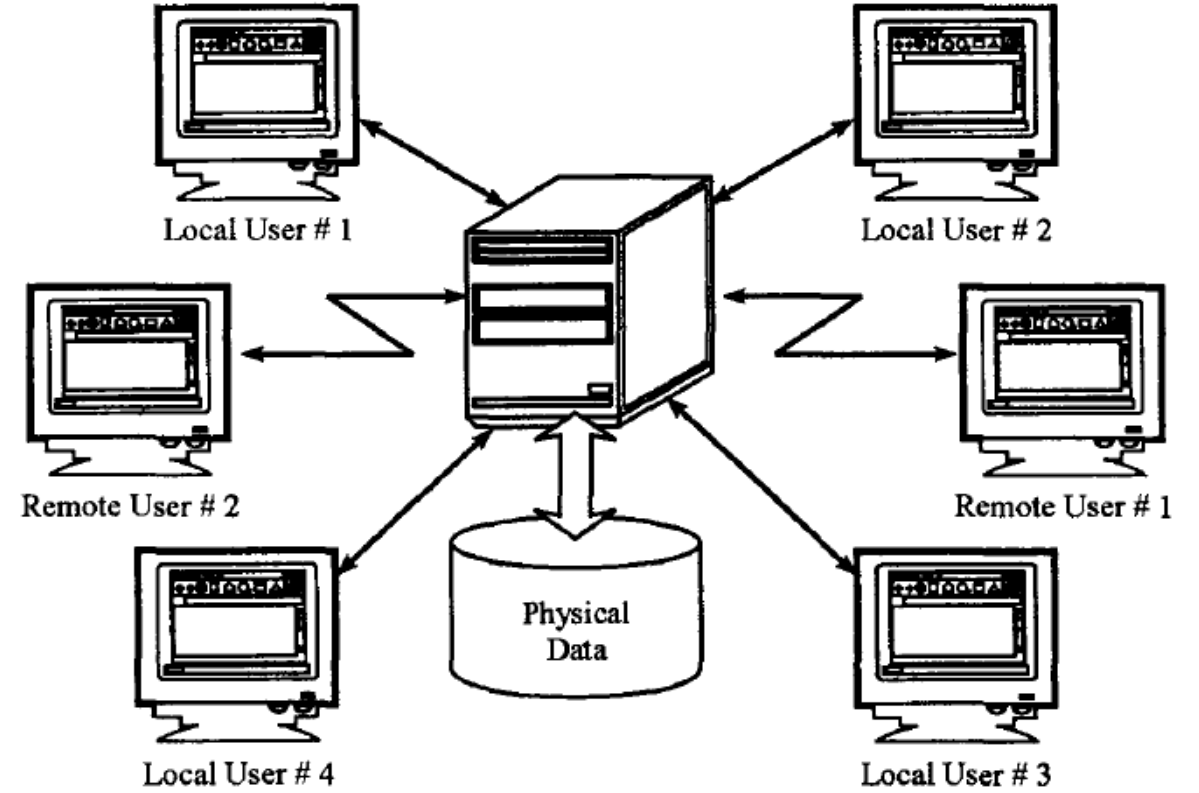
Merkezi DBMS

○Avantaj

- Güncelleme, yedekleme, sorgu, kontrol erişimi kolay
- Veritabanı boyutu, konumlandırılan bilgisayarı etkilemez

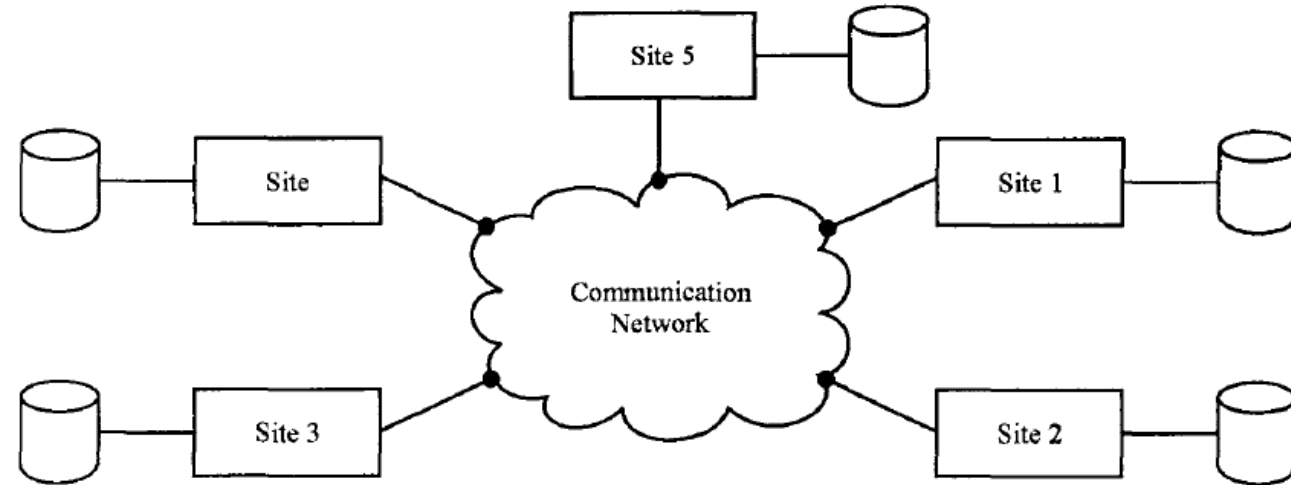
○Dezavantaj

- Bozulma durumunda tüm kullanıcılar etkilenir
- İletişim maliyetleri pahalı olabilir



Dağıtık DBMS

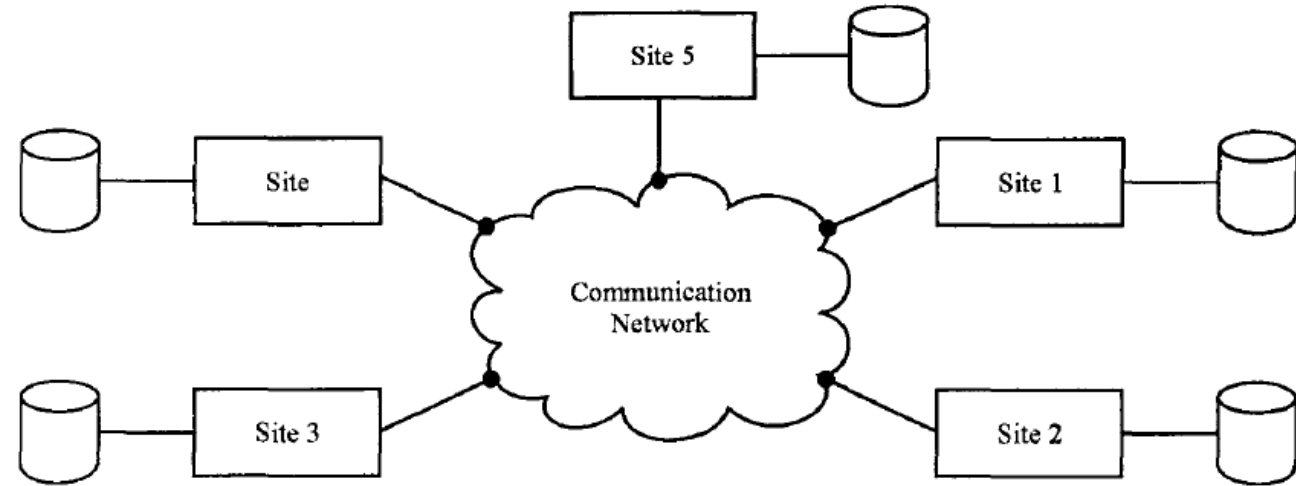
- Bir uygulama coğrafi olarak farklı makinelerle dağıtılan veriler üzerinde çalışabilir.
- Avantaj:
 - Farklı seviyelerde transperancy ile dağıtılmış verilerin yönetimi.
 - Artan Güvenilirlik ve Kullanılabilirlik.
 - Dağıtılmış sorgu işleme.
 - Performansı iyileştirme.
 - Geliştirilmiş ölçeklenebilirlik
 - Paralel değerlendirme.
 - Dağıtılmış veritabanı kurtarma.
 - Çoğaltılmış Veri yönetimi
 - Güvenlik
 - Ağ şeffaflığı.
 - Daha fazla verimlilik ve daha iyi performans sağlar.
 - Çoğaltma saydamlığı.



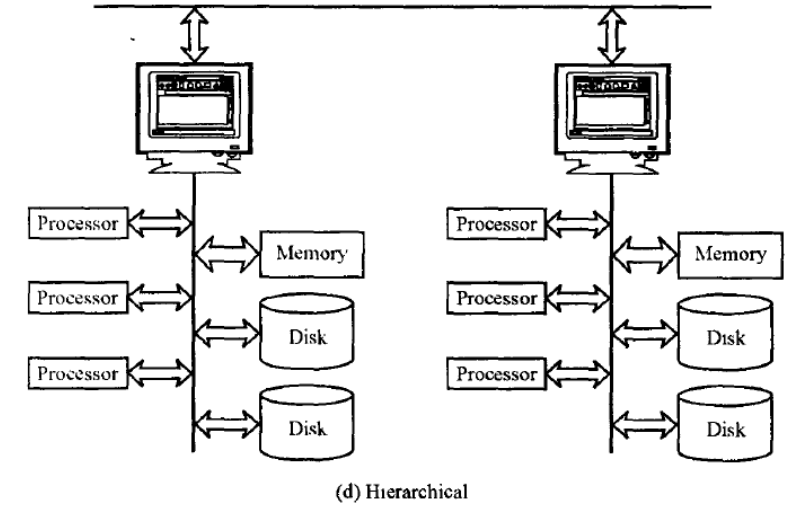
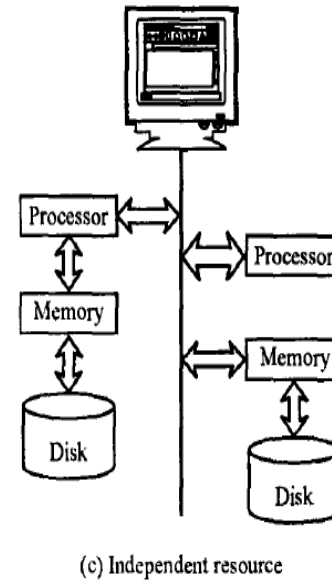
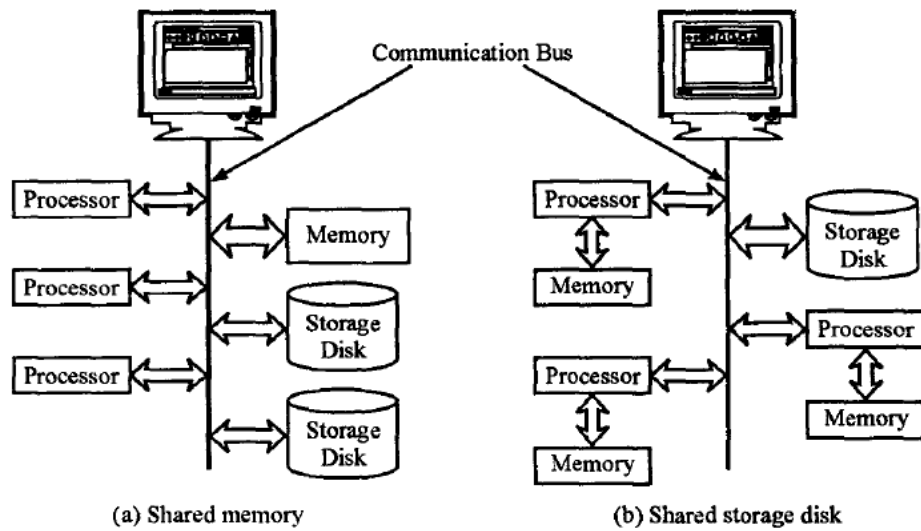
Dağıtık DBMS

○ Dezavantajları

- Birbirine benzemeyen makinenin bağlanması ile ilgili teknik sorun.
- Yazılım maliyeti ve karmaşıklığı.
- Veri bütünlüğü kontrolünde zorluk.
- Genel gider işleme.
- İletişim ağı arızaları.
- Başarısızlıktan kurtarma daha karmaşıktır.



Parallel DBMS



Paralel DBMS

○Avantaj

- Büyük veritabanlarını sorgulamak veya saniyede çok fazla sayıda işlem yapmak zorunda kalan uygulamalar için çok kullanışlıdır.
- Veri sunucusu sistemlerinde, belirli teknikler kullanılarak sistem hızlandırılır.
- Geçiş (belirli bir zaman aralığında tamamlanabilecek görev sayısı) ve yanıt süresi (yani, tek bir görevi tamamlamak için harcadığı zaman miktarı) gönderildiği zaman) çok yüksektir.

○Dezavantajları

- Başlangıç maliyeti vardır ve başlangıç zamanı gerçek işlem süresini gölgeleyebilir.
- Genellikle paylaşılan kaynaklara eriştiğinden, yavaşlama, paylaşılan veri depolama diskleri, sistem veriyolu vb. gibi yaygın olarak tutulan kaynaklar için mevcut işlemlerle rekabet ettiğinden, her yeni işlemin müdahalesinden kaynaklanabilir.

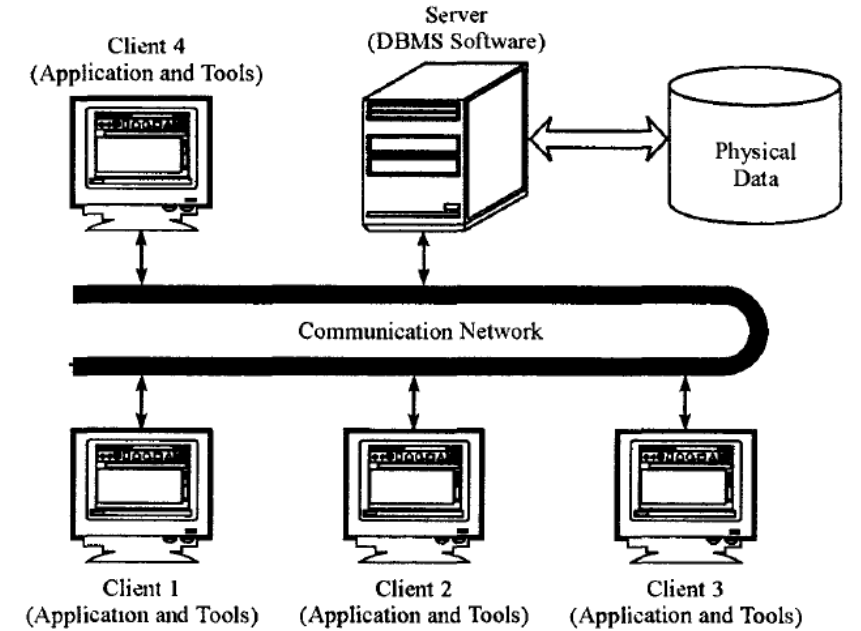
Client/Server DBMS

○Avantajları:

- Kullanıcıların daha verimli çalışmasını
- Mevcut verilerin daha iyi kullanılmasını
- Merkezi sisteme göre daha esnektir.
- Tek bir veritabanı (sunucuda) birkaç farklı istemci (uygulama) sistemi arasında paylaşılabilir.
- İstemci / sunucu mimarisi daha iyi bir DBMS performansı sağlar.

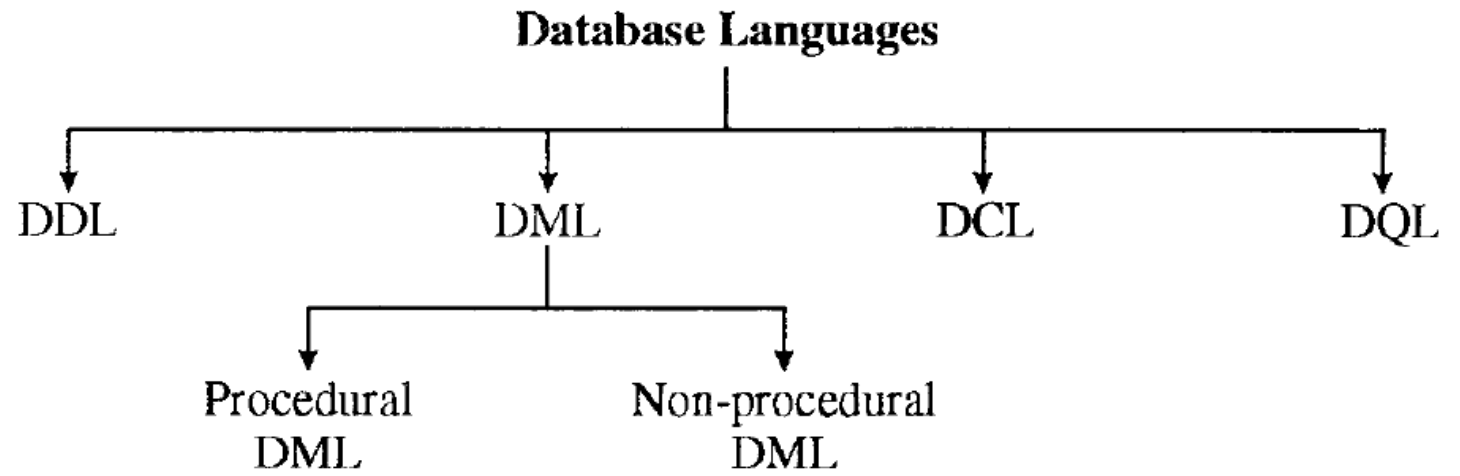
○Dezavantajları

- İşçi veya programlama maliyeti
- Ağ ortamları için performans izleme ve ayarlama ve güvenlik kontrolü için yönetim araçları eksikliği vardır.



Veritabanı Dilleri

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Data Query Language (DQL)



DDL(Data Definition Language)

Varlıkları oluşturmak için kullanılan dildir.

(i) CREATE: Veritabanında nesneler oluşturmak için.

(ii) ALTER: Veritabanının yapısını değiştirir.

(iii) DROP: Nesneleri verilerden silin.

(iv) TRUNCATE: Kayıtlar için ayrılan tüm alanlar dahil, tüm kayıtları bir tablodan kaldır.

(v) COMMENT: Veri sözlüğüne yorum ekleme

DML(Data Manipulation Language)

- Veritabanından veri alınması.
- Veriyi veri tabanından silme
- Yeni verilerin veritabanına eklenmesi
- Veritabanındaki verilerin değiştirilmesi.
- DML temelde iki türdür:
 - Prosedürel DML: Prosedürel DML'ler bir kullanıcının hangi Verilere ihtiyaç duyulduğunu ve bu verilerin nasıl elde edileceğini belirlemesini gerektirir.(SELECT value)
 - Procedürel Olmayan DML'ler: Bu DML'ler, bir kullanıcının bu verilerin nasıl alınacağını belirtmeden hangi verilere ihtiyaç duyulduğunu belirlemesini gerektirir. (SELECT *)
- SELECT, INSERT, UPDATE, DELETE, LOCK TABLE

DCL(Data Control Language)

- Veriye ve veritabanına erişimi kontrol eden SQL ifadelerinin bileşenleridir.
- COMMIT
- ROLL-BACK
- SAVE POINT
- GRANT/REVOKE
- SET TRANSACTION

DQL(Data Query Language)

Veritabanından veri almayı ve üzerine sipariş vermeyi sağlayan

SOL ifadesinin bir bileşenidir.

Sorgu: Bir sorgu bilgi alınmasını isteyen bir ifadedir.

